Between Discipline and Profession

A History of Persistent Instability in the Field of Computer Engineering, circa 1951-2006

by Brent K. Jesiek

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in
Science and Technology Studies

Gary L. Downey (Chair)
Janet Abbate
Daniel Breslau
Timothy W. Luke
Michael S. Mahoney (Princeton University)

December 13, 2006
Blacksburg, VA

Keywords:
history, computer, computing, design, technology, engineers,
engineering, engineering studies, discipline, profession, instability

Between Discipline and Profession

A History of Persistent Instability in the Field of Computer Engineering, circa 1951-2006

by Brent K. Jesiek

Abstract

This dissertation uses a historical approach to study the origins and trajectory of computer engineering as a domain of disciplinary and professional activity in the United States context. Expanding on the general question of "what is computer engineering?," this project investigates what counts as computer engineering knowledge and practice, what it means to be a computer engineer, and how these things have varied by time, location, actor, and group. This account also pays close attention to the creation and maintenance of the "sociotechnical" boundaries that have historically separated computer engineering from adjacent fields such as electrical engineering and computer science. In addition to the academic sphere, I look at industry and professional societies as key sites where this field originated and developed. The evidence for my analysis is largely drawn from journal articles, conference proceedings, trade magazines, and curriculum reports, supplemented by other primary and secondary sources.

The body of my account has two major parts. Chapters 2 through 4 examine the pre-history and early history of computer engineering, especially from the 1940s to early 1960s. These chapters document how the field gained a partially distinct professional identity, largely in the context of industry and through professional society activities. Chapters 5 through 7 turn to a historical period running from roughly the mid 1960s to early 1990s. Here I document the establishment and negotiation of a distinct disciplinary identity and partially unique "sociotechnical settlement" for computer engineering. Professional societies and the academic context figure prominently in these chapters. This part of the dissertation also brings into relief a key argument, namely that computer engineering has historically occupied a position of

“persistent instability” between the engineering profession, on the one hand, and independent disciplines such as computer science, on the other.

In an Epilogue I review some more recent developments in the educational arena to highlight continued instabilities in the disciplinary landscape of computing, as well as renewed calls for the establishment of a distinct disciplinary and professional identity for the field of computer engineering. I also highlight important countervailing trends by briefly reviewing the history of the software/hardware codesign movement.

Acknowledgments

First and foremost, I thank my family for their encouragement and understanding as I worked on this project, as well as my graduate degrees more generally. In fact, this dissertation may have never been completed without the extensive support – and gentle prodding – provided by my wife Julie and son Preston. Thanks also to my parents for their continued encouragement in all of my endeavors, no matter how seemingly far-flung.

I owe another debt of gratitude to my many friends and colleagues at Virginia Tech – for the most part, you know who you are. But I especially want to thank Jody Roberts for many conversations that helped enrich this document. I also acknowledge Shelli Fowler, who not only greatly encouraged and supported me as a graduate student, but who is also an all-around exemplary colleague and mentor. Following her model, I fully intend to pay the favor forward. Members of Virginia Tech's Research in Engineering Studies (RES) group, on the other hand, deserve mention for being willing and thoughtful commentators on an early draft of Chapter 2.

I am of course much indebted to each and every one of my committee members, who from the start saw the value and potential of this project, and helped nurture it along the way. Their willingness to thoughtfully engage with a large body of draft material and provide countless valuable suggestions made this a much better document. I very much look forward to continuing our conversation about this material in coming months and years.

Gary Downey, in particular, has been an outstanding committee member and chair throughout this process, and I thank him for both keeping me on schedule and pushing me to figure out where my analysis was headed. At countless points along the way, Gary helped me see the proverbial forest for the trees when I found myself neck-deep in historical details.

There are still others worthy of mention, including the many library staffers who retrieved or tracked down sources for me. Staff at the Charles Babbage Institute, National Academies, and IEEE History Center also deserve mention for responding to many queries and for finding and sending me relevant materials. Again, I thank you all.

Table of Contents

ABSTRACT II

ACKNOWLEDGMENTS IV

LIST OF FIGURES AND TABLES VIII

CHAPTER 1 – INTRODUCTION 1
- Research Questions and Objectives 4
- Historical Literature Review 5
- Theoretical Literature Review 9
- Methodology 14
  - Professional Publications 15
  - Trade Magazines 16
  - Curriculum and Model Program Reports 16
  - Other Sources 17
- Summary of Chapter Contents 17

CHAPTER 2 – FROM ENGINEERS AND COMPUTING TO COMPUTER ENGINEERING 22
- A Brief Early History of Electrical Engineering and Its Institutes 23
- Intersections of Expertise in Early Computer Development Projects 27
- Connecting the Islands: Early Steps toward a Field of Computing 34
- Mauchly, the ENIAC, and the Machine-Instruction Boundary 37
- The AIEE and Computing 41
- The IRE and Computing: From Technical Committee to Professional Group 46
- The Joint Computer Conferences 50
- Positioning Computers in Engineering 52
- Computer Engineering Identities 55
- Employing Computer Designers and Engineers 58
- The Relational Ontology of Computer Engineering 60
- Conclusion 65

CHAPTER 3 – A SYSTEM OF PROFESSIONAL SOCIETIES: NEGOTIATING THE SOCIOTECHNICAL SETTLEMENTS 67
- The Early History of the ACM: "What Computers Do" 68
- IRE-PGEC: The Voice of the Computer Engineering Profession 79
- The AIEE CDC: Committee-Bound and Power Industry-Oriented 85
- Merger, Identity, and Scope: Forming the IEEE Computer Group 91
- Stabilizing the System: The Joint Computer Conferences and Committees 97
- "Is It Overhaul or Trade-In Time?": The 1959 Rand Symposium 103
- From the NJCC to AFIPS: Preserving Stability in the System 106
- Conclusion 111

CHAPTER 4 – DICHOTOMOUS DEVELOPMENTS IN THE EARLY COMPUTER FIELD: PROFESSION, TECHNOLOGY, AND EDUCATION, C. 1955-1963 114

Part I – Mirrored Dichotomies: Hardware/Software and Engineer/Programmer 115
Computer Engineering Identities: From Interpretive Flexibility to Stable Jurisdictions 116
Divisions of Labor and Hierarchies of Design: Bounding and Segmenting Computer Engineering 121
The Hardware/Software Ensemble: Constructing and Questioning the Dichotomy 126
Artificial Barriers versus Integration: Carr and Gorn on the Boundaries 132

Part II – Education and Discipline: (Re)Negotiating the Boundaries of Computing 141
Computer Education: An Inchoate Early Assortment of Courses and Curricula 141
Toward a Scientific Discipline of Computing 150
Educating Computer-Using Engineers 157
Educating Computer Engineers and Designers 163
Conclusion 168

CHAPTER 5 – COMPETING IMAGES OF DISCIPLINARITY: COMPUTER SCIENCE, COSINE, AND COMPUTER ENGINEERING 171

Part I – (The) Computer/Computing/Information Science(s): A Formative First Decade 172
Defining Computer Science as a Discipline 173
Positioning and Settling Computer Science 178
Instituting Computer Science – Departments and Programs 182
Instituting Computer Science – Courses and Curricula 187

Segue – On the Boundaries of Computer Science and Engineering 192

Part II – Shifting Disciplinary Images: From Computer Science to Computer Engineering in Electrical Engineering 198
Bringing Computer Science Into the Fold: Lotfi Zadeh at Berkeley and Beyond 198
Engineering Images of Computer Science: Discipline, Department, and/or Program? 204
An Introduction to the COSINE Committee: Historical Origins and Trajectory 209
COSINE, The Early Years: Promoting "Computer Sciences in Electrical Engineering" 214
Transitional COSINE: From Computer Science to Computer Engineering 218
COSINE and Computer Engineering: Expanding EE From the Inside Out 224
Evaluating the "Impact" of COSINE and the Growth of Computer Engineering Education 228
Conclusion 233

CHAPTER 6 – JANUS-FACED TECHNOLOGY, JANUS-FACED FIELD: (RE)NEGOTIATING THE SOCIOTECHNICAL SETTLEMENTS 239
From Computer Group in Crisis to a More Autonomous Computer Society 241
Expansion and Identity, Merger Talks and Mediation (Part I) 247
Sociotechnical Expansion and Mediation: New Committees for Emergent Fields 252
Computer Architecture 254
Software Engineering 257
Microprogramming 259
Expansion and Identity, Merger Talks and Mediation (Part II) 262
Conclusion 269

CHAPTER 7 – BRIDGING THE TAR PIT?: CONSTRUCTING CSE AND COMPUTING EDUCATION, CIRCA 1974-1991 273
Claiming CSE: The Computer Society Makes Moves in Education 278
Bridging the Tar Pit?: Toward a Curriculum in Computer Science and Engineering 282
A(n Engineer's) Curriculum in Computer Science and Engineering 287
Supporting Curricular Reform in Electrical Engineering Education 290
Disciplining CSE: Taxonomies, COSERS, and Encyclopedias, oh my! 293
Managing Complexity: The Hybridization of Hardware and Software Engineering 299
Research Directions in Computer Engineering: (Re)Defining the Discipline 303
From Curriculum to Program: The Engineers Revisit CSE Education 306
Engineering Accreditation and The Discursive Politics of Professional Certification 310
CSAB and CSAC: Independent Accreditation for an Independent Discipline 314
The Diversification of Computer Science Curricula 317
From Discipline in Crisis to Computing as a Discipline 319
The Shifting Institutional Landscape of Computing 325
Conclusion 327

EPILOGUE – COMPUTING CURRICULA AND CODESIGN: DIVERGENT PATHWAYS? 329
Software/Hardware Codesign: Blurring the Sociotechnical Boundaries 333
From Software/Hardware Codesign to Sociotechnical Codesign 338

APPENDIX A – ACRONYMS AND ABBREVIATIONS 340

BIBLIOGRAPHY 343

List of Figures and Tables

Figure 4.1 – Functions and Responsibilities of Computer Design Groups (Phister, 1958, p. 3) 122

Figure 4.2 – "Mesa Men" (Mesa Scientific Corporation, 1964) 131

Table 5.1 – Containment Table for Computer Science (Zadeh, 1968b, p. 913) 203

Table 5.2 – COSINE Committee Reports 213

Figure 6.1 – Hardware vs. Software: The Two Faces of Computers (Jensen, 1973, p. 14) 238

Figure 7.1 – Distribution of Computer Science Programs with Present and Projected Accreditation 316

Chapter 1
Introduction

In an important sense, this dissertation lives between the fourth and fifth floors of Virginia Tech's Newman Library. Organized according to the venerable Library of Congress (LoC) classification scheme, the library's fourth floor is dominated by the holdings for the call letter Q, which represents a wide swath of the physical, natural, and biological sciences, as well as mathematics. And it is within the QA subclass for Mathematics that we find further relevant divisions, including QA71-90 for "Instruments and Machines," and then QA75-76.95 for "Calculating Machines," QA75.5-76.95 for "Electronic Computers" and "Computer Science," and QA76.75-76.765 for "Computer Software" ("Q – Science," n.d., p. 3). The library's fifth floor, on the other hand, is largely filled with holdings for the call letter T, which covers Technology generally and an array of engineering subjects more specifically. Digging deeper still, we find that within the TK subclass (which itself covers electrical engineering, electronics, and nuclear engineering) a smaller sliver of materials in the TK7885-TK7895 range has been earmarked for "Computer Engineering" ("T – Technology," n.d., p. 11). On the surface, some may find it puzzling that the LoC scheme segregates "Computer Engineering" from seemingly related subjects such as "Electronic Computers," "Computer Science," and "Computer Software." However, this apparent quirk in classification begins to hint at a key goal of this project, namely to develop a better understanding of the unique historical position of computer engineering "between" electrical engineering, on the one hand, and mathematics and computer science, on the other.[1]

Electrical engineering – which is often viewed as the main "parent" discipline of computer engineering – is a broad area of activity that melds engineering method with electrical

[1] This example also hints at some of the potential implications of these divisions. Perhaps most obviously, the physical separation of these books and journals means that researchers from one field or another may be less likely to stumble upon potentially relevant texts that happen to be shelved in another area or even on a different floor.

and electromagnetic phenomena. Beginning in the early 1950s, computer engineering started to gain an identity that was at least partially distinct from electrical engineering, and since then the dominant image of the field has remained closely linked to computer "hardware."[2] More specifically, the line between electrical and computer engineering is often negotiated at the lowest levels of computer structure, where electrical engineers tend to concern themselves with the physical properties of integrated circuits and other microelectronics, leaving computer engineers to focus on the more abstract upper levels of computer technology, such as logic design, system design, and low-level programming. The Institute for Electrical and Electronics Engineers (IEEE) has been the professional society with the closest ties to computer engineering, especially via the IEEE Computer Society (IEEE CS) subgroup and its historical predecessors. Computer science, on the other hand, emerged in the 1950s and 1960s with significantly more multidisciplinary origins, and with a stronger orientation toward mathematics, algorithms, computer programming, and "software." The Association for Computing Machinery (ACM) has been – and largely remains – the major professional organization for computer scientists.

As the preceding passage suggests, the dominant image of computer engineering and computer science as respectively linked to hardware and software is deeply rooted in history. Yet much of the research on which this project is based suggests that these generalizations hide a much more complicated – and interesting – historical reality. From the earliest days of the field there has been wide recognition among computer professionals that the divide between computer software and hardware is anything but fixed or easily identifiable. Some of the earliest discussions about the ambiguous and shifting nature of this boundary date back to the 1940s and 1950s, and similar commentaries have periodically cropped up time and again, despite ongoing and major changes in computer technology. In his 1976 textbook, for example, computer

[2] My use of the term "dominant images" in this dissertation is significantly informed by the prior work of Gary Downey (1998, pp. 5-6). As nicely summarized by his colleague Juan Lucena, "Dominant images create expectations about how individuals in that location are supposed to act or behave. In this new concept of culture, the image remains the same over a period of time, while individual or group reactions to the image's challenges might differ. When challenged by the same image, individuals or groups resist, accommodate, fully accept, or experience ambiguity in different ways" (Lucena, 2005, pp. 6-7). While my own account places somewhat less explicit emphasis on "culture," I use the concept of "dominant images" to highlight how competing definitions and conceptions (or "images") of "computer engineering" and "computer engineers" have developed and circulated over time, in the process challenging many actors and groups. My use of the term "field," on the other hand, is used very generally throughout this document in reference to a given domain of activity, and without any major theoretical or normative assumptions. I use alternate concepts such as "discipline" and "profession" more judiciously, as outlined in detail below.

scientist Andrew Tanenbaum noted in an introductory chapter on the historical development of computer organization and architecture that “one man's hardware is another man's software,” and he went on to describe the boundary between these two domains as “arbitrary and constantly changing” (p. 11).

Yet Tanenbaum’s remarks about the software-hardware boundary – or the lack thereof – seem to stand in marked tension with the mid-1970s relationship between computer engineering and adjacent fields, such as computer science. In a December 1975 issue of the IEEE journal *Computer* – which was dedicated to the topic of “computer education” – guest editors David Irwin and C. V. Ramamoorthy summarized that “from the educator's point of view, perhaps no problem is so apparent as that of overcoming the dichotomy between computer science and computer engineering” (Irwin and Ramamoorthy, 1975, p. 27). And in another article in the same journal issue, engineer Michael Mulder used the evocative image of the “tar pit” to describe the difficult meshing of computer science and computer engineering curricula (Mulder, 1975, p. 28). How might we explain this apparent tension between increasingly blurred technological boundaries coexistent with deeply entrenched – and perhaps even conflicting – disciplinary and curricular boundaries? One might postulate that these tensions were eventually worked out through some combination of technological and disciplinary change, limiting their significance to the historical moment and actors identified here. Yet ample evidence suggests that no clear resolution was achieved, and that tensions like these can be traced throughout much of the history of computing, even to the present.

To take a second – and more recent – example of these tensions, a task force representing both the ACM and IEEE-CS was formed in the late-1980s to develop a new set of curricular recommendations for what the committee came to call “the discipline of computing” (Tucker, et al., 1991). This seemingly unprecedented move – toward a more integrated “meta-discipline” of computing – looked like an important step toward overcoming some of major the social and technological rifts that had long persisted in the various computing fields. Yet jumping ahead roughly a decade, we find a very different set of recommendations coming out of the Computing Curricula 2001 effort. Also a joint venture of the IEEE-CS and ACM, this new task force was charged with reviewing and updating the 1991 report. However, the group quickly splintered out to develop separate reports with separate recommendations for five different disciplinary domains, namely computer science, computer engineering, information systems, software

engineering, and information technology. In justifying these divisions, the authors of the computer engineering volume argued that while efforts to mesh or merge the computing curricula "may have seemed reasonable in the past, there is no question that computing in the twenty-first century encompasses many vital disciplines with their own identities and pedagogical traditions" (Hughes, et al., 2004, p. 1). Those familiar with the professional and disciplinary tensions that marked earlier eras of computing will likely read such passages with a sense of déjà vu.

Research Questions and Objectives

In order to better understand the types of trends and tensions outlined above, this project uses a historical approach to study the origins and trajectory of computer engineering as an area of academic and professional activity, in the United States context, and from the pre-history of the field in the 1940's and 1950's to the present. Expanding on the general question of "what is computer engineering?," the project investigates what counts as computer engineering knowledge and practice, what it means to be a computer engineer, and how these things have varied both across time and space and between various publication outlets, actors, and groups. In addition, this dissertation pays close attention to the creation and maintenance of the social and technological boundaries that have historically separated computer engineering from adjacent fields, such as electrical engineering and computer science. In addition to the academic sphere, I also pay close attention to industry and professional societies as other sites where this field originated and developed. The evidence for my analysis is largely drawn from journal articles, conference proceedings, trade magazines, and curriculum reports, supplemented by a range of other primary and secondary sources.

In summary, the account that follows documents how each step in the historical development of computer engineering has involved important social and technical negotiations, some managed within the field, and some requiring engagement and even conflict with the representatives of adjacent fields. Central to this project is the idea that carefully and closely exploring the long sequence of dilemmas over what it means to be a computer engineer or computer scientist – or alternatively, over where to draw the line between software and hardware, theory and design, and/or science and engineering – can reveal key insights about the past, the present, and even the imagined future of computing. My analysis also plays particularly

close attention to the ongoing failure of computer engineering to be clearly identified as either an independent discipline or branch of the engineering field, thereby contributing to the field's long and persistent instability.

The larger significance of this project is three-fold. First, it makes documentary contributions to engineering studies and the history of computing, two areas of scholarship that have dealt neither directly nor extensively with computer engineering. Second, the project uses leading edge theory and method – drawn from the literature on professions, disciplines, and Science and Technology Studies (STS) – to argue that the existence of computer engineering has involved the ongoing and active "co-production" of the social, material, and epistemological. Further, the project hypothesizes that the history of computer engineering is a history of persistent instability, with the field's very existence requiring ongoing efforts to align diverse elements, such as professional and disciplinary identities, organizations, computing technologies, and bodies of knowledge. Third and finally, studying the foundations and trajectory of computer engineering can suggest possibilities for transformation and reform, both in computing generally and computer engineering specifically. More specifically, I claim that assessing both dominant and alternative ways of organizing computing disciplines and technologies can reveal important new pathways toward more thoroughly contextualized, reflective, and socially responsible cultures of computer design and use.

Historical Literature Review

Given that this project contributes to a number of major bodies of scholarship, including the history of engineering and engineering studies, I begin my literature review with a survey of relevant work in these areas. In most general terms, some important historical work has been done on the professional and disciplinary development of various engineering fields, but it remains somewhat scant. Notable exceptions include Layton's well-known *The Revolt of the Engineers* (1971), which is largely premised on the claim that understanding the historical development of the engineering profession demands that we analyze ongoing efforts to negotiate the boundaries between engineering, science, and business. Reynolds (1986) follows similar themes in his detailed history of chemical engineering, and he places particular emphasis on how early definitions of this particular field were not so much about tensions between business and science, but rather about how chemical engineers both worked their way into management and

came to identify with management interests. Vincenti's (1990) case studies from the history of aeronautical engineering, on the other hand, approach the history of engineering from a somewhat different angle by shedding light on the important role that "design hierarchies" often play in engineering work.

Moving closer to the subject of this dissertation, a handful of authors have developed general histories of American electrical engineering. Texts by Ryder and Fink (1984) and McMahon (1984), for example, were published with IEEE support around the time of the Institute's centennial anniversary. The former stands as a broad yet somewhat celebratory history of electrical engineering in the United States, while McMahon's volume provides a more critically engaged history of electrical engineering as a profession, with particular emphasis on the IEEE and its predecessor societies. As McMahon claims, looking at the history of the relevant professional societies can provide a window into "the state of the profession" (p. xiii), although his account also discusses the pivotal role of industry and the academy in the development of electrical engineering (p. xiv). And while both Ryder and Fink and McMahon comment on the evolving relation of engineers and computing, they fail to provide in-depth discussions about the development of computer engineering as a field, and they do not seriously engage with issues of disciplinary identity or tendencies toward disciplinary fragmentation.

The history of computing is another large and growing body of scholarship that is relevant to my project, although relatively little work in this area has focused on the disciplinary landscape of computing. For example, historian Paul Ceruzzi's otherwise wide-ranging *History of Modern Computing* (2003) devotes just a few pages to the origins and early history of computer science, and entirely avoids discussing the development of computer engineering as a distinct field (pp. 101-103; 201-203). By contrast, historical work on well-known inventors, devices, companies, and sub-industries is somewhat more common, both in Ceruzzi's work and beyond.[3] Surveying the history of computing literature, Mahoney nicely summarizes this trend when he notes that "[b]iographies of men or machines – some heroic, some polemical, some both – are a prominent genre, and one reads a lot about 'pioneers'" (1988, p. 114). Calling for a "decentering of the machine," Mahoney has identified a number of under-explored topics in the

[3] The mainstream history of computing literature is well represented by texts such as those authored by Campbell-Kelly and Asprey (1996) and edited by Akera and Nebeker (2002) and Rojas and Hashagen (2000). Although beyond the scope of this review, I periodically turn to these texts in later chapters for various historical details.

history of computing, including most notably for this project the software-hardware relationship and the history of computing disciplines and institutions (1988, 2004a). Noting “the need for histories of the main communities of computing,” he offers an important follow-up question: “How has the balance of professional power shifted among these communities, and how has the shift been reflected in the technology?” (1996).

Mahoney's own historical studies of the field of software engineering stand as important attempts to grapple with these topics (1990; 2004b). Especially noteworthy here is the author's interest in tracing out ongoing efforts to establish both a canonical history for software engineering and a common “agenda” for its practitioners. As Mahoney argues, “Software engineering began as a search for an engineering discipline on which to model the design and production of software” (2004b, p. 17), and elsewhere he describes at length how the proponents of the emergent field drew inspiration for their endeavor from domains as diverse as applied science, mechanical engineering, and industrial engineering (pp. 9-16). And while Mahoney’s historical account also hints at persistent tensions between the image of software engineering as either a discipline or profession, he leaves many open questions about the extent to which such tensions have inhibited the development of this particular field.

Other secondary sources have paid only modest attention to the historical emergence and ongoing development of other relevant fields, such as computer science and computer engineering. Wildes and Lindgren (1985) and Guttag (2005), for instance, help fill in some important pieces of this puzzle via their institutional histories of electrical engineering and computer science at MIT, although the wider relevance of their accounts is limited by their site specificity. Following a parallel line of inquiry, Aspray’s (2000) in-depth analysis of the early decades of computing at five major universities – namely MIT, Harvard, the University of Pennsylvania, Columbia, and Princeton – provides other clues about how the various flavors of computer research and education emerged and evolved in the American academic context. And while Aspray’s analysis is primarily focused on discussing whether early entry into computing provided these schools with a “competitive advantage,” he repeatedly touches on the disciplinary tensions that were often in play, especially as mathematicians and electrical engineers, and later computer scientists, staked out their claims.

Luiz Ernesto Merkle’s research on computing-related disciplines stands as another important contribution in this area (Merkle, 2001; Merkle and Mercer, 2002). One important

facet of Merkle's work centers on its historical exploration of the many fields that fall under the broad "informatics" or "computing" umbrella, ranging from computer engineering and computer science to human-computer interaction (HCI) and information systems (IS). Hence, Merkle's work draws our attention to historical shifts in – and struggles over – the disciplinary and professional boundaries of computing. In fact, he and co-author Robert E. Mercer explicitly argue that computing specializations and subfields become "reified across educational institutions and their enacted curricula, across industry and commerce and their organizations, and across governments with policies and resources" (Merkle and Mercer, 2002, p. 92). Following this line of reasoning, these authors go on to claim that the many fields and subfields of informatics have historically tended toward over-specialization and disciplinary exclusivity, thereby hampering the types of cross-disciplinary collaboration and pollination that Merkle and Mercer clearly favor (Merkle and Mercer, 2002, p. 92).

The historical review and forward-looking vision presented by these authors resonates with my own work. However, Merkle's historical research is largely limited to a high-level review of key professional organizations and curriculum reports, setting aside important questions about how and why various computing fields have been linked to particular disciplinary knowledge claims and computer technologies. In addition, his ambitious efforts to both locate and re-theorize the "human" and "social" dimensions of computing lead him to concentrate much of his analysis on domains such as HCI and semiotics, leaving fields such as computer engineering under-analyzed. Hence, Merkle begins with many source materials and questions that are relevant to this dissertation, but he follows them in very different directions.

Casting a wider net in the history of computing reveals other texts that are generally relevant to this project. In one of his earlier articles, for instance, Ceruzzi looks at the "co-evolution" of electronics technology and computer science in the 1940-1975 period (1989). In doing so, he provides a lengthy description of the "continuous and reciprocal interaction between electronics and computing" (p. 257), leading him to important insights regarding the role of technological change in the development of computing disciplines. As Ceruzzi argues, electrical engineering initially "took over" the work of those involved in computing, but the tables later turned as "the science of computing 'took over' the discipline of Electrical Engineering, in the sense that its theory of digital switches and separation of hardware and software offered EE a guide to designing and building ever more complex circuits" (p. 257). While such claims are

provocative, the author can be critiqued for glossing over the tripartite relationship linking electrical engineering, computer engineering, and computer science. Further, he assumes an oversimplified distinction between software and hardware, thereby downplaying both the multi-level complexity of this boundary and the extent to which it is has both shifted and been contested over time.

Tracy Kidder's Pulitzer Prize winning *The Soul of a New Machine* (1981), on the other hand, documents the design and building of the new "Eagle" minicomputer system at a major American computer manufacturer in the late 1970s. In this journalistic-styled account, Kidder provides us with a rare glimpse of computer engineers and computer scientists working in their native corporate habitat. For starters, Kidder excels at explaining arcane computer concepts to general audiences – a valuable lesson given my own desire to produce a readable and accessible historical narrative. But even more importantly for the project outlined here, Kidder's text closely follows the trials and travails of "the Hardy Boys" and "Microkids," the two main groups of computer engineers who were responsible for designing the hardware and low-level microcode of the Eagle, respectively. In following these groups, the author reveals the deep interplay of social, organizational, and technical divisions of labor in the building of a new computer. In addition to highlighting how new technologies and unconventional management and design techniques seemed to be inaugurating a new phase of development for America's high-technology industries, Kidder's account provides an important historical snapshot of the interplay of the various fields and subfields of computing within a corporate context.[4] By accounting for the emergence and persistence of the major sociotechnical boundaries that have long separated computer engineers from both one another and other computer professionals, this dissertation fills in a key part of the historical backdrop against which Kidder's story unfolds.

Theoretical Literature Review

The chapters that follow also draw on and inform a number of bodies of theoretical literature. In this section I introduce theoretical work on the social and historical studies of professions and disciplines, the concept of "co-production," and discourse. It is first worth noting that a handful of writers have already done work on issues of professionalism in the context of

[4] I draw here from a recent book review by Moon (2004) that nicely summarizes Kidder's analysis and major claims. She also makes a strong case for the continued significance of this book, both for historians of technology generally and for historians of computers and computing specifically.

computing. Historian and social scientist Nathan Ensmenger, for example, nicely documents and analyzes how various segments of the computer field dealt with the "question of professionalism" in the 1950s and 1960s (2001). Yet Ensmenger largely avoids critically retheorizing professionalism or connecting it with discipline building, focusing instead on how computer scientists and programmers understood professionalism and used it to their strategic advantage. Computer scientist Stuart Shapiro, on the other hand, has comprehensively reviewed the many different models of professionalism that have been applied to the various fields and subfields on information technology, with a focus on both accounting for the historical lack of a dominant model and suggesting alternate ways of understanding what it means to engage in professional computing practice (1994). While Shapiro's analysis is especially useful in pointing us toward issues such as the persistent gap between science- and engineering-based conceptualizations of the various computing fields, the present analysis requires a somewhat more substantial body of theory than Shapiro provides.

For additional theoretical support I turn to Andrew Abbott, whose work on the professions has largely been focused on traditional subjects such as medicine and law, albeit with some forays into the so-called "information professions" and some examples drawn from engineering and computing. In *The System of Professions* (1988), Abbott resists framing professions in monolithic terms or as "silos," instead claiming that they are both located within larger systems and defined relationally. Central to understanding this systems-oriented view of professionalization and professional development is the author's concept of "jurisdiction," or "the link between a profession and its work" (p. 20). For Abbott, jurisdictions are by definition "strong" and "exclusive," and jurisdictional control can be negotiated in various contexts. The public and legal realms play roles in this process, but Abbott emphasizes worksites as pivotal for maintaining professional jurisdictions, especially through ongoing efforts to control work tasks. The academic context is also framed as a context where some degree of professional legitimation, research, and instruction can occur (p. 56-57). More specifically, Abbott notes that the academy is often where the most abstract forms of professional knowledge are cultivated and transmitted, although Abbott clarifies that "professional education takes place in institutions controlled by the professions" (p. 205).

Chaos of Disciplines (2001), on the other hand, reflects Abbott's engagement with issues of discipline formation and disciplinarity, with a primary focus on the social sciences. The author

frames academia as the main locus of disciplinary development, with departments and graduate degree programs standing as pivotal hallmarks of disciplinarity in the American context, albeit with national disciplinary societies playing a strong supporting role (pp. 125-126). In further contrast to his work on professions, Abbott uses the metaphor of "settlement" to describe the "interactional field of academic disciplines" (p. 136), where claims to academic work and disciplinary bodies of knowledge are often complex and shifting, in no small part due to "an extraordinary interpenetration of settlements" (p. 142). Building on his earlier "ecological model for professional knowledge" (p. 136, fn. 21), the author further fleshes out his settlement framework by suggestively describing disciplines as "amoebas putting out pseudopods as they move in a multidimensional intellectual space" (p. 138).

Casting a wider net reveals a variety of science studies scholars whose views are largely synergistic with the work of Abbott. Contextualist and constructivist accounts of disciplinary history are particularly relevant here (Messer-Dabidow et al., 1993; Lenoir, 1997). In fact, one central insight to take away from this body of texts centers on the claim that scientific disciplines are ever changing and adaptive. In his study of the field of geophysics, for example, historian Gregory Good nicely summarizes that "scientific activities may achieve degrees of identity development," and that disciplines "pass through no regular stages on their way from immature to mature status" (2000, p. 259). These arguments stand as a further corrective to the more traditional and idealized view of disciplines as uniform and monolithic.

Concerns by these and other authors regarding the negotiation of professional and/or disciplinary boundaries also lead us to concepts such as "boundary objects" and "boundary work." Regarding the former, Star and Griesemer have convincingly described how actors and groups with different interests often use common points of reference (or "boundary objects") to communicate with one another, even if their understanding or interpretation of these entities differs considerably (1989). Their adaptation of Hughes' "institutional ecology" framework to describe the larger institutional backdrop against which these interactions occur can also be usefully adapted to grapple with disciplines and professions. In fact, the preceding overview hints at the extent to which the ecological approach employed by Star and Griesemer resonates with Abbott's work. Akera (2004a; 2004b; 2006), on the other hand, has usefully looked at how ecologies of both knowledge and institutions inflected some early developments of computing field, albeit largely in relation to the career and work of computer pioneer John W. Mauchly.

Gieryn's theorizing on the concept of "boundary work" is similarly helpful here, especially given his insightful discussions about how various demarcation processes are used to bound off domains of disciplinary knowledge (1983; 1995; 1999). More specifically, Gieryn describes four central categories of boundary-work, namely monopolization, expansion, expulsion, and protection (1995, pp. 424-439). Another important theme evident in Gieryn's writings – and also brought to the fore in the work of Golinski (1998, Ch. 2) – centers on the idea that the formation, legitimation, and ongoing development of fields and disciplines often involves the active and power-laden "disciplining" of knowledge, people and even the physical world.

It is further worth noting that much of the literature on the topic of disciplines has tended to focus on the sciences, yet my own work supports the argument that engineering and other technology-oriented fields are equally significant sites for applying the aforementioned concepts. In making this move, it is important to wrestle with questions about the role of knowledge claims in the formation and development of disciplines and professions. In Abbott's analysis of the social sciences, for example, we find that the author's settlement framework is primarily used to uncover and examine competing knowledge claims. While this approach may work well for his particular case, elsewhere Abbott discusses how various non-epistemological factors – such as technological developments and organizational changes – can shift professional jurisdictions. As more specific examples, he notes that the "increasingly technical quality of machinery and physical structures" helped stimulate the establishment and growth of the engineering profession (1988, p. 92), while the later development of "higher-level" programming languages such as FORTRAN and COBOL helped create new areas of expertise that were claimed by computer programmers (1988, p. 93).

In order bring into further relief the full range of factors that impinge on the development of disciplines and professions, I turn to the concept of "co-production," as explored at length in a recent edited volume (Jasanoff, 2004). Central to the co-production framework is the idea that neither social nor natural order can be assumed to have explanatory or causal priority in studies of science and technology, and that we must instead view the social and the natural as actively "co-produced." As editor Sheila Jasanoff describes, scientific knowledge, as well as technology and technological knowledge, "both embeds and is embedded in social practices, identities, norms, conventions, discourses, and institutions – in short, in all the building blocks of what we

term the social" (p. 3). Jasanoff adds that the co-production framework emphasizes the constant intertwining of the cognitive, material, social, and normative, leading us to ask questions such as: "what sorts of scientific entities or technological arrangements can usefully be regarded as being co-produced with which elements of social order; ... what are the principal pathways by which such co-production occurs[?]" (p. 6; p. 18).

My analysis of disciplinary and professional development also looks beyond sociotechnical factors to engage with discourses. My work here is significantly inspired by Paul Edwards' *The Closed World* (1996), which works at the intersection of computing machinery and metaphors of computing in the Cold War era. More specifically, central to Edwards' analysis is the idea that discourse "is a self-elaborating 'heterogeneous ensemble' that combines techniques and technologies, metaphors, language, practices, and fragments of other discourses around a support or supports" (p. 40). Other important points to take away from Edwards include his emphasis on the social processes that are at the heart of discourses, as well as his claim that computer technology acted as a crucial "support" for Cold War, closed-world discourses. The value of discourse as a theoretical framework is also evident in recent work by other scholars. Ronald Kline, for example, carefully traces the long historical development of the phrase "information technology," with particular emphasis on how various discourse communities promoted their own particular interpretations of what this "keyword" signified (2006).

Building on the work of these authors, my own historical study of the field of computer engineering frames disciplines and professions as "heterogeneous ensembles" that are composed of diverse sociotechnical elements, ranging from discourses, identity markers, and institutional structures to technologies and bodies of knowledge.[5] Hence, I depart from Edwards by seeing discourse as one among many important facets of disciplinary and professional development, rather than the fundamental plane on which the field of computer engineering has historically been constructed. Further building on the preceding theoretical moorings, I claim that establishing and legitimating a discipline or profession requires that its proponents bring many heterogeneous elements into alignment in order to achieve some level of stability. Yet this stability is necessarily both partial and temporary, in no small part because these fields and

[5] As authors such as David Hess noted, Foucault is generally credited with initially developing the concept of a "heterogeneous ensemble" (1997, p. 107). It has been widely applied in the field of science and technology studies, in no small part due to the ease with which it can be applied to a wide variety of sociotechnical subjects.

subfields always exist against the backdrop of pre-existing systems of professions and/or ecologies of disciplines. Drawing on the coproduction framework, on the other hand, helps remind us that the realization of computer engineering as a distinct field of activity involves not only discursive achievements, but also successful alignments of social and technical order, often in the midst of rapid sociotechnical change.

I also explicitly and intentionally deal with issues of discipline formation *and* professional development in my account, thereby helping to bridge some of the analytic and topical divides reflected in the preceding literature review. More specifically, I document the role of professional societies and educational institutions as key interfaces or mediators between the professional and disciplinary realms. In fact, professional societies are particularly important in this regard, given that they often serve as a key common point of contact for members of industry and the academy. Further, professional society publications and activities frequently provide unique high-level perspectives on the state of a given field. Finally, and as suggested by its title, one of the key themes of this dissertation centers on the somewhat ambiguous position of computer engineering between profession and discipline, which I claim has been a major yet oft-overlooked source of instability in the more than five-decade-long history of the field.

Methodology

As a guiding principle, the research on which this project is based is largely focused on those persons and texts most closely associated with the field of computer engineering. Hence, my goal is to develop a history of the field that emphasizes the perspectives of computer engineers themselves. While this approach keeps the project somewhat more focused and manageable, I avoid a monolithic account by grappling with the multiplicity of viewpoints and agendas that have existed within the field. The project is also concerned with individuals, technologies, and texts from adjacent fields, but with an emphasis on how engineers have reacted to, commented on, and/or interacted with these "outsiders."

It is also important to emphasize that this account is almost exclusively concerned with developments in the United States. In terms of benefits, this approach makes the scope of this project far more manageable, especially with respect to placing reasonable bounds on both the research required and the length of the resulting analysis. On the other hand, I am acutely aware that many of the themes developed in this dissertation are partially or even wholly peculiar to the

American context. In some nations and regions, for example, the field of computer engineering does not exist *per se*, while other disciplinary designators such as informatics or software engineering may take on different meanings and/or have much greater prominence, especially as compared to the United States. In the future, I intend to look more closely at the unique development of computer engineering and related fields in a variety of national and cultural contexts, with the present case standing as one example among many.

The data presented in the following chapters has been culled from a wide variety of primary sources, including professional journals, trade magazines, conference proceedings, committee reports, and textbooks. I also draw on secondary historical accounts, as well as a handful of oral histories. My approach is largely qualitative in nature as I examine historical patterns, explore the positions and interests of different actors and groups, and seek out the deeper values, ideologies, and meanings of an array of discourses and texts. Some quantitative data is presented to account for broad-based trends, including the historical development of academic departments and degree programs in computer science, computer engineering, and related fields. Below I discuss in more depth the significance of the major types of source material on which this project is based.

Professional Publications

The various publications of professional societies – such as transactions, conference proceedings, journals, and magazines – are a main source of data for this project. In narrowing down the scope of this material, I concentrate on the publications of the most relevant professional groups, including the AIEE, IRE, IEEE, and ACM. At times my analysis also narrows to relevant sub-groups, such as the IRE-PGEC and the IEEE's Computer Society. In addition to allowing us to glean the larger orientations, agendas, and institutional histories of these groups, many professional publications act as sounding boards for high-profile actors in the field. They also sometimes serve as outlets for debate over contentious issues, especially via letters to the editor, published speeches, and/or special messages from the leaders of organizations. I also use a number of personnel ads published in AIEE and IRE journals to document some early employment trends in the field. To varying extents, these publications have historically served mixed audiences of professionals with ties to both the academy and industry. As this overview suggests, professional society publications are valuable not only because they

tell us much about the parent organizations, but also because they frequently provide valuable windows into contexts that are otherwise difficult to access or assess, such as the private sector.

Trade Magazines

The early chapters of my analysis present significant evidence drawn from two of the largest computer-oriented trade magazines from the 1950s and 1960s, namely *Computers and Automation* and *Datamation*. And while these outlets tended to lack the prestige and technical rigor of other types of publications, they often featured articles and commentary that were more candid, daring, and accessible. *Datamation*, for example, became well-known for carrying the witty yet biting editorial remarks of Herb Grosch, the so-called "enfant terrible of the computing world" (Shapiro, 1994, para. 14). These magazines also maintained closer ties to the computing industry, frequently discussed larger trends in the field, and periodically published critical evaluations of the major professional societies and computer conferences. Looking at the articles and advertisements published in these magazines provides valuable opportunities for comparison and contrast, especially through juxtapositions with professional society publications.

Curriculum and Model Program Reports

The latter chapters of this dissertation place significant emphasis on a long series of curricular recommendations and model program reports. Many of these were authored and published by subcommittees of the ACM and IEEE, while others were developed by quasi-independent groups like the COSINE Committee. In order to further enrich my analysis, I seek out and analyze related documents such as interim reports, summary articles, reviews, and follow-up commentaries. These types of sources are important in that they often explicitly describe what counts – or what an author or group thinks should count – as computer engineering and/or computer science. These often involve idealized depictions of a given field's history and agenda, supported by in-depth evaluations and surveys of the structure and content of academic departments, programs, and courses. In addition, the authors frequently articulate an imagined future for various computer-oriented disciplines by presenting recommended reforms and detailed curricular recommendations and model programs. At the same time, these documents often reveal some of the ways in which the field's academic and professional spheres are linked. Perhaps not surprisingly, many of these reports have triggered extended discussion and heated

debate over the identity, agenda, and scope of computer engineering, computer science, and related fields.

Other Sources

I use a large number of sources that do not fall neatly within the categories describe above. Material in this category includes primary sources such as textbooks, reference volumes, and other types of committee reports. I also make extensive use of secondary sources, including other historical accounts, retrospective histories by primary actors, and oral history interviews conducted by both others and myself. I often draw on these sources to compare and contrast how various actors, groups, and sectors have dealt with common themes and issues. As such, these materials help me to both "triangulate' my analysis and develop more compelling arguments.

Summary of Chapter Contents

The historical account that follows features six body chapters and a concluding epilogue. The body chapters can be further divided in two major parts, with Chapters 2 through 4 focused on the pre-history and early history of "computer engineering." In these chapters I place particular emphasis on documenting how the field of computer engineering gained a partially distinct *professional* identity, largely in the context of industry and through the activities of professional societies.

Turning to individual chapters, the major goal of Chapter 2 is to account for the historical emergence of "computer engineer," "computer engineering," and closely related terms. Doing so, however, demands a summary review of the history of electrical engineering, with a focus on both the early decades of the twentieth century and the development of the American Institute of Electrical Engineers (AIEE) and the Institute of Radio Engineers (IRE). I then turn to the 1940s, when a handful of influential electronic computer projects got off the ground, and when a nascent computing community first started to emerge. My account looks closely at engineers, both by examining the roles they played in designing and building the first high-speed, digital computers, and by documenting the early movement of groups such as the AIEE and IRE into various areas of computing.

In the latter parts of this chapter I turn to some of the distinct identities, activities, and bodies of knowledge that were growing up at the intersection of engineering and computing,

especially in the early and mid-1950s. And by focusing on some of the earliest uses of terms such as “computer engineer” and “computer designer” in the context of industry and by professional societies, I analyze how the boundaries around this emergent branch of the engineering profession were defined and negotiated in relation to both the development of computer technology and some of the other major subfields of computing. My account also gives voice to a handful of commentators who were beginning to critique these boundaries, especially in light of the apparent, ongoing expansion of the divide between computer designers and computer programmers.

In Chapter 3 I turn to the internal development and relational interaction of three major groups that maintained interests in the computer field through the 1950s and into the 1960s, namely the IRE’s Professional Group on Electronic Computers (IRE-PGEC), the AIEE’s Computing Devices Committee (AIEE CDC), and the Association for Computing Machinery (ACM). My account frames these organizations as constituting a dynamic “system of professional societies” that was united not by a shared association with a single profession or discipline, but rather by their overlapping and interpenetrating settlements in various bodies of knowledge and domains of technology. Further, I argue that the overall stability of this system can be accounted for by looking at a long series of negotiations and compromises that were worked out within and between these groups.

I place particular emphasis in this chapter on the long-running Joint Computer Conference (JCC) series and associated National Joint Computer Committee (NJCC). In fact, I claim that the JCC and NJCC played key roles in 1950s as common points of contact where these groups could negotiate their respective sociotechnical settlements. The success of this process is all the more striking given the presence of various destabilizing forces, including rapid technological developments, changes in the size and scope of each group, and incursions from outsiders. My account also speaks to how the stability of this system was maintained through a series of major institutional changes in the early and mid-1960s, including the merger of the AIEE and IRE to form the Institute of Electrical and Electronic Engineers (IEEE) and the founding of the American Federation of Information Processing Societies (AFIPS). And finally, throughout this chapter I document the close association of the IRE-PGEC and its successor organization (the IEEE Computer Group) with “computer engineering” and “hardware.”

While my third and fourth chapters look at roughly the same historical period, the latter is focused on three additional sites where the dominant images of computer engineering developed, namely in industry, hardware-software discourses, and the academy. To begin with, I examine how the term "computer engineer" and its variants went through a period of interpretive flexibility in the mid-1950s, finally stabilizing in the 1960s to refer to design-oriented work in computer circuits, logic, and systems. In so doing, I show how computer engineers and designers were associated with the domain of computer "hardware," linked to engineering education and the engineering profession, and positioned in relation to other types of computer professionals. These themes provide an appropriate segue to a more general discussion of the computer field's evolving sociotechnical boundaries. More specifically, I juxtapose the fragmentary tendencies of the software-hardware dichotomy with a variety of calls for "integrating" the computer field's major divisions of labor, technologies, and bodies of knowledge.

The second major part of this chapter reviews some early trends in the formal education of computer professionals, especially through the 1950s and into the early 1960s. More specifically, I document the slow development of computer-oriented courses and degree programs in electrical engineering departments, as well as some of the earliest efforts to promote the "computer sciences" as a formative academic discipline. My account also raises questions about the extent to which the academic context was positioned to serve as a site for either challenging or reinforcing the computer field's major sociotechnical boundaries.

Chapters 5 through 7 cover a historical period running from roughly the mid 1960s to late 1980s and early 1990s. Topically, these chapters are primarily focused on both the establishment of a distinct *disciplinary* identity and negotiation of a partially unique "sociotechnical settlement" for the field of computer engineering, especially through developments in the academic context. This part of the dissertation also engages with two other major themes. First, I discuss the evolving character and role of the relevant professional societies. Second, I document how computer engineering came to occupy an unstable position between the engineering profession, on the one hand, and independent disciplines such as computer science, on the other.

Chapter 5, in particular, carries my analysis of the educational sphere through the remainder of the 1960s and into the early 1970s. The first major part of this chapter documents ongoing efforts to define, position, and institutionalize the discipline of computer science. In addition to emphasizing the role of the ACM and its constituency in this process, my account

also points to the importance of "bottom-up" processes of disciplinary development, where the establishment of computer science courses, programs, and even departments greatly bolstered the legitimacy and independence of this young field. In the middle part of this chapter I turn to a handful of "insiders" who raised concerns in the mid- and late-1960s about computer science education, including its continued movement away from engineering and technology.

This line of analysis helps sets up the second major part of this chapter, which details how a new cadre of electrical engineers lobbied for a thorough reorientation of electrical engineering education toward computers and computing. By focusing on the activities of the COSINE Committee and its members, I document how the initial efforts of these reformers to bring computer science "into the fold" of electrical engineering were largely replaced by calls for the establishment of computer engineering degree options and programs within existing engineering programs. While these developments laid important foundations for the ongoing development of computer engineering education, they also suggested that the major sociotechnical schisms of the computer field were being reproduced in the academic sector. On a related note, my analysis also reveals growing tensions between the dominant images of computer science as an independent *discipline* and computer engineering as a branch of the engineering *profession*.

My sixth chapter is largely focused on the history of the IEEE Computer Group – later renamed the Computer Society – from approximately the mid-1960s to late-1980s. To begin with, I demonstrate how this group's position between the IEEE as its parent organization and the ACM as its independent sibling society was maintained during this period. More specifically, I show how various structures and processes of "sociotechnical mediation" helped create a modicum of stability in this system of societies, especially against the backdrop of rapid technological and institutional change, and irrespective of extensive overlap and penetration between the sociotechnical settlements of each group. In fact, I argue in this chapter that the relationship of the IEEE Computer Society and the ACM through the 1970s and into the 1980s bore a striking resemblance to the evolving relation of hardware and software. By more closely examining the interests and activities of these two groups in a handful of "boundary" domains – such as computer architecture, microprogramming, and software engineering – I argue that these similarities were no coincidence, but rather a potent reflection of the ongoing coproduction of the

computer field's social and technical order. This chapter also reveals a gradual fading of the Computer Society's image as primarily an organization of and for computer engineers.

Chapter 7 starts by examining the Computer Society's expanding activities in the educational arena, beginning in the 1970s. My analysis reveals two countervailing trends. On the one hand, I show how various actors and groups worked to develop curricular and program recommendations that were designed to better integrate or unify computer science and computer engineering education, beginning with the Computer Society's "Computer Science and Engineering" movement from the mid-1970s to mid-1980s. This trend culminated in the late-1980s and early 1990s with the "Computing as a Discipline" and "Computing Curricula 1991" projects, which involved unprecedented levels of cooperation between the Computer Society and the ACM and seemed to point the way toward major curricular reforms.

On the other hand, this chapter reveals the perennial fragmentation of computer science and computer engineering education, as reflected by the publication of alternative and competing curricular recommendations, the prevailing structure of degree programs and departments, and the establishment of multiple accreditation systems and processes. In addition to emphasizing the destabilizing character of these forces, my analysis speaks to how these trends are linked to other persistent schisms in the field, including those based on the poles of software and hardware, science and engineering, and profession and discipline.

Finally, I use a concluding Epilogue to accomplish two major goals. First, I review some recent developments in the educational arena to highlight continued instabilities in the disciplinary landscape of computing, as well as new calls for the establishment of a distinct disciplinary and professional identity for the field of "computer engineering." Second, I bring into relief important countervailing trends through a brief historical introduction to the software/hardware codesign movement. My analysis also points to some of the larger implications of these trends, especially as related to the future training of computer professionals and the future shape of computer technology. As my account makes clear, debates about the sociotechnical boundaries of computer science and computer engineering are not only deeply rooted in history, they are also alive and well today, and their outcome will likely have far-reaching implications.

Chapter 2

From Engineers and Computing to Computer Engineering

Among the many important developments that often appear on timelines and chronologies of computer history, the first Joint Computer Conference (JCC) is an oft-cited event. And indeed, the conference is worthy of recognition. Held in Philadelphia in 1951, the meeting attracted almost 900 attendees, making it one of the largest – if not *the* largest – computer conferences to date. Yet most historical accounts fail to discuss the distinct scope and tenor of the event. Unlike other early events of this type – which tended to attract a wide variety of attendees and cover a broad swath of topics – the inaugural JCC was primarily focused on the "engineering aspects" of computer design and construction. It was also largely organized under the auspices of two professional societies, namely the American Institute of Electrical Engineers (AIEE) and the Institute of Radio Engineers (IRE), and electrical engineers dominated both the conference planning committee and the roster of conference speakers.

In light of this overview, how might we account for the fact that this conference was both largely focused on the physical technology of computing and primarily organized by and for electrical engineers? Further, does the unique character of this event have some larger historical significance? As I argue in this chapter, the orientation of the first JCC was not an anomaly, but rather an important element in a more general movement. By the late 1940s and early 1950s, a growing band of electrical engineers had recognized the rapid expansion and increasing importance of high-speed, electronic computers, and they started to actively stake out their territory in this nascent yet burgeoning domain of activity. And as this movement gained momentum through the early and middle part of the 1950s, terms such as "computer engineer" and "computer designer" emerged to provide a distinct social and professional identity for engineers who were working in the computer field. Further, these new identities emerged in tandem with – and became linked to – the general sphere of computer technology designated by the term "hardware."

Yet before I develop a more detailed account of these trends, it is necessary to provide the relevant background, especially with regard to both the history of electrical engineering and early development of high-speed computing. The first section of this chapter reviews the history of electrical engineers and electrical engineering knowledge, with an emphasis on the founding and development of the AIEE and IRE. I then turn to the 1940s as a key decade for high-speed electronic computing, when a handful of major computer development projects were getting underway, and when tentative efforts to cultivate a more cohesive computing field gained momentum. In order to further appraise the role of electrical engineers and electrical engineering in this historical account, this chapter also provides an extended discussion of AIEE and IRE efforts to enter various areas of computing, from roughly the mid-1940s to mid-1950s.

The latter sections of the chapter turn more specifically to the distinct identities, activities, and bodies of knowledge that were emerging at the intersection of engineering and computing. Using the first joint conference as a window into this theme, I review a number of early comments about both the historical and prospective links between electrical engineering and computing. In doing so, I uncover some of the earliest uses of labels such as "computer designer" and "computer engineer." As I demonstrate, exploring the history of these terms brings into further relief early efforts to establish a distinct professional identity for the many electrical engineers who were designing, building, or otherwise working with the first generation of high-speed electronic computing machines. Further, tracing out the history of the term "computer engineering" reveals some of the ways in which the boundaries around this area of activity were being defined and negotiated in relation to both the ongoing development of computer technologies and the other major subfields of computing. Yet as suggested in the account that follows, the long-term relationship between engineers, on the one hand, and computers and computing, on the other, was neither obvious nor fixed.[6]

A Brief Early History of Electrical Engineering and Its Institutes

For many decades, the AIEE and IRE were the two major professional societies for American electrical engineers, and tracing out their respective histories forcefully reveals how the development of engineering fields and subfields frequently involves the negotiation of social,

[6] In historical terms, the noun "computers" has frequently been used to refer to the machines themselves, while the verb "computing" more often refers to the use or application of such machines.

technological, and epistemological boundaries. A review of this history also provides important background framing for the remainder of this chapter. Established in 1884, the AIEE was formed as the first national professional society for American electrical engineers. In summary, the founding of the group was largely stimulated by the rapid development and maturation of electrical science and technology, along with a growing recognition that the relatively young field of electrical engineering was distinct from its historical forerunners, such as civil and mechanical engineering. As explained by historian A. Michael McMahon, the new field "was veering off sharply from America's traditional engineering culture" (1984, p. 1).

With its leadership and membership ranks initially filled out with a "broad spectrum of electricians and capitalists" (McMahon, 1984, p. 29), the founding and early expansion of the AIEE paralleled the growth and maturation of the electric power industry, and to a lesser extent the telegraphy industry. Yet the Association was also developing in concert with the increasing professionalization and academization of electrical engineering work and education. By the late decades of the nineteenth century, the formal training of electrical engineers was gaining momentum in colleges and technical schools, initially in physics departments, but increasingly in standalone engineering departments. The composition of the AIEE – as well as the field of electrical engineering more generally – was increasingly populated by rank-and-file engineers, scientist-engineers, and engineering managers, many who happened to hold engineering degrees (Layton, 1971, p. 39).

By the early 1910s, the AIEE's primary orientation toward both power engineering and the commercial sphere was well established. Further, it was clear by this time that the electrical engineering field was entering a phase of rapid diversification, in no small part due to ongoing scientific breakthroughs and technological developments. For instance, the discovery of the electron in the latter part of the nineteenth century opened up a broad swath of research into related electrical phenomena, devices, and applications. The communications field in particular benefited greatly from this research, especially as the invention and refinement of vacuum tubes – one of the most important of the early "electron devices" – helped pave the way for the development of radio broadcasting and other new types of wireless communication. In a move that was clearly intended to bring these and other new areas of research under the AIEE's umbrella, the institute formed a series of new technical committees from around 1910 onward (McMahon, 1984, pp. 126-127).

Yet tentative steps to establish a committee in the increasingly important area of radio – or "wireless telegraphy," as it was called at the time – came too late. Responding to the AIEE's lack of coverage in low-voltage and wireless technologies, two upstart radio engineering societies merged in 1912 to form the IRE. And while this new group was in many ways indebted to the AIEE as its historical forerunner and organizational prototype, the IRE also represented a new and increasingly divergent subculture of electrical engineering. As described by McMahon, radio engineering, "though close intellectually and institutionally to power engineering, possessed a distinctive social and technical basis" (McMahon, 1984, p. 131).[7]

The distinct character of the new group was linked to a number of factors. For starters, and in contrast to the make-up of the AIEE, radio engineers tended to be younger and often worked for smaller companies (Abbott, 1988, p. 180). In addition, many of the concerns faced by the IRE and its members centered on the sorts of issues that typically challenge new fields and subfields. For example, standardization of units and measures was one such issue that received significant coverage. Another important topic centered on new technological developments, especially in the area of electron devices. And finally, Layton has argued that the IRE's early orientation toward science and "scientific professionalism" further distinguished it from the AIEE (Layton, 1971, p. 43; 251).

Yet as the field matured in subsequent decades, the IRE remained largely oriented toward radio industry and technology, a trend that mirrored the persistent links between the AIEE and the power industry. This tilting of the IRE toward radio was increasingly problematic, especially given the growing prominence and expansion of electronics as a more general field of interest through the 1920s and the 1930s.[8] While electronics had largely grown out of radio engineering and technology, it quickly diversified and grew to the point where radio was overshadowed by both the rapid expansion of electronics research and a proliferation of new applications. And in a

[7] While this comment is ostensibly about the radio engineering, it nicely captures the more general processes of division and fragmentation that have led to the periodic creation of new engineering fields and subfields. Yet it is often difficult to sort out the primary reasons for such divisions. Regarding the IRE, Layton has argued that the motivations for founding the group were largely professional rather than technical. More specifically, he explains that the formation of the IRE was significantly a reaction to AIEE moves around 1912 to relax membership requirements – and hence make the organization more friendly to business (Layton, 1971, p. 43). Per Layton, business interests often tend to promote division in the engineering field, while professionalism often encourages greater unity (p. 44).

[8] Further, and as noted in an early historical retrospective authored by well-known Cambridge computer pioneer Maurice V. Wilkes, the rising prominence of electronics was in part reflected by the establishment of the monthly magazine *Electronics* in 1930 (Wilkes, 2004, p. 1).

somewhat ironic reversal of roles, it was the AIEE's Communications Committee that moved to fill this gap in the 1930s, largely by expanding conference activities and publications in the area of electronics (McMahon, 1984, pp. 188-189). On the one hand, the AIEE succeeded in serving some of the needs in this budding field, while also encouraging joint AIEE-IRE activities in areas of common interest. On the other hand, McMahon has argued that the AIEE nonetheless "remained a predominantly power engineering society" (1984, p. 194), both during this period and beyond.

The Second World War further shaped the social and technological boundaries of the electrical engineering field. For starters, wartime research helped set the stage for the continued ascendance of electronics engineering and technology. Per McMahon, "the wartime R&D program powerfully launched electronics as the nation's dominant technology in the postwar era" (1984, p. 195). Even more importantly for the present analysis, an even larger number of developments from this period were pivotal for many subsequent technological developments, including high-speed electronic computers. Research in the area of radar, for instance, led to innovative new devices like mercury delay lines, as well as to dramatic improvements in existing devices, such as vacuum tubes. Further, considerable experience was gained during the war in the design and construction of electronic systems of unprecedented scale and complexity. A number of major research projects, for instance, advanced the state of the art in the area of analog computing devices, especially as applied to problems such as automatic gun control.

A wide range of difficult computational problems encountered during the war also helped sow the seeds for the emergence of large-scale digital computers. The calculation of ballistics tables was a particularly intense area of activity that provided much of the justification for the design and building of war-era computers such as the ENIAC. These early projects also hinted at the major potential areas of involvement for electrical engineers in computing. More specifically, these areas included components (or devices), systems, and applications. The distinction between analog and digital computers surfaced as another important boundary that cut deeply through all phases of computer design and use.

The war also brought into relief the major social and technical divisions of labor in the area of electronics. Most important for the present analysis, it was increasingly evident that electrical engineers wielded no clear monopoly over this expanding domain. To be sure, many electrical engineers played important roles in war-oriented electronics research, but many of their

best-known contributions were in the administration of research programs and shaping of R&D policy (McMahon, 1984, Ch. 6; pp. 195-206). In terms of actual research and development work, engineers were often overshadowed by physicists generally, and "electronic physicists" specifically.

While McMahon sketches the outlines of this story, historian Peter Galison provides a detailed description of wartime research activities that nicely highlights the typically lop-sided relation of physicists and engineers (McMahon, 1984, 233-245; Galison, 1997, Ch. 4). Abbott similarly notes that physicists were often at the forefront of innovation in the area of electrical engineering and electronics, while engineers were more frequently involved with specific applications and routine types of work (Abbott, 1988, pp. 181-182). As these accounts make clear, science in general – and physics in particular – maintained an upper hand over engineering in terms of prestige and status, both during and after the war. In fact, a prominent speaker at the 1946 National Electronics Conference revealed the extent to which engineering was viewed as "downstream" from science by titling his talk "Physics of Today Becomes the Engineering of Tomorrow" ("1946 National Electronics Conference," 1946, p. 665). As I discuss below, the early development of the computer field was similarly marked by the presence of both physicists and engineers, at times working in cooperation, and at times standing in tension.

Finally, it is worth noting that from the 1940s onward the IRE can be credited for reversing its tendency toward specialization, allowing it to secure a position at the forefront of the electronics field. As McMahon notes, "the electronics engineers' choice of the IRE over the AIEE for their professional society made all the difference for the futures of the two societies" (1984, p. 214). And as subsequent passages make clear, this was one of a number of factors that tended to inhibit AIEE's involvement in computing. On the other hand, the AIEE was one of the earliest professional organizations to express formal and active interests in computing. Exploring this interest, however, first requires a review of some early developments in computing.

Intersections of Expertise in Early Computer Development Projects

As noted above, wartime research activities provided significant impetus for the design and construction of high-speed computing machines. Other key developments can be traced back to earlier decades and even earlier centuries. Since much of this history has been covered in detail elsewhere, I will largely restrict my attention to surveying the social composition and

technical character of some of the early computer development projects, labs, and conferences, with particular emphasis on the roles played by electrical engineers and electrical engineering knowledge. This section is also focused on the 1940s, when the first high-speed electronic computing machines were being built and becoming operational, and when a small but growing cadre of organizations and individuals were coalescing around common areas of interest.

Regarding the overall scope and scale of computing during this time period, Aspray estimates that by the mid-1940s there were at most ten major computer research centers, ten operating high-speed computers, and 1,000 persons interested in computer development (Aspray, 1985, p. ix). In rather general terms, each of the early computer projects involved the assembly of a diverse range competencies, knowledge, and resources, all of which helped facilitate the successful design, construction, and use of the first high-speed computing and calculating machines. Cortada, commenting on the skills required to build these machines, explains that "[t]he electrician had to work with the engineer and the mathematician with the physicist to make it happen" (Cortada, 1993, p. 14).

More specifically, electricians and low-level (or up-and-coming) engineers often acted as the "higher technical labor" that was essential for physically building and testing these large and very complex calculating machines. Numerous electrical engineers and physicists, on the other hand, contributed a wealth of codified and tacit knowledge, especially in the overall design of electronic systems, as well as in research and development activities involving electronic devices, such as vacuum tubes. A large number mathematicians and scientists added important theoretical angles and mathematical foundations, while also frequently grappling with the challenges of actually using high-speed computing machines to solve mathematical problems.

This blending of disciplines and talents was evident in all of the early, large-scale computer projects, albeit with interesting local variations. Yet credit for the design and development of these early machines was rarely distributed evenly, and was frequently skewed toward those individuals with more prestigious backgrounds in science or mathematics. At Bell Labs, for instance, a series of large relay computers was designed and built from the late-1930s to late-1940s. Much of this effort was led by George Stibitz and Samuel Williams, the former with a background in mathematics and physics, the latter with significant training and experience

in engineering generally and electrical engineering specifically.[9] As historian Atsushi Akera has documented, in later years Williams was disappointed by the credit that was heaped on Stibitiz for his role in the design of the Bell computers (Akera, 1998, p. 575).[10]

A variant of this theme played out in the story of the Automatic Sequence Controlled Calculator (ASCC) or Mark I, another early computer that was largely based on electromechanical technology. The design of this high-speed calculating device was primarily formulated by Harvard's Howard Aiken, who held an undergraduate degree in electrical engineering and a Ph.D. in Physics (Aspray, 2000, p. 51).[11] Yet the detailed design work and overall construction of the Mark I was almost entirely in the hands of IBM engineers, who assembled much of the machine from off-the-shelf components such as relays, switches, rotating shafts, and clutches.

Completed in 1943, the Mark I was commissioned and used by the Navy and used to solve numerous war-time computing problems. Yet just as the machine was being put into service, a dispute between Harvard and IBM surfaced when Aiken failed to acknowledge IBM's role in the building of the computer. As described by Aspray, "The dispute may have stemmed in part from the different ways in which scientists and engineers value contributions: Harvard thought of Aiken's functional specifications for the machine as foremost, while IBM regarded the real work as residing in the engineering design and construction" (2000, p. 51). Far from an isolated incident, the tendency for these kinds of links to form between university research, science, and theory, on the one hand, and industry, engineering, and technology, on the other, was an important factor in the unique trajectory of many subsequent computer development projects, as well as in the more general development of the computer field.

---

[9] Stibitz held an undergraduate degree in Math and Physics from Denison University, an M.S. from Union College, and a Ph.D. in physics from Cornell ("Inventor Profile – George Stibitz," 2002). Williams received an M.E. degree in Electrical Engineering from Ohio State in 1905 ("Retirements," 1946, p. 252).

[10] Later in this same account, Akera goes so far as to describe Williams as "a mechanic and not a mathematician" (Akera, 1998, p. 578). In light of both Williams' education in electrical engineering and his working experiences as an engineer at Bell Labs, "engineer" is a clearly a more appropriate and accurate label for Williams.

[11] Aiken's dissertation work, which he completed in 1939, was doubly relevant for his later work in computing. First, his research was focused on theoretical problems associated with vacuum tubes. Second, these problems involved non-linear differential equations, which were very difficult to solve via conventional means. As early as 1936, Aiken was considering how calculating equipment might assist in solving such equations (Aspray, 2000, p. 51).

Surveying some of the major computer projects that got underway in the mid and late 1940s provides additional clues about the disciplinary composition of the formative computer field. In terms of the technical foundations of large-scale computing, a shift from electromechanical components (such as relays) to electronic components (primarily vacuum tubes) was well underway during this time period. Further, much research in the design and application of computing machines was situated in university computer labs, often through extensive government support. In fact, universities were especially well suited to such projects given the relative ease with which diverse talents, abilities, and resources could be assembled and coordinated in academic research environments. Among the handful of early focal points for computer research, Harvard, Princeton, the University of Pennsylvania, and MIT were some of the key institutions in the 1940s.[12] Surveying the activities at these four schools helps shed additional light on how university research and development activities in computing variously intersected with the major fields and subfields of science, mathematics, and engineering.

After the Mark I project soured Harvard's relationship with IBM, Aiken's Computation Lab undertook the design and construction of three large-scale computers, namely the Mark II, III, and IV. But in contrast to other early sites of computer research and development, the relatively low status of engineering at the school contributed to the evolution of a unique culture of computing at Harvard, one that emphasized mathematics, theoretical science, and applications. As evidence for this tendency, the computer lab had close ties to the Department of Engineering Sciences and Applied Physics, and Aiken's tenure was in applied mathematics. Further, many among the "strong cadre of master's and doctoral students" that passed through the lab were trained from 1947 onward in an applied mathematics program that was more specifically oriented toward the use of computing machinery, rather than its design (Aspray, 2000, p. 52).

Yet historians also suggest that Aiken held a favorable view of engineering education as a pathway into computing, even if it was not a readily available option for Harvard students. As recounted by Kathleen Broom Williams, in 1945 Aiken told a visitor from the Aberdeen Proving

[12] Aspray provides an excellent review of the entry and ongoing involvement of these four schools in computing, with particular emphasis on educational activities such as courses and degree programs (2000). And while he also discusses key developments at Columbia University, this school did not spearhead an early computer design and development project. Instead, it maintained close ties with IBM, which furnished the school with computers and other calculating equipment, and utilized Columbia researchers in developing new and improved machines.

Grounds that he should seek engineers rather than mathematicians in staffing the Ballistics Research Laboratory (1999). Indeed, Aiken may have gained this insight by observing the researchers with whom he worked. As described by Williams, well-known mathematicians at the Harvard Computation Lab such as Richard Bloch and Grace Murray Hopper worked very hard to gain an in-depth understanding of the engineering and technical aspects of the Mark I.

As suggested by this overview, Harvard researchers frequently grappled with aspects of computing that were likely labeled engineering elsewhere. But at Harvard, the term was generally avoided. Further, Aiken gained an early reputation for his interest and expertise in computer use, and both Aiken and Hopper published a number of important early articles focused on programming and applications. In addition, historians such as Aspray have documented Aiken's notoriously "conservative approach to machine design" (2000, p. 54). In more general terms, the movement of the Harvard Lab away from the engineering and design dimensions of computing was established relatively early, with Aiken announcing in 1949 that only one more computer (the Mark IV) would be built at the school (Aiken, 1951).[13] Aiken's moves hinted at larger trends that were emerging at the time, as the primary locus of computer design and development activities shifted from universities to industry, a point to which I return.

In many ways, the early history of computing at Princeton paralleled developments at Harvard. For starters, computing at the school was largely situated in the Institute of Advanced Studies (IAS), which was primarily composed of distinguished mathematicians and scientists. One member of the lab, John von Neumann, developed an interest in applied mathematics and related topics through his war-related research activities. Recognizing the value of high-speed computing devices in his own work, von Neumann visited the Moore School and gained familiarity with the ENIAC and EDVAC projects (Ceruzzi, 2003, pp. 21-23; Aspray, 2000, p. 72). By 1945 he was actively pursuing a computer project at the IAS, yet he faced many obstacles (Aspray, 2000, p. 72). In addition to the challenges that came with finding funding for such an ambitious project, von Neumann had to work around the dominant culture and orientation of the IAS, which placed utmost value on theory, science, and mathematics. As Aspray explains, "Most of the faculty regarded computing as a practical subject area, not worth

[13] Soon after the Mark IV was completed in 1952, Aiken explicitly "ceded hardware development to industry" (Aspray, 2000, p. 54), a point to which I will return.

of their investigation," adding that "there was great concern among the faculty over having to share their hallowed grounds with engineers, technicians, coders, and operators" (2000, p. 73).

The resourceful von Neumann ultimately attracted the necessary support and backing for his venture, in part by soliciting additional design and engineering support from the Radio Corporation of America (RCA), as well as from other Princeton departments and labs (Aspray, 2000, p. 73). And even though this particular machine didn't go into operation until 1952, the technical knowledge gained during the project quickly spread beyond Princeton, and the IAS computer was even used as a prototype in building machines at other sites, such as the RAND Corporation (Aspray, 2000, p. 73). Yet formal education in computing at Princeton remained sparse, and the relationship between the IAS computer project and Princeton's Electrical Engineering Department was minimal, at best. Willis Ware, who worked on the IAS computer project while pursuing a Ph.D. in electrical engineering, more recently noted the difficulties he faced as he tried to find electrical engineering faculty at Princeton who were qualified to review his computer-oriented thesis (2005). In more general terms, computing at Princeton went into a short period of decline in the mid-1950s. But as documented by Aspray (2000), this trend reversed in the 1950s, in large part due to computer-oriented research and educational activities that were led by Princeton's Electrical Engineering Department. I return to this topic below.

The University of Pennsylvania, on the other hand, leaned much more strongly toward the engineering end of the science-engineering spectrum, perhaps not surprising given that the Moore School of Electrical engineering was the university's principal site for computer-related research. One of the earliest and best-known computer projects at the School involved the design and construction of the ENIAC between 1943 and 1945. Supported by Army funding and built for the Ballistics Research Laboratory, the ENIAC's somewhat esoteric design in part reflected the wartime context in which it was developed. In fact, a "freeze" was placed on the design of the machine at a relatively early stage, in hopes that the ENIAC would be up and running as quickly as possible (Winegrad, 1996, p. 8). The successful construction of the computer also brought together a broad array of expertise in areas such as engineering, science, and mathematics, and electrical engineers and electrical engineering knowledge were especially prominent in this particular project. [14]

[14] Akera's history of the ENIAC is especially valuable in revealing the multi- and interdisciplinary character of the project (Akera, 2000, pp. 62-121).

John W. Mauchly and John Prester Eckert shared much of the credit for the design and construction of the ENIAC. Mauchly boasted a Ph.D. in Physics and a long-standing interest in electronics, especially as related to scientific instruments (Stern, 1980; Mauchly, 1984). Eckert, on the other hand, held B.S. and M.S. degrees in Electrical Engineering, both earned at the Moore School (Lee, 1995, p. 271). Other important contributions came from the "senior engineers" assigned to the project. These included T. K. Sharpless, another M.S. graduate of the Moore School, and Arthur Burks, who both held a Ph.D. in Philosophy and possessed extensive expertise in the area of logic. Mathematicians at the Moore School such as Hans Rademacher also assisted with the mathematical dimensions of the ENIAC.[15] Another mathematician, Herman Goldstine, acted as the Army's liaison between the Ballistics Research Laboratory and the Moore School, but his contributions to the ENIAC were mainly managerial and logistical. And finally, it is worth noting that a host of relatively "junior" Moore School engineers and researchers made significant contributions in many areas, including component design and testing.

The ENIAC project clearly boosted the Moore School's reputation as an influential and well-known early location for computer development and research. It also became a key site for the training of computer-oriented electrical engineers, and the historical account below is checkered with Moore School staffers and graduates. Many of these engineers contributed to the early computer-related activities of the AIEE and IRE, especially by serving on committees, publishing articles, and presenting papers at conferences. Yet for various reasons, the Moore School's prominent position in computing started to fade from around 1946 onward.[16]

As a final case, the history of computing at MIT can be placed somewhere between the Harvard and Penn examples. Research in the general area of computing machinery started early

[15] The well-known mathematician John von Neumann also visited the project regularly, although his precise contributions have been the subject of much debate. He later became well known for his leading role in the design and constriction of a high-speed electronic computer at Princeton's Institute for Advanced Studies (IAS). This machine was fully operational in 1952. von Neumann was also the sole author of a 1945 report titled *First Draft Report on the EDVAC*, which introduced the basic concepts of "stored-program" computing. While questions remain regarding the role of other computer pioneers in the genesis of the stored-program idea, the term "von Neumann architecture" is often used in reference to this type of computer design, which remains dominant today.

[16] Aspray identifies three major factors that contributed to the declining prominence of the Moore School, namely post-war civilian redeployment of key staffers, the reluctance of university administration to support peacetime military research, and the inability of the school to support the commercial interests of its staff (2000, pp. 59-60).

at MIT, especially through the building of differential analyzers by electrical engineers such as Vannevar Bush and Samuel Caldwell (Aspray, 2000, pp. 43-46). MIT was also an important site for the development of network analyzers, another early type of analog computing device. Project Whirlwind, on the other hand, quickly became one of MIT's most important and well-known computer projects. From the mid-1940s onward the Whirlwind effort was headed by Jay Forrester, who held B.S. and M.S. degrees in Electrical Engineering, the latter earned at MIT.

From the mid-1940s into the early-1950s, the Whirlwind team included top-level staff and graduate students drawn from electrical engineering, mechanical engineering, physics, and mathematics (Wildes and Lindgren, 1985, Ch. 17; Redmond and Smith, 1980). Yet despite this diversity, electrical engineering students made up a very large share of the lab's ranks. Further, electrical engineering at MIT around this time had a particularly high level of prestige, as well as a long-standing reputation for both its roots in physics and its orientation toward science and mathematics (Aspray, 2000, p. 44). By all appearances, these characteristics made MIT engineers especially well-suited to computer-related work, and the school was recognized early on as a key site for computer research and development.

As suggested by these examples, electrical engineers and electrical engineering knowledge played important roles in early computer research. Yet these roles varied significantly from site to site, making it clear that there were no long-term guarantees regarding the position of electrical engineers in the computer field. In fact, it was clear that these engineers often stood in the shadows of other types of experts. As Akera explains, the hierarchies of prestige in postwar computing research were largely a continuation of earlier trends: "the applied mathematicians who aided the physicists in their wartime work first garnered the highest authority with respect to computing research" (1998, p. 336). Following this line of inquiry, Akera adds that mathematicians, engineers, and other types of specialists were each beginning to pursue their own unique approaches to computer research and development, especially in the postwar era. As the field expanded and diversified, the boundaries between these unique approaches and particular areas of interest were increasingly evident.

Connecting the Islands: Early Steps toward a Field of Computing

As Akera has argued, the various computer development projects that were launched both during and soon after World War II were leading toward a "more unified body of knowledge"

and a "more identifiable community" (1998, p. 207). Yet even as these projects started to lay the social and technical foundations for a more recognizable computer "field," the points of contact between isolated researchers and labs remained relatively sparse, even well into the 1940s. Aspray nicely summarizes the situation, explaining:

> There were no professional organizations, regularly scheduled conferences, or journals concerned primarily with high-speed computation. What few papers were published appeared mostly in the proceedings of either the Institute of Radio Engineers (IRE) or the American Institute of Electrical Engineers (AIEE), or in *Mathematical Tables and Other Aids to Computation*. … The main channels of communication between the isolated research centers and individual workers, aside from individual personal contacts and occasional reports, were one-of-a-kind seminars and conferences (1985, pp. ix-x).

Perhaps not surprisingly, many of the aforementioned universities and labs hosted these one-off events, and they often published associated proceedings and reports. This early period was also marked by some of the earliest expressions of interest in the computing field by the electrical engineering institutes.

Early meetings of note include a relatively small gathering at MIT in 1945, which Aspray identifies as one of the first computer conferences (Aspray, 1985, p. x). And another series of six meetings on digital and analog computing machinery was held at Columbia University in 1946 and 1947, organized by the New York chapter of the AIEE (Alt, 1962, p. 300). While these AIEE meetings are rarely referenced in today's historical literature, Alt later claimed that each attracted more than 200 attendees. And finally, Jay Forrester was instrumental in organizing a series of five or six lectures on digital computers at MIT in the Spring of 1947, as part of the Department of Electrical Engineering's seminar program. Each of these lectures attracted 100 or more persons (Alt, 1962, p. 300; Wildes and Lindgren, 1985, p. 287).

As researchers in the field came into more frequent contact with another through these and other events, the social and technological contours of computing became more evident. Take, for example, the "Moore School Lectures." This influential and well-known eight-week lecture course was held during the summer of 1946 at the University of Pennsylvania's Moore School. Appropriately titled "Theory and Techniques for Design of Electronic Digital Computers," the course brought together a diverse assortment of speakers and attendees, most

with backgrounds in engineering, the sciences, and mathematics (Campbell-Kelly and Williams, 1985, pp. xv-xvii). In terms of content, the course included introductory lectures covering a wide range of topics, as well as numerous days dedicated to the close study of specific machines.

Yet the schedule for this event also reflected a growing boundary between the engineering and mathematical dimensions of computing. As explained by one of the event's coordinators, roughly two and a half weeks of the course were planned around "two almost independent programs … one program will treat certain mathematical topics in greater detail, and the other will be concerned with the engineering design features relating to specific components" (Campbell-Kelly and Williams, 1985, p. xxx). As suggested by this passage, the splitting out of these two topic areas revealed that component design and mathematical analysis were being viewed as quite distinct areas of activity. The "system" level of analysis, on the other hand, tended to bridge the diverse phases of computer research and development.

The 1947 Symposium on Large-Scale Digital Calculating Machinery provides another important early snapshot of the field (Harvard Computation Laboratory, 1948). Hosted by Harvard University and jointly sponsored by Harvard and The Navy Department Bureau of Ordnance, the relative size and scope of the event made it one of the most influential of the early computer meetings. For starters, the symposium attracted more than 300 participants, drawn from academic institutions, the government, and private industry. And while incomplete information makes it difficult to compile detailed information about the attendees, members of the symposium identified themselves under a wide variety of occupational and professional designations, with the most common title being engineer, followed by mathematician and then physicist (Aspray, 1985, pp. xvii-xxix).[17] This was clearly a diverse group, in terms of institutional affiliation, professional identity, and agenda.

The early heterogeneity of the computer field was further reflected in the range of topics explored in the eight symposium sessions. A large number of papers and sessions covered machine design and construction, with particular emphasis on descriptions of existing system designs, storage devices, and input-output devices. And as a partial reflection of the research

[17] Of the 335 "members" of the symposium, 86 were clearly identified as engineers or professors of engineering. While many of these individuals were listed as electrical and electronic engineers, many others were simply identified as "engineers," and still others were affiliated with subfields such as aeronautical or civil engineering. Another 49 of the participants were identified as mathematicians or professors thereof, and 35 as physicists or professors thereof.

orientation of the host institution, a large number of sessions and papers coalesced around topics such as numerical methods, computational techniques, and problem preparation and "coding."

Follow-up events followed similar patterns. Take, for instance, the "Symposia on Modern Calculating Machinery and Numerical Methods," which was held in July of 1948 at UCLA and attracted more than 500 attendees ("Symposia on Modern Calculating Machinery," 1949, p. 381). As suggested by the title of the event, the program was organized around two major areas of interest. The first centered on a series of "progress reports" that reviewed recent developments at "principal" sites of computer research and development (p. 382). These reports were split out into two separate sessions, one dedicated to reports from academic research centers, the other to speakers from commercial laboratories.[18] While it is difficult to determine the content of these presentations, both the roster of speakers and the paper titles suggest that engineering and design dimensions were prominent topics. On the other hand, an entirely separate set of sessions was dedicated to an array of topics in areas such as programming, numerical analysis, and applied mathematics (pp. 381-382).

While the content and organization of these symposia hinted at growing boundaries between the design and application – or engineering and scientific/ mathematical – dimensions of computer research and development, commentators like John W. Mauchly were explicitly commenting on the emergent social and technological boundaries of computing. In order to foreground these remarks, I turn to a brief description of the ENIAC. While this machine was one of the best known of the early electronic computers, it also played an important role in stimulating subsequent technological developments and new divisions of sociotechnical labor.[19]

Mauchly, the ENIAC, and the Machine-Instruction Boundary

Following a pattern that is well-known among historians of technology, the ENIAC's overall ease of operation was clearly limited by its rapid construction, esoteric design, and

[18] These two panels also tell us much about the shifting landscapes of computer research and development in the 1940s. Academic research centers represented at the conference included Harvard, MIT, the University of Pennsylvania, Princeton, and the Illinois Institute of Technology (IIT). Discussions of commercial developments included Eckert-Mauchly Computer Corporation, Engineering Research Associates, IBM, Bell Labs, and Raytheon.

[19] There is little question that many important computing devices preceded the ENIAC, including high-speed electromechanical machines like the Harvard/IBM Mark I and the Bell Labs series of relay computers. Yet the vacuum-tube-based ENIAC stands apart as one of the earliest, large-scale computers that was largely based on electronic components and technology (Van der Spiegel, et al., pp. 121, 123).

reliance on a host of new technological developments. In order to solve mathematical problems using the ENIAC, machine operators set myriad electronic switches and physically interconnected, via wires or bundles of wires, the various sub-units of the device. The configuration of the computer at any given time reflected a wide range of considerations, including the specific mathematical equations to be solved and the sequence of operations to be performed (Van der Spiegel, et al., 2000). As noted in many historical accounts, literally "rewiring" the machine to change its operation proved time-consuming and tedious, a reality that was evident even before the ENIAC was complete and operational.

One limitation of the ENIAC design that quickly became evident centered on the fact that the sparse memory of the machine could only be used for "data," while the machine instructions (that is, the operations to be performed on that data) were entirely "hardwired." In light of this issue, the ENIAC project is often credited with stimulating the design of "stored-program" computing machines. At a 1947 symposium, Mauchly provided one early discussion of this novel approach.[20] Describing a new class of computers called "EDVAC-type" (Electronic Discrete Variable Arithmetic Computer) machines, Mauchly identified three features that set these computers apart from previous designs: "(1) an extensive internal memory; (2) elementary instructions, few in number, to which the machine will respond; and (3) ability to store instructions as well as numerical quantities in the internal memory, and modify instructions so stored in accordance with other instructions" (Mauchly, 1948, p. 203). Mauchly's presentation was one of a handful of early attempts to describe the design characteristics of a true "stored-program" computer, where a much larger and more flexible machine memory could be used to store *and* manipulate both numerical data *and* machine instructions.[21] Indeed, historians often point to these characteristics as the central defining characteristics of a "modern" computer.

[20] Around this same time, Eckert and Mauchly left the Moore School to start Electronic Control Company, which was one of the first commercial manufacturers of high-speed electronic computers. The company was quickly renamed the Eckert-Mauchly Computer Corporation, and was purchased by Remington Rand in 1950 (Norberg, 2005).

[21] While the origins of the "stored-program" concept are not entirely clear, Akera notes that Eckert and Mauchly were exploring related ideas in as early as 1944. John von Neumann's well-known "First Draft of a Report on the EDVAC," released in 1945, is another key document that "contained many of the fundamental ideas for the stored-program computer" (Akera, 2002, p. 67). According to computer scientist and historian Michael Williams, the stored-program concept gained significant traction at the 1946 summer lecture series at Pennsylvania's Moore School (2002, pp. 23-24). Mauchly's remarks at the 1947 symposium were a clear extension of these efforts, as suggested by his use of the EDVAC acronym.

Yet Mauchly's comments about "elementary instructions" in the preceding passage point to another dimension of computer design that has received somewhat less attention in the historical literature. In the same Symposium presentation, Mauchly made one of the earliest attempts to identify and explore the boundaries that were forming around two major areas of activity in computing, namely machine design and "coding."[22] As Mauchly explained:

> A decision must be made as to which operations shall be built in and which are to be coded into the instructions. Reference has already been made to the uncertain status of division as a built-in operation. Many others, such as forming logarithm, cosine, arctangent, or square root, have been built into existing machines. Ultimate choice must depend upon the analysis by the designer of the character of the work to be performed by the machine, the frequency of occurrence of operations, and the ease with which non-built-in operations can be compounded from those which are built in (Mauchly, 1948, p. 205).

As suggested by this passage, it was evident that the design of a given computing machine needed to include some minimum set of elementary instructions for manipulating data and controlling program flow. Instructions such as add, shift or subtract, for instance, were literally hardwired in the physical machine and called upon as needed. But more complex operations – such as division –required more complex algorithms, where longer sequences of basic instructions (such as adding or shifting) were executed to achieve the desired results.[23] In fact, the study of such algorithms was a central concern for many of the mathematicians who were using computers to solve problems.

---

[22] As noted by Aspray, Mauchly's paper "was perhaps the earliest published discussion of the consequences of stored programming on logical design and programming" (1985, p. xvi). In contemporary terms, it should be noted that Mauchly's paper was one of the first to identify and discuss the nascent boundary between the physical "hardware" of the machine and the programs (or what would later come to be called the "software") that could be run on it.

[23] Building on many of the ideas developed by researchers such as Mauchly and von Nuemann, the first stored-program machines went into operation in the late 1940s and early 1950s. These included the Manchester University Mark I in 1948, the EDSAC (Electronic Delay Storage Automatic Calculator) at Cambridge University in 1949, and the Institute for Advanced Studies (IAS) computer at Princeton in 1952 (Randell, 2002, p. 41; Aspray, 2000, p. 73). Each of these machines dealt differently with the sort of design trade-offs described by Mauchly. The EDSAC, for instance, featured a set of 18 instructions for basic operations, including addition, subtraction, copy, multiply, and shift. Other instructions were also included, such as for transferring data to different memory locations, and for performing simple branching operations (Campbell-Kelly, 2002, pp. 415-416).

In terms of design trade-offs, when many complex operations are "built into" a machine, it can significantly streamline the work of the machine's users or operators, since they can easily call on these instructions as needed. Further, such operations can run more efficiently given that they are literally "hardwired" into the machine's logical circuitry. On the downside, built-in operations can greatly increase the complexity of a device, especially in the case of the early machines, when thousands or even tens of thousands of vacuum tubes were involved. Further, the particular algorithm used to implement any given operation might be inefficient or error-prone.[24] If such an algorithm is built directly into a machine, it can be exceedingly difficult to make changes or improvements, except by literally rebuilding some part of the computer, or perhaps by bypassing the existing operation with an entirely new set of programmed instructions. And as suggested by Mauchly, the intended application of a given computing device was a key consideration in determining which operations should be built-in, and which should left to be "coded" by the users of a machine. If one were designing a computing device to calculate tables for the firing of ballistics, for instance, it might be highly advantageous to include trigonometric functions in the machine's instruction set.

In light of Mauchly's remarks, we find that even in the mid-1940s distinct research and development activities were growing up around the "physical machine," on the one hand, and "coded instructions." Further, the essential point of negotiation between these two areas surfaced as an increasingly important issue in subsequent years, albeit often in tandem with new terminology and a host of related issues. Before tracing out the development of this theme, however, it is worth appraising the early role of the AIEE and IRE in the computing field. As suggested above, AIEE and IRE interests in computing gradually coalesced around a handful of common areas, including components and devices, systems, and certain areas of application. And as I demonstrate below, questions about the technological boundaries of the computing field quickly became intertwined with other concerns, such as the appropriate scope of organizations, the definitions of various professional identities, and even the actual design of electronic computers.

[24] Errors in hardware-based algorithms have repeatedly surfaced as a problem in the computing industry. One more recent and widely-publicized example involved a bug in one of the division instructions that was built into the Intel Pentium processor ("Pentium FDIV Bug," 2006).

The AIEE and Computing

While AIEE and IRE involvement in computing happened relatively early, the AIEE was the first of the two institutes to express formal interest in this new area of activity. In fact, the earliest direct evidence of AIEE interest in computing machines can be traced back to 1946. As mentioned above, a series of meetings at Columbia University was organized by the New York chapter of the AIEE in 1946 and 1947. In addition, a subcommittee on Large-scale Computing Devices was established in 1946 within the AIEE's larger Basic Sciences technical committee ("AIEE Officers and Committees," 1946, p. 1219).[25] According to General Electric power engineer Charles Concordia – who served as the first chairman of the subcommittee – much of the initiative behind this development came from AIEE President A. S. Lee and two successive chairmen of the Basic Sciences committee, John G. Brainerd and Julian D. Tebo (Concordia, 1976, p. 42). This push is not entirely surprising, given Brainerd's background as a Moore School graduate and professor, as well as his role as one of the main supervisors of the ENIAC project (Weiss, 1988). Concordia's interests, on the other hand, centered largely on analog computing devices, which had been used in the power industry since at least the 1920s.[26] Brainerd and Concordia's respective associations with digital and analog computing technologies symbolized two of the major areas around which AIEE activities were coalescing.

Yet in the mid-1940s, interest among engineers in the area of calculating and computing devices remained limited. In fact, Concordia later acknowledged that he faced some difficulty in forming the original group given that "there were not then a great many AIEE members familiar with the field" (1976, p. 42). He ultimately assembled the subcommittee around seven founding

[25] Aside from chapter activities, interest in particular subject areas was organized around AIEE technical committees and subcommittees. Through these groups, relatively small groups of engineers who were knowledgeable and interested in a topic or field would spearhead relevant activities, such as organizing panels and recommending papers for publication. The somewhat unusual position of the computing subcommittee within the basic sciences group is also worth noting because it reveals the difficulties that often came with positioning computing within pre-existing networks of disciplinary expertise, bodies of knowledge, and professional identities.

[26] In fact, Concordia co-authored a 1945 article on the use of "analyzers" (an early type of analog computer) in solving engineering and scientific problems (Peterson and Concordia, 1945). This paper reviewed four major types of devices, namely DC network analyzers, AC network analyzers, transient network analyzers, and differential analyzers. For more details on Concordia's career trajectory and areas of technical expertise and interest, see Kaplan (1999).

members.[27] And despite an apparent lack of familiarity and interest in the area of computing among the AIEE's general membership, Lee soon pushed for the elevation of the subcommittee to full committee status (Concordia, 1976, p. 42). The AIEE board approved this change in 1948, and the group was officially renamed the Committee on Computing Devices ("AIEE Forms Committee on Computing Devices," 1948). This group – which later was also frequently referred to as the Computing Devices Committee (CDC) – claimed nine members in 1948 and thirteen by 1949, with the most notable new member being Mauchly of Moore School and ENIAC fame ("AIEE Officers and Committees for 1948-1949," 1948, p. 1792; "AIEE Officers and Committees for 1949-1950," 1949, p. 802).

While the group's early focus was primarily on analog computing devices – especially as applied in the area of electric power systems analysis – it quickly moved to expand its scope. By 1949 Brainerd had taken the chairmanship and two subcommittees had formed, one led by Mauchly in the area of digital computing, and the other led by Harder and focused on "continuous-variable" computers – or "analog" computers, as they came to be widely known later ("AIEE Officers and Committees for 1948-1949," 1948, p. 1792). A third subcommittee dedicated to "computer bibliography" appeared in 1951, ostensibly formed to collect and manage the growing body of literature in the emergent field ("AIEE Technical Subcommittees 1950-1951," 1950, p. 942).

From the beginning, the orientation of the AIEE group tended to lean toward systems and components rather than applications. In one interview, Concordia explained that the first meeting of the subcommittee was focused on "computing devices, not on applications" (Concordia, 1994, p. 26).[28] This orientation was also evident in a statement of scope that was published after the group was elevated to full technical committee status:

---

[27] In addition to Concordia and Brainerd, the group included notables such as Samuel H. Caldwell, an MIT electrical engineering professor known for his research on differential analyzers with colleague Vannevar Bush, and Edwin L. Harder, a Westinghouse electrical engineer. Other members included J. D. Tebo, a Bell Labs engineer, Gilbert D. McCann, a Westinghouse electrical engineer who took a position at CalTech in 1946, and Princeton's W. C. Johnson. Regarding the selection of this group, Harder later explained that the members "were chosen by people that were already on the committees or the organization knowing that they were the knowledgeable people and that they should be there" (Harder, 1991, p. 49). As suggested by this remark, pre-existing social networks often played a central role in the creation and ongoing development of technical committees, both within and beyond the AIEE.

[28] Concordia added, "the next year [1947] we also had a small presentation on applications. It was harder to find people for that. Everyone wanted to talk about what was coming and what was new, or about what

> The scope of the committee is [t]he treatment of all matters in which the dominant factors are the requirements, design, construction, selection, installation, and operation of machinery and devices relating to computing devices, including studies of the electromagnetic, electronic, and mechanical phenomena of such devices. Fundamental mathematic, electronic, and properties of materials entering into these devices are not included ("AIEE Forms Committee on Computing Devices," 1948).

In addition to demarcating the group's boundaries in relation to a pre-existing milieu of dozens of other AIEE technical committees and subcommittees, this statement also distanced the group's activities from other areas of computing, such as those more closely linked to mathematics and "applications."

The AIEE computing group was instrumental in organizing a broad range of early publications and presentations, many featuring content that was rather generalized and introductory. This material served to familiarize the general membership of the AIEE with the computer field, potentially generating further interest in the topic. Much of this material appeared in the AIEE journal *Electrical Engineering.* Aiken and Hopper authored the first such article in 1946, and their three-part piece provided readers with a description of the Harvard/IBM Automatic Sequence Controlled Calculator, or Mark I (Aiken and Hopper, 1946).[29] In addition to briefly summarizing the history of mechanical computing machines, the article provided a lengthy discussion of the construction and operational aspects of the Mark I.

In subsequent years, news and articles about computing appeared in *Electrical Engineering* with increasing frequency. Other articles of note include a 1947 review of recent developments in electronics, which included a brief survey of "electronic computing devices" (Condon, 1947, pp. 355-256). A paper published in 1948, on the other hand, surveyed both historical and contemporary developments in the area of calculating machines, with an emphasis

---

they have been doing to develop computers. But it wasn't as interesting to talk about applications" (Concordia, 1994, p. 26).

[29] Others have recognized the historical significance of this particular article. It was reprinted, for instance, in Brian Randell's *The Origins of Digital Computers: Selected Papers* (1982). It is also worth noting that much of the article was adapted from the much longer and more detailed *A Manual of Operation for the Automatic Sequence Controlled Calculator*, an impressive 500+ page tome that was largely authored by Hopper and published in 1946 as the first volume of the *Annals of the Computation Laboratory of Harvard University* (Harvard Computation Laboratory, 1946).

on describing the operational and technical characteristics of the ENIAC, EDVAC, MANIAC, and UNIVAC computers (Tumbleson, 1948).

Also in 1948, Brainerd and fellow Moore School engineer T. K. Sharpless authored a more extensive article on the design, construction, and operation of the ENIAC (Brainerd and Sharpless, 1948). In the article's introduction, the authors noted that "[e]lectrical engineers in the United Stated have had a major interest in the development of large-scale computing devices" (p. 163). Brainerd and Sharpless also identified AC calculating boards, differential analyzers, and electromechanical machines (such as the Bell Labs relay computers and the Mark I) as noteworthy predecessors to the ENIAC. Given that many electrical engineers were already familiar with these and other analog calculating devices, the author's remarks provided further emphasis on the role of electrical engineers in the history of computing devices.

In addition to publications, the computing subcommittee arranged many conference sessions at AIEE district and general meetings. The first of these, at the 1947 Winter Meeting, featured "men from each of the six leading centers of computer development" ("Tentative Program, AIEE Winter Meeting,"1947, p. 78; see also "Large Scale Computer Developments Discussed," 1947). Notable speakers – including Aiken, Bigelow, McPherson, Sharpless, Forrester, and Williams – discussed both current and probable developments in digital computing, and the event attracted an impressive audience of about 350 (Concordia, 1976, p. 42).[30] A conference panel at the AIEE's 1947 summer meeting, on the other hand, was focused on the engineering applications of computing devices (Condon, 1947, pp. 355-356; "Program, AIEE Summer General Meeting," 1947, p. 594). As noted by Concordia, it proved more difficult to find speakers on this topic, and the session ultimately featured only two presentations and drew around 70 attendees (Concordia, 1976, p. 42). The relatively lack of engineers working in the area of applications was increasingly apparent by the late 1940s.

Computing devices and related topics continued to appear regularly in the late 1940s and early 1950s, both in the publications and in the technical programs of AIEE general and regional meetings. Through this period, AIEE panels and papers also tended to cluster around a handful

[30] Aiken discussed the Harvard/IBM Mark I machine, Princeton's Bigelow spoke on the design of the proposed IAS machine, IBM's McPherson discussed the history of difference engines and current uses of IBM calculating machines, Sharpless from the Moore School presented on the completed ENIAC and in-progress EDVAC, Forrester described the proposed MIT computer (later to be called Whirlwind), and Williams discussed relay computing developments at Bell Labs.

of major topics. In the digital area, systems and components attracted much attention, while applications received considerably less coverage. At the 1949 Winter General Meeting, for instance, a session on digital computers featured four papers on the description and design of systems and components, but only one on applications (“Electronic Digital Computers,” 1949).

As digital computers came to be viewed as increasingly “general purpose,” it helped pave the way for this deepened division between the areas of design and application. However, the situation was very different in the analog area. At the same Winter meeting, a session of five papers on analog computing revealed that analog machines were not so easily divorced from specific problem areas or types of applications. In addition to describing the design of various analog computers, this panel discussed how these devices were being applied in areas such as nonlinear mechanics, heat flow, electric power, vibration, and flight simulation (“Computing Devices Conference,” 1949). Interest in analog computing was particularly strong at many AIEE meetings in the early-1950's, and *Electrical Engineering* carried many articles and news items on the topic around this time. Yet by at least 1950, it was evident that digital computers might replace many analog calculators. As one review article noted, “One topic which has been discussed recently has been the question of whether a-c calculating boards may be replaced by some of the new large-scale electronic computers” (“1949 Engineering Developments,” 1950, p. 4).

In more general terms, the level of interest in computing among the general membership of the AIEE is difficult to gage. But in light of the long-standing orientation of the AIEE toward power engineering rather than electronics, it is reasonable to conclude that interest remained confined to a relatively small subset of the organization’s membership. Further, computer pioneer Herb Grosch retrospectively explained that those involved with the AIEE’s computer activities tended to be older, as well as “more conservative, and more old fashioned” (Grosch, 1971, p. 57). Another witness, Willis Ware, noted that the AIEE remained strongly oriented toward the increasingly marginalized area of analog computing, even well into the 1950s (Ware, 2005). In light of these and other factors, AIEE activities in computing were increasingly overshadowed by the IRE’s Professional Group on Electronic Computers (PGEC) in the 1950s.

The IRE and Computing: From Technical Committee to Professional Group

As noted above, the IRE's history was closely tied to electronics and communications, making the group's move into computing a somewhat natural extension of its scope. One of the earliest efforts to move into this area involved a session on "Electronic Digital Computers" at the 1947 IRE National Convention ("Extensive Plans Set," 1947, pp. 176-177). The panel was comprised of computer pioneers such as Forrester (MIT) and Goldstine (Moore School), as well as Samuel Alexander (National Bureau of Standards), Jan Rajchman (RCA), and Perry Crawford (Office of Naval Research). Further, the session was primarily oriented toward digital computing and covered a full range of topics, including g system design, input devices, component devices, and applications. Speaking on the topic of "Electronic Computing," Goldstine's rather general presentation was particularly noteworthy given its focus on the "interrelationship between the engineer and mathematician in the development of computing instruments" (p. 177). This session was one of four at the conference that was repeated due to over-attendance, suggesting significant early interest in computers among IRE members ("1947 IRE National Convention, 1947," p. 499). This interest was further reflected in the coordination of two panels on computers – one dedicated to "systems" and the other to "components" – at the IRE's National Convention in 1948 ("1948 IRE National Convention Program," 1948, pp. 377, 379).

An IRE Technical Committee on Electronic Computers was also formed in 1948, with an initial roster of 21 members. Initially headed by chairman James R. Weiner (Raytheon) and vice-chairman George Stibitz (formerly of Bell Labs, but by this time an independent consultant), the group included many other well-known figures such as Alexander, Eckert, and Forrester (Smith, 1991, pp. 6-7; "Technical Committees," 1948, p. 761).[31] And while the original definition of the committee's scope – approved by the IRE Executive Committee in 1948 – was rather broad, it did identify some of the group's major intended areas of activity:

> The Technical Committee on Electronic Computers is responsible for all work relating to digital and continuous computers. Included are applications to scientific computing, fire control, and industrial control problems. A primary duty of the Committee will include the compilation of a glossary of definitions ...

[31] Other notable early members included J. V. Atanasoff, J. H. Bigelow, Perry Crawford, C. S. Draper, N. Goldstine, E. L. Harder, B. L. Havens, E. Lakatos, G. D. McCann, C. H. Page, J. A. Rajchman, Nathaniel Rochester, Robert Serrell, T. K. Sharpless, R. Snyder, and C. F. West.

> Additional duties of the Committee include standardization of test methods, coordination with the Papers Procurement Committee, and computer session planning ("Executive Committee," 1948, p. 633).[32]

By 1949, the group had undertaken a number of activities characteristic for those working in a new field, such as the compilation of a computer bibliography and development of a list of computer definitions ("Technical Committee Notes," 1949, p. 63). The latter was published in the *Proceedings of the IRE* in 1951 ("Standards on Electronic Computers," 1951).

At a 1949 committee meeting, the group revisited its scope and subcommittee structure, especially with regard to giving "equitable coverage to analog and digital computers" ("Technical Committee Notes," 1949). Yet despite such concerns, numerous papers on systems and components, both in the analog and digital areas, were being published in the *Proceedings of the IRE* through the late-1940s. The group was also instrumental in organizing a slightly expanded presence at the IRE's 1949 national convention. Scheduled events included a panel on Electronic Computers that was oriented toward analog computing and a larger symposium on Electronic Computing Machines. The latter was primarily focused on "recent advances in the state of the art," especially in the digital area ("1949 IRE National Convention Program," 1949, pp. 165-166).

The committee's membership ebbed and flowed – hitting a high of 24 in mid-1949 and low of 16 by 1954 ("Technical Committees," 1949, p. 668; "Institute Committees – 1954," 1954, p. 1583). The group's leadership also shifted, with Jay Forrester gaining the chairmanship in 1949 and IBM's Nathanial Rochester taking the position in 1951 ("Technical Committees," 1949, p. 668; "Technical Committees," 1949, p. 721). However, the efforts of the technical committee were increasingly overshadowed by the emergence of IRE professional group activities in the computing field. The IRE's new professional group structure, which was formally adopted in 1948, allowed the diverse membership of the Institute to cluster more cohesively around special interests, including specific problem areas, particular technical interests, or various combinations thereof (Van Atta, 1950).[33] It also opened the way for the

[32] The term "continuous" slowly fell out of favor in the and 1940s and 1950s, and was largely replaced by "analog."

[33] Additional details about the new system appeared in the *Proceedings of the IRE* in 1948 ("The Institute on the March," 1948; The IRE Professional Group System," 1948). A summary and evaluation of the

formation of local professional group chapters, while simultaneously retaining technical committees to coordinate specific activities at the national level, such as standards setting. In light of the historical record, the emergence of the membership-based professional group structure was timely, for it helped pave the way for a rapid expansion of IRE activity in the computing field. By contrast, both the AIEE's orientation toward power engineering and its organizational reliance on a system of regional chapters and national technical committees limited AIEE involvement in computing to a relatively small group of members and range of activities.

In the IRE, interest in a new professional group in the area of electronic computers was expressed as early as 1948, but the first concrete steps toward this development happened in 1950 ("The IRE Professional Group System," 1948). In that year, the IRE's Los Angeles section organized its own Electronic Computers Professional Group, headed by Harry Huskey of the National Bureau of Standards Institute for Numerical Analysis at UCLA and Harry Larson of Hughes Aircraft (Astrahan, 1976, p. 43). While it was perhaps unusual for such a group to initially emerge at the local level, organization at the national level followed closely behind.[34] Nathaniel Rochester – at the time the chairman of the IRE's computing committee prodded IBM's Morton M. Astrahan to champion the formation of a national group. This led to the successful establishment of the Professional Group on Electric Computers (PGEC) in 1951, with Astrahan promoting the group on the East coast and Larson on the West ("Professional Group Notes," 1951; Astrahan, 1976, p. 43). The Los Angeles group was formally approved as a PGEC chapter in 1952 (Astrahan, 1976, p. 44). The PGEC's administrative committee was initially comprised of twelve members, jumping to fifteen under the group's first constitution. Astrahan served as the first chairman and Huskey first vice-chairman, again revealing the group's bi-coastal orientation (Astrahan, 1976, p. 43).

The professional group and technical committee operated in parallel for a number of years, with the former focused on the formation and organization of regional groups and meetings, and the latter's subcommittee structure used to organize work in other areas, such as

---

professional group system was published in 1950 (Van Atta, 1950). For a good secondary account, see McMahon (1984, pp. 215-218).

[34] The initial origins of the PGEC in the LA area ties into larger issues about the early emergence of distinct computing communities on the east and west coasts, each having a partially distinct culture of computer design and application. As Akera has noted, this bi-coastal division was also an important issue for the ACM in the 1950s (1998, Ch. 7).

terminology and storage devices. Yet technical committee activities were beginning to pale in light of the rapid growth of the PGEC. The professional group initiated a membership drive soon after it was founded, leading to a surging member roster. A report issued in late 1953 indicated that the size of the group had jumped to an impressive 1100 paid and 400 unpaid members (Astrahan, 1976, p. 44). And by October of 1953 the PGEC boasted 2000 members, and it claimed active local chapters in Los Angeles, Philadelphia, San Francisco, and Washington, D.C. (Gannett, 1953a). By September of 1954 the group had around 2,500 members, making it not only the largest Professional Group in the IRE, but also the largest computing-oriented professional group in the nation ("News," 1954).

Rapidly expanding membership was accompanied by an impressive scaling up of PGEC publication efforts. This flurry of publications from 1951 onward revealed an impressive expansion of the relevant material available for publication, in no small part due to the ongoing growth and diversification of the computer field as a whole. In late 1952, for instance, the PGEC founded its very own journal, titled the *Transactions of the IRE Professional Group on Electronic Computers.* Another important and related effort involved the group's extensive involvement in the October 1953 publication of a special computer issue of the *Proceedings of the IRE*. This issue was especially noteworthy for being roughly three times average size. The group also played a key role in the publication of proceedings from conferences and meetings. One such event was the as the 1952 Electronic Computer Symposium, which was organized by the PGEC's Los Angeles chapter in cooperation with the UCLA Department of Engineering. The PGEC also published the proceedings of the well-known Joint Computer Conferences (or JCCs).

In summary, the mid-1940s computing field was characterized by isolated researchers and research groups, scattered publications and commentary, one-off conferences, and one-off machines. The flurry of publications and conference sessions from the late-1940s onward – both within and beyond the AIEE and IRE – hinted at the emergence of a more cohesive computer field, albeit one that was undergoing considerable growth and diversification. More specific trends were also becoming evident, such as a shift from universities to industry as the principal site of computer design and construction, as well as the release of the first commercially available computers. Even more importantly, we find the origins and early negotiation the key sociotechnical boundaries in the field. The remainder of the chapter uses the early joint computer conferences as a window onto these and other themes.

The Joint Computer Conferences

By all appearances, the early relationship of the IRE and AIEE computing committees was generally congenial and cooperative. By the late-1940s, the two groups were working together to develop definitions and bibliographies for the computing field. And from the mid-1940s to early-1950s, at least five individuals were at some point simultaneous members of both the AIEE and IRE computing committees. Indeed, such cooperation was not unprecedented, given that the IRE and AIEE had a long history of working together in areas of common interest. This cooperation was also reflected in the Joint Computer Conferences (JCCs). In historical terms, these conferences were significant in setting the stage for later developments, including the formation of the American Federation of Information Processing Societies (AFIPS) in 1961. But more importantly for the present analysis, surveying the content and scope of the early joint conferences provides further insights regarding the evolution of the computer field's major social and technological boundaries.

The origins of the joint computer conferences can be traced back to a series of meetings on electron tubes that were co-organized by the AIEE and IRE. In fact, by the late-1940s it was widely recognized that electron tubes were a crucially important computer system component, as evidenced by a talk on the topic of "Digital Computers" at the 1948 joint conference on electron tubes. Presented by Moore School engineer R. L. Snyder, this presentation emphasized two major issues, namely ongoing efforts to both improve the reliability of existing tubes and develop new tubes specifically for computers ("Tentative Program, Conference on Electron Tubes," 1948).[35] In late 1950 the AIEE and IRE co-sponsored a two-day conference that was more specifically dedicated to the topic of "Electron Tubes for Computers," and it drew more than 300 attendees ("Five Sessions Held at Conference," 1951).[36] According to one summary, important stimulus for the event came from a survey that revealed a need for further

[35] Snyder's presentation revealed two distinct types of technological change. On the one hand, off-the-shelf components might be applied in entirely new ways in new technological systems. Second, existing component devices may act as a springboard in the development of new or improved components, often for use in particular applications or systems.

[36] The conference was also a result of collaboration with the Panel on Electron Tubes, a part of the Department of Defense Research and Development Board.

"clarification" in this area.[37] While much of the conference was focused on tube reliability, design innovations and manufacturing challenges were also discussed.

On the one hand, the computer tubes conference revealed the extensive involvement of both the AIEE and IRE in the area of electronic devices generally, and computer components specifically. On the other hand, the conference set the stage for subsequent events. As Akera explains, the "enthusiasm displayed at this meeting convinced both groups to organize a larger, regular meeting in the computing field" (1998, p. 577). Others point to John Brainerd, the second chairman of the AIEE Committee on Computing Devices, as the first to propose the idea of large computer conference ("Reflections on a Quarter-Century," 1986, p. 226). The resulting gathering – which was dubbed the Joint Computer Conference (JCC) and topically titled a "Review of Electronic Digital Computers" – was held in December of 1951. With 900 attendees, it was one of the largest computer conferences to date ("Joint AIEE-IRE Computer Conference," 1952). In late 1952, more than 1100 attendees were attracted to a second joint computer conference that was focused more specifically on the use of input-output equipment in computing systems ("Record Attendance at Computer Conference," 1953). In 1953 the conference went bi-annual, with meetings alternating between the east and west coasts.

As I will discuss in subsequent chapters, the planning and scope of the joint conferences in the 1950s and 1960s brought into further relief the social and technical contours of the computing field. Yet the first JCC stands out as particularly important, not only because it occurred at a pivotal time, but also because the scope and tenor of the conference program revealed important themes and trends. In the following section, I use the JCC to highlight three such themes. First, the event's program and speakers placed both implicit and explicit emphasis on the role of engineers and engineering in the computing field. Second, the conference provided some of the earliest evidence for the emergence of a distinct and identity and bounded area of expertise for these engineers, especially through the use of terms such as "computer designer" and "computer engineer." And finally, the event hinted at how the major subfields of computing were developing in relation to one another, as well as in relation to computer technologies.

[37] The survey was conducted by Samuel Alexander, an engineer who would become well-known for his computer-related work from at the National Bureau of Standards from 1946 onward.

Positioning Computers in Engineering

In most general terms, the inaugural JCC was notable in that it was largely organized by engineers and primarily focused on engineering. In fact, the event was explicitly dedicated to exploring the "engineering aspects" of computer design and construction, and most of the presentations and papers were oriented accordingly. As indicated in a foreword that was included with the published conference proceedings, it was suggested that the meeting "would be of permanent value in the development of engineering knowledge of this new field of activity" ("Foreward," 1952a). And in delivering a closing summary and address to attendees, Jay Forrester emphasized this theme by pointing to the "magnitude of engineering involved" in the building of the early digital computers (Forrester, 1952, p. 109). He also noted:

> A comparison of the present status of the digital computer field with any of our older branches of engineering shows that we are not far advanced. We are firmly on the threshold of a new field, but the digital computer work has reached no real maturity. … We have first models of a new type of machine. There is no reason to believe that they are relatively more advanced than were the first models of automobiles, the first aircraft, or the first radio sets (p. 109).

Forrester's remark clearly suggested that the technology of computing fell within the jurisdiction of engineers. Further, his reference to other major technologies implied that the ongoing expansion of the computer field would require larger numbers of engineers and large amounts of engineering expertise and knowledge.

Yet Forrester was by no means the first to comment on the role of engineers and engineering in the computing field. As noted above, in a 1948 article Brainerd and Sharpless emphasized the historical position of electrical engineers in computer development. And Forrester himself had commented on the topic in a panel paper titled "Outlook for Electronic Digital Computers – The Scope of the Engineering Involved," presented in January of 1949 at the AIEE's Winter General Meeting. As described in one summary report on the session, Forrester used this particular presentation to call for additional study and research in the area of "systems engineering" ("Electronic Digital Computers," 1949, p. 266).[38]

[38] According to one review of the panel, Forrester framed systems engineering as "the integration of computer components into equipment" ("Electronic Digital Computers," 1949). Further paraphrasing

From the early 1950s onward, commentary on the position of engineers and engineering knowledge in the computer field – as well as the more specific role of electrical and electronics engineering – surfaced with increasing regularity. One important source of evidence for this trend can be found in a 1950 treatise by electrical engineer Lofti Zadeh. While in later years he became well-known for his work in areas such as system theory and fuzzy logic, the article appeared just after Zadeh had completed a Ph.D. in electrical engineering at Columbia University. Suggestively titled "Thinking Machines: A New Field in Electrical Engineering" (1950), this particular piece is probably the first extended discussion of the engineering-computing relationship to appear in print.

Discussing the emergence and larger implications of "electronic brains" or "thinking machines," Zadeh started the article by asking: "[W]hat is the role played by electrical engineers in the design of these devices?" (p. 12).[39] Reponding to this question, the author went on to emphasize the role of mathematicians, both in the historical and contemporary development of computing devices:

> Thinking machines are essentially electrical devices. But, unlike most other electrical devices, they are the brain children of mathematicians and not of electrical engineers. Even at the present time most of the advanced work on thinking machines is being done by mathematicians (p. 12).

Zadeh returned to this theme later in the article by adding that "[i]t is true that most of the fundamental principles on which thinking machines are based, have been contributed by mathematicians" (p. 31).[40]

While the historical record suggests that Zadeh was guilty of exaggeration in these passages, his remarks were nonetheless well-suited to his forward-looking agenda. He went on to emphasize, for instance, "the ability of electrical engineers to supply the techniques that make possible the storage devices, processors, computors [sic], decision makers, and other less important elements of thinking machines" (p. 31). And after noting that engineers had been

---

Forrester's remarks, the summary added: "Greater study must be applied to co-ordination of computers with communications systems and automatic control devices."

[39] From the late-1940s onward, terms such as "electronic brains" or "thinking machines" were often used to describe high-speed electronic computers, especially in more popularized accounts.

[40] While Zadeh did not specifically mention the role of scientists in the computing field, his use of the mathematician label likely included some scientists, especially in mathematically-intense fields such as physics.

exposed to computing-related subjects such as Boolean algebra and multivalued logic through their association with mathematicians, the author added that the dominance of mathematicians in the field "will last until electrical engineers become more proficient in those fields of mathematics which form the theoretical basis for the design of thinking machines" (p. 31).[41] The strategy implied by Zadeh was fairly straightforward. If electrical engineers could meld their existing knowledge of electronic components and systems with the appropriate mathematical moorings, they might lay the foundations for a new branch of engineering focused on computers and computing. And indeed, computer pioneers such as Forrester and Eckert – not to mention pioneering research groups such as MIT's Computation Laboratory and Penn's Moore School – had already demonstrated the potential success for this type of approach to computer research and development.

While Zadeh's commentary was thought-provoking and provocative, the overall impact of the article was probably limited, especially given that the publication in which it appeared – *Columbia Engineering Quarterly* – was probably not read widely outside of some relatively small circles of engineers. Yet the significance of the article for the present analysis is two-fold. First, it was likely Zadeh's first attempt to formulate a disciplinary agenda for computer-oriented electrical engineers. In fact, he would revisit, refine, and pursue this agenda with vigor in later years. And second, the article raised important questions about the position of engineers with respect to computing, Such questions were increasingly salient from the early-1950s onward.

Looking beyond the aforementioned JCC, for instance, reveals a 1952 computer symposium organized by the IRE-PGEC's newly-formed Los Angeles chapter. The title of the meeting, "Engineering Tomorrow's Computers," reflected both the event's intended audience and its topical orientation. One speaker – computer pioneer and self-described "human computer" Ida Rhodes – pandered to this theme by referring to the early electronic computers as a "brainchild of electronic engineers," and she went on to praise the "achievements of electronic engineers" in the computing field (1952, pp. XII:1, XII:4). Such remarks were not confined to conference proceedings. An introductory article published in the special 1953 computer issue of the IRE's *Proceedings*, for instance, noted that "[t]he design of a successful computer demands a high degree of engineering skill" (Buchholz, 1953, p. 1220). Not to be outdone, the editors of the

[41] It is possible that much of the motivation behind this argument came from Zadeh's own impressive abilities in various areas of mathematics. I return to this point in subsequent chapters.

same issue described “the growth of Electronic Computers as a branch of the radio engineering field” (Gannett, 1953b).

In summary, the evidence presented here reveals a number of early moves to describe and probe the relationship between electrical engineering and computing. And while some commentators simply pointed to the historical role and/or contemporary position of engineers in the computing field, authors such as Zadeh hinted at a more ambitious forward-looking agenda. The development of this agenda was intertwined with the emergence of a more distinct and cohesive identity for these same engineers, and also linked to the idea that computing could indeed be framed as a “new branch” of electrical engineering.

Computer Engineering Identities

The use of pre-existing labels and titles to describe the engineering dimensions of computer development can be traced back to the early days of the field. Terms such as “electronics engineer” or “circuit designer,” for instance, were regularly applied from the 1940s onward to those engineers who worked on computer systems and components (Felker, 1952b, p. 1584). More specific labels also started to appear in the late-1940s and early-1950s. Not only did these titles become linked to specific areas of expertise, they also played an important role in the ongoing efforts of electrical engineers to claim major areas of the computing field as their own. Some of this boundary-work was explicit, such as when commentators mentioned or defined new identities for computer-oriented engineers. Additional evidence for these trends can be gleaned from the scope and content of various publications and conferences.

The most prominent identity markers that were surfacing in the early 1950s were clearly situated at the intersection of computing and engineering. The term “computer designer,” for instance, both took hold early and emphasized the dominant image of engineering work as linked to the theory and practice of design (“Radio Progress During 1951,” 1952, p. 430). “Computer engineering” also gained currency around this time, mirroring the prior emergence of engineering titles that were linked to specific areas of technology, such as “radio engineering” and “power engineering.” The importance of these new engineering titles is two-fold. First, they facilitated the extension of the major engineering fields into entirely new domains of technology and technological knowledge. Second, the use of the “engineering” moniker provided these new fields and subfields with convenient and pre-existing disciplinary and professional structures,

ranging from conferences and publication outlets to professional societies and curricula. In an important sense, the identity of a radio engineer or computer engineer is as much about engineering as it is about a specific domain of knowledge and technology.

With regard to "computer engineering," the term can be traced back to at least the first JCC, held in 1951. One conference announcement explained that the meeting was "held specifically to review accomplishments in the relatively new field of large-scale digital computer engineering" ("Joint IRE/AIEE Computer Conference Slated," 1951). And while another report noted that "more than 900 engineers, scientists, and mathematicians" ("Joint AIEE-IRE Computer Conference," 1952) attended the conference, a somewhat oblique comment made by one of the keynote speakers at the conference implied that "computer engineers" were his primary audience (MacWilliams, 1952, p. 6).[42] And if the conference was ostensibly dedicated to the general area computer engineering, the conference program hinted at the scope of this field of activity. With an overarching emphasis on "the characteristics and performance of working, large-scale electronic digital computers," the vast majority of the papers and published discussion were focused on the individual computing machines, with particular emphasis on engineering and design challenges, performance characteristics, and issues of reliability ("Foreward," 1952a, p. 3). Only one paper more narrowly dealt with computer components, although the author's discussion of the possible uses of transistors in computers was an important early exposition of the topic, especially when vacuum tube technology still dominated computing (Felker, 1952a).

On the other hand, topics such as applications, programming, and analog computers garnered only scattered commentary. One noteworthy exception was a paper on the possible uses of both analog and digital machines in the solution of aircraft engineering programs. As noted in a foreword published with the conference proceedings, this presentation "gave the members of the conference a better understanding of the ultimate usefulness of their efforts" ("Foreward," 1952a). Yet this particular session was relegated to a luncheon meeting on the last day of the

[42] MacWilliams stated: "One could say really that we have been optimists to schedule a meeting like this. We feel that in addition to keeping computer engineers employed – in itself a praiseworthy objective – a great deal of worthwhile experience has been obtained from the perhaps $30,000,000 that have been spent so far on large high-speed digital computers. It is important to get the most out of the experience resulting form this large amount of work, so that our new machines can be made as good as possible" (p. 6). In addition to identifying computer engineers as a distinct group of workers, this comment also suggests that the title was gaining particular salience in the commercial sector. I return to this point below.

conference, suggesting that the conference organizers viewed the area of computer applications as separable from – and perhaps even tangential to –computer design and engineering.

While the topic of computer applications received short shrift on the official schedule, a series of informal sessions were hastily convened to "discuss problems arising in programming" (Carr, 1952, p. 113).[43] Interested conference attendees met to address topics such as computer operating procedures, the prospects for universal machine operating codes, and methods for preventing and locating programming mistakes (p. 113-114).[44] As noted in the conference review, only eighteen individuals participated in these sessions, and the small discussions that did take place were marked by an "absence of mathematicians and programmers." (p. 114). These informal gatherings – as well as discussions of applications and programming more generally – were clearly a footnote to an event that was almost wholly dedicated to computer design and engineering.

Surveying other publications from around this time period reveals that the use of the term "computer engineering" at the first JCC was part of a more general trend. The term surfaced again, for instance, in the special 1953 computer issue of the IRE's *Proceedings*. In an introductory article, IBM engineer Werner Buchholz explained that the special issue was intended:

> To provide a set of stimulating and informative articles which would introduce the non-specialist reader to the new and exciting field of electronic computer engineering, and to furnish the specialist with a single volume of reference material on a wide variety of computer subjects (Buchholz, 1953, p. 1220).

In another paper in the same issue, non-specialists were treated to an elementary introduction and overview of computers that explicitly emphasized "the 'lingo' of the computer engineer"

[43] This analysis is based on a written summary of these informal discussions, submitted by mathematician John W. Carr III and published at the very end of the conference proceedings. While I will revisit Carr's work in the next chapter, this was one of his earliest attempts to raise the visibility of programming issues among computer designers.

[44] The use of the term "operating codes" here refers to the various machine instructions that are used to control and direct the operation of a given computer. At the time, each computer had its own unique set of operating codes, and these codes served as the fundamental building blocks for the development of more complex sequences of operations, or "programs." The discussion at the JCC explored the tentative possibilities for a universal set of codes for all machines. These might be built into a given machine, or run through an "interpreter" that would convert universal codes into machine-specific instructions. The idea of developing higher-level languages followed closely behind.

(Samuel, 1953, p. 1223).[45] While these passages revealed the increasing importance of computing – even for IRE members whose primary interests lay elsewhere – they also suggested the emergence of both a new field called computer engineering and a new identity for the so-called "computer engineer."

If the first JCC and the 1953 special issue hinted at the anticipated purview of computer engineering, early issues of the *Transactions of the PGEC* both named this new area of activity and provided a more explicit outline of its scope. A forward in the first issue stated: "It is hoped that this issue will be the start of a major publication in the field of digital and analog computer engineering" ("Foreward," 1952b). And in a second issue, the editors further expressed their belief that the membership of the PGEC was expected to be principally interested in "hardware," adding that papers about the "physical components of which computers are made ... are the backbone of an engineering journal" ("Editorial," 1953). Not only did these remarks provide a definition for the term "hardware" in the context of computing, they also linked this term to the new field of computer engineering and promoted the *Transactions* as its preferred journal.

Employing Computer Designers and Engineers

As suggested by the preceding analysis, the general area of computer design and engineering was emerging as an increasingly distinct area of activity from the early 1950s onward. Yet it is important to note that this formative subfield was developing in tandem with the growing dominance of private-sector industry in the design and construction of computers. While this trend has been well-documented by others, it is worth summarizing that the mid- and late-1940s were marked by a handful of commercial computing ventures that were slowly gaining momentum. The trend accelerated in the early 1950s, with many new and existing companies entering the field. And as the industry grew, computer development activities at universities entered a period of relative decline.

As noted above, Aiken was one of the first university researchers to explicitly distance his lab from the area of computer design and construction, perhaps not surprising given both Harvard's unfriendly stance toward engineering and Aiken's long interest in the application of computing machines. Hinting at larger trends that were afoot, in 1949 Aiken explained:

[45] The rich and expansive lexicon of the computing field and its subfields forcefully reveals the importance of terminology in ongoing efforts to define and delineate particular areas of disciplinary activity and expertise.

> Therefore, at our laboratory we have decided not to undertake the construction of any more large-scale computing machines with the exception of one, which we hope to build for our own use and keep at Harvard. There is an ever-increasing number of industries interested in constructing computing machines outside the universities (1951, p. 7).

Aiken's observation certainly captured trends that were afoot, and by the mid-1950s there were at least a dozen major commercial outfits producing digital computers (Flamm, 1988, p. 81). And while some schools maintained active research programs in computer design during this period, commercial computer research quickly overshadowed university research.

Further, the expanding commercial sector appeared to be the primary locus of the new subfield of "computer engineering." As one piece of evidence for this trend, the organizing committee of the first joint conference – which was substantially oriented toward engineering and design – was almost entirely dominated by those whose primary affiliations were in the commercial sector ("Joint AIEE-IRE Conference Committee," 1952). Even more importantly, a canvass of employment listings in a number of major publications reveals that the term "computer engineering" quickly gained currency from the early 1950s onward, both within budding computer companies and beyond. These listings also reveal the major areas of expertise that were being linked to this newly demarcated area of computer work.

The earliest examples of this trend can be traced back to at least 1952. Surveying the many open positions published each month in the AIEE's *Electrical Engineering* reveals an October 1952 listing for an "electronic or computer engineer" with a B.S.E.E. or M.E. (Engineering Societies Personnel Service, Inc., 1952, p. 86A). Also in 1952, a series of ads from the Gilfillan Corporation that called for "experienced radar and computer engineers" were published in the *Proceedings of the IRE* (Gilfillan, 1952a; 1952b).[46] Later in the same year, Engineering Research Associates (ERA) – which was founded in the mid-1940s as one of the first commercial developers of computer equipment – was similarly seeking "digital computer engineers" (Engineering Research Associates, Inc., 1952a; 1952b; 1952c). These ERA ads more specifically called for electrical engineers and physicists with expertise in the design and development of circuits and system. A 1954 ad from ERA, on the other hand, called for "electrical engineers and physicists to do digital computer engineering" (Engineering Research

[46] Gilfillan was particularly active in radar research and development around this time.

Associates, 1954). And in the same year, the Jet Propulsion Laboratory at Cal Tech posted an opening for "Computer Engineers (Analog and Digital)," with specific emphasis on circuit design, logical design, transistors, and "theory of automatic digital computers" (Jet Propulsion Laboratory, 1954).

From the mid-1950s onward, employment listings from a variety of companies solicited "computer engineers" with increasing frequency. Other ads from around this time omitted this specific term, but nonetheless called for electronics engineers and physicists with expertise in computing and related areas.[47] These advertisements also tended to avoid reference to mathematics, programming, or numerical analysis. For starters, this suggested that the ideal prospective employees for computer design and development work were male engineers, albeit with some room for research scientists. Corporate employers were likely eager to employ computer designers and engineers who had been trained as engineers, and who could be expected to behave as predictable "professionals." Further, the relatively large number of postings for computer-oriented engineers revealed an early division of labor between the design and application aspects of computer development, as well as a relatively low level of early demand for application-oriented workers. As I discuss in the following chapter, openings for computer programmers, numerical analysts, and related positions appeared more regularly in the mid-1950s and beyond. Yet terms such as "computer engineer" and "computer engineering" persisted, and they remained closely linked to research and development activities in the commercial sector.

The Relational Ontology of Computer Engineering[48]

Like earlier terms such as "radio engineering," the "computer engineering" moniker was coupled with a specific and relatively young technology. It also encompassed a broad array of engineering sub-specialties – including electronics engineering, circuit design, and systems engineering – that played central roles in the computing field. Further, the term was starting to subsume new areas of expertise, such as logical design, that were growing out of work in the computing field. Yet the area of computer engineering was not emerging in isolation, and the

[47] For instance, a 1954 Hughes ad that carried the heading "Digital Computer Techniques" called for engineers, physicists, and "computer applications specialists." More specific areas of expertise listed in the ad included logical design, component development, programming, circuit design, and systems analysis (Hughes Research and Development Laboratories, 1954).
[48] On the concept of relational ontology, see Breslau (2000).

definition of “hardware” was not without contestation. It was increasingly necessary to position the field and its associated technologies with respect to the wider disciplinary and technological landscape of computing.

As the general area of computer design and computer engineering gained a more cohesive identity and scope, a number of commentators explicitly discussed how the major subfields of computing were related. These remarks often centered on the boundaries around two major areas, the first centered on design, engineering, and “hardware,” and the second encompassing applications, users, and programming. While this particular theme can be traced back to Mauchly’s aforementioned remarks at the 1947 symposium, it was revisited with increasing frequency from the early-1950s onward. At the first JCC, for instance, we find echoes of Mauchly in Forrester's closing remarks: “A great deal of machine time can be saved by analyzing computing programs and providing special machine logic or facilities for saving time in the more frequent types of operations” (1952, p. 113).

While this type of trade-off was an important issue on its own, it was also closely linked to the boundaries around the activities of computer design and use, as well as the identities of computer designers and users. A keynote address at the same meeting, delivered by Bell Labs engineer W. H. MacWilliams, added that one of the major objectives of the conference was:

> [T]o assess the adequacy of the designs of present working high-speed digital computers in order to point out the direction in which computer design should go, to make computers best for the jobs that they have been doing and for the jobs that they will have to do. This is basically an engineering or design objective, but it is clear that it also involves the users in an important way. This is a meeting of both builders and users, all of whom are actively interested in the field (1952, p. 5).

As suggested by this remark, by the 1950s computer builders and users were being portrayed as increasingly distinct groups, each linked to particular areas of expertise, activity, interest, and technology. Yet given that the two groups were united under the larger umbrella of the computing field, their emergence and ongoing development were necessarily happening in relation, not isolation.

MacWilliams's comment also revealed that the budding divide between computer builders and users was accompanied by a growing recognition that computer designers needed to more actively study how computers were being applied, especially as they worked to refine existing computers and imagining new designs. Further, much of this critique was coming from mathematicians and other computer users. For instance, Murray Lesser of Northrop spoke at the 1952 LA symposium about the challenges of using high-speed calculating equipment for solving engineering problems. Playfully chiding engineers for their lack of knowledge regarding the actual use of computers, Lesser explained: "Although the viewpoint about to be expressed appears to be largely ignored by the designers and builders of the new breeds of automatic high-speed digital computing machines, it is the opinion of this writer that the primary reason for the existence of such devices is to aid in the solution of problems" (Lesser, 1952, p. IX:1).

At the same event, mathematician Derrick Lehmer pushed in a similar direction when he mentioned the lack of programming knowledge among engineers: "We had a little session on coding this morning. I think a number of engineers got a clearer picture of how important this part of computing has to be" (1952, p. XX:2). Lehmer also echoed Goldstine's aforementioned 1947 conference summary as he commented on the budding tensions between the engineers and mathematicians involved the computing field. "The engineer and the mathematician are involved in a joint effort in this particular field," Lehmer stated, adding that "this symposium ought to record ... the possibility of cooperation between these two groups" (p. XX:2).

Early editorial remarks in the *Transactions of the PGEC* provide further evidence regarding the more general position of engineers with respect to computing. For starters, the aforementioned emphasis on computer engineering and "hardware" by the editors of the publication was accompanied by efforts to frame programming and applications as largely beyond the bounds of the journal: "We may think of programming as relating to applications and being outside the sphere of interest of most computer engineers" ("Editorial," 1953). Yet the same editorial acknowledged that the topic of programming might prove relevant to the journal's audience, particularly when it was clearly related to issues of computer design:

> It is a fairly recent discovery that, with general-purpose computers, we can replace hardware by programs [...] The design of suitable programs is analogous to the design of the computer circuits or the development of the internal logic. We might call it program engineering, for the existence of good programs to assist in

> running a computer can be as vital to its success as good circuits. We plan to publish papers on a wide range of subjects, including circuits, components, systems, input and output, logic, and "program engineering" ("Editorial," 1953).

On the one hand, this statement is striking in that it subtly foreshadowed the much later development of "software engineering" as a new subfield of computing.[49] But more importantly for the present analysis, the editorial revealed a central point of tension in the field's nascent social and technical boundaries. To whit, if hardware could be replaced by programs (and vice-versa), to what extent were the boundaries around the areas of computer engineering and computer programming justifiable or maintainable? This tension, I contend, is a primary and persistent source of instability in ongoing efforts to bound off and define a field of computer engineering.

Hopper and Mauchly also revisited the design-programming relationship in a 1953 article titled "Influence of Programming Techniques on the Design of Computers" (Hopper and Mauchly, 1953). In general, the article provides further evidence for the major divisions of labor that were increasingly common in the field. The authors described design engineers as being principally concerned with circuits and "hardware," while programmers were mainly focused on "discover[ing] new ways of adapting the computer to particular applications" (p. 1250). Sketching out the relationship between these two groups, Mauchly and Hopper added:

> This relation between the programmer and the designer of computers is by no means a static one. While the engineer is developing new components and better ways of using such components, the programmer is likewise developing new techniques for the application of computers and is continually enlarging the range of applications as well. ... The development of new techniques in programming may have as profound an influence on computer design as would be produced by an entirely new type of memory or switching element (p. 1250).

This rather taken-for-granted description of the two subfields is noteworthy, especially given that the tentative boundaries around these areas had only emerged a few years prior, largely in tandem with the advent of the first stored-program, general purpose computers.

Yet Mauchly and Hopper's major concern in this article centered on the relation of the two groups, rather than on their definition or even existence. In fact, they followed in the

[49] On the history of software engineering, see the work of historian Michael Mahoney (1990; 2004b).

footsteps of earlier commentators as they pushed computer designers to grapple with the concerns and techniques of programmers. “Certainly the programmer must help the engineer in evaluating proposed engineering plans” they stated, adding that “he can often suggest possibilities for the engineer to consider. Sometimes a relatively minor design modification can result in savings in programming” (p. 1250). In the remainder of the article, Mauchly and Hopper outlined a number of specific areas where programming developments had informed – or could potentially inform – the design of computing machines.

In a 1953 article on the topic of “compiling routines,” Hopper pushed in similar directions (Hopper, 1953). Noting that computers had already been designed with instruction sets of widely varying sizes – ranging from a total of about eight to eighty individual machine orders – Hopper explained that decisions about the composition of a machine’s instruction set were often linked to key design trade-offs, such as ease of programming versus machine complexity. Coming down in favor of relatively small instruction sets, Hopper described how various techniques – including the use of programmed subroutines – could be used to tailor general-purpose computers for particular applications. She also stressed the need for close cooperation between programmers and engineers in the design and building of these machines:

> [I]t is desirable that programmers work side by side with logical designers and engineers at the time that the design of a computer, large or small, is begun. Thus, a computer will be delivered with its basic programs tested and proven, ready to be used flexibly and conveniently (pp. 1-2).

Yet this call for reform seemed to be gaining little traction outside of a handful of outspoken commentators, many of whom happened to be mathematicians and programmers. Computer-oriented engineers might acknowledge the relation between programming, applications, and computer design, but speaking of close cooperation with programmers was all but taboo.

It is also worth noting that the influence of Hopper’s remarks was further blunted by the fact that she was one of only a handful of women in the computing field. Through the early-1950s, it had become abundantly clear that while women might make their way into computer programming, their contributions in the area of computer design were almost entirely limited to documenting existing work, and perhaps even commenting on it. Hence, it is perhaps not surprising that Hopper’s 1953 article was relegated to Edmund Berkeley’s upstart computer journal *Computers and Automation*, while her article with co-author Mauchly – who, unlike

Hopper, happened to be a member of IRE – was published in the more prestigious and technically-oriented *Transactions of the IRE*. No matter the impressive the technical expertise accumulated by pioneers such as Rhodes and Hopper, the early computer engineering field was all but impermeable to women, as well as to others without the appropriate credentials, experience, or identity.

Conclusion

In this chapter I have discussed the emergence of a more recognizable computer "field" from the mid-1940s onward, as reflected in the scope and scale of numerous meetings, symposia, publications, and computer development projects. Yet I also emphasize the countervailing forces that were simultaneously deepening the field's social and technical boundaries, leading to an incipient tendency for fragmentation. These themes are brought into further relief as we look more specifically as the role and position of engineers and engineering knowledge in the early development of high-speed computing. As I have argued, electrical engineers made a number of early moves to bring computers *into* engineering, just as their predecessors had done with radio decades before. Authors such as Zadeh, for example, clearly painted the computing field as a new branch of electrical engineering. Further, the term "computer engineering" created an important semantic link between the pre-existing professional identities and practices of engineers, on the one hand, and the new activities and bodies of knowledge that were associated with computer design and construction, on the other.

As this vision for a field of computer engineering started to materialize in the early and mid-1950s, it was also increasingly evident that the interests of electrical engineers were limited in scope. Events such as the first JCC were almost exclusively dedicated to the engineering aspects of computing, and the IRE-PGEC explicitly itself positioned at the intersection of computer design, engineering, and "hardware." Early issues of the PGEC's *Transactions* went so far as to explain that topics such as such as numerical analysis and programming were largely tangential to computer engineers, except where application and design were most directly and obviously related.[50] As suggested by this historical account, parceling off computer design and

[50] By this time there were growing numbers of engineers who were interested in the use of high-speed digital computers for engineering problem solving, but these were rarely the same engineers who were doing computer design work. On the other hand, analog computing was an area where the users and

engineering as a distinct field required acts of boundary definition and negotiation that were as much social as they were technical. To put it another way, the proponents of this disciplinary project were engaged in extensive sociotechnical boundary-work.

But just as the social and technical foundations for a more recognizable field of computer design and engineering were laid, large swaths of computing were being claimed by other interested individuals and groups, many of them without strong ties to engineers or engineering. This trend can be traced back to the prominent early role of mathematicians and scientists in computing, as well as to early conferences and publications where topics such as computer programming and applications were frequently separated from discussions of computer design and engineering. As I document in the following chapter, the major social and technical divides between the design and application dimensions of computing persisted through the 1950s and into the 1960s, as evidenced in the ongoing evolution of educational programs, professional groups, and university-industry relations, to name a few important themes.

By the early 1950s, however, prescient commentators such as Lehmer, Mauchly, and Hopper were already recognizing some of the possible consequences of the field's deepening social and technical boundaries. More specifically, they argued that closer cooperation between designers and programmers might be an important step toward making computer systems more flexible, reliable, and easy to use. While the vision offered by these pundits was both compelling and rather straightforward, it tended to obscure the challenging social and technical realities that were at the heart of the situation, and that would grow increasingly salient in subsequent years.

---

designers of computers were often one and the same. But as noted above, my primary focus here is on the field of high-speed digital computing.

Chapter 3
A System of Professional Societies: Negotiating the Sociotechnical Settlements

Professional societies often play important roles in the development of professional fields and academic disciplines. In fact, their activities often span and bridge the social and the technical – as well as the disciplinary and professional – such as in ongoing efforts to establish professional identities, define the scope of fields and subfields, codify relevant bodies of knowledge, set standards, and develop curricular recommendations. Professional society publications and activities can also help reveal major trends and issues in contexts that are otherwise difficult to access or assess, such as the private sector. The present chapter is largely focused on the internal development and relational interaction of three organizations that maintained interests in the computer field from the mid-1950s to mid-1960s, namely the Association for Computing Machinery (ACM), the Institute of Radio Engineers Professional Group on Electronic Computers (IRE PGEC), and the American Institute of Electrical Engineers Computing Devices Committee (AIEE CDC).

Yet unlike other historical and social studies of disciplines and professions, my focus on these particular groups stems not from their unambiguous association with a single, common professional or disciplinary domain. Rather, these organizations maintained partially overlapping interests – or "settlements" – in various domains of technology, bodies of knowledge, and types of work. I therefore frame these three organizations as constituting a dynamic "system of professional societies." Further, I document how a modicum of stability was achieved in this system through a long series of negotiations and compromises that were worked out both within and between these groups, often against a backdrop of rapid sociotechnical change.

I place particular emphasis on the role of the Joint Computer Conferences (JCCs) in this process. As discussed in the preceding chapter, by the early 1950s a growing band of electrical

engineers was staking out territory in the expanding computer field. The first JCC – which was held in 1951 – provides important evidence for this movement. In fact, the proceedings from this event include some of the first published uses of the term "computer engineer," revealing the emergence of a distinct professional identity for computer designers and related types of workers. Yet if the preceding analysis stressed the role of the joint conferences in the early and tentative emergence of "computer engineering" as a new field, the present chapter documents key shifts in the orientation and function of the JCCs through the 1950s and into the early 1960s. As I discuss below, the prominence of engineers in the early joint conferences gradually faded as these events became a common point of intersection for a more diverse assortment of actors and groups. In fact, the joint conference series and its associated joint committee both reflected and reinforced the respective sociotechnical settlements of the ACM, IRE PGEC, and AIEE CDC.

Even more generally, this chapter sheds light on the interplay of both stabilizing and destabilizing forces in the context of a system of professional societies. Potential sources of instability in this system include technological developments, changes in the size and scope of organizations, and incursions from "outside" groups. Sources of stability, on the other hand, include relative homogeneity in the composition of a given group, joint committees and other activities between groups, and the mirroring of various social and technical boundaries in diverse contexts, ranging from professional groups and the workplace to educational sites. Ultimately, I argue in this chapter that the joint conference series and its associated organizing committee helped ameliorate the persistent risk of instability in this system of professional societies.

The Early History of the ACM: "What Computers Do"

In the preceding chapter I largely sidestepped the early position and role of the ACM in the computer field. And while the history of the ACM has to some extent been covered elsewhere, an overview is necessary to frame the evolving relation of the ACM, IRE, and AIEE, as well as to set the stage for later historical developments. To begin with, the ACM deserves credit as the first stand-alone professional group in the computing field. Originally dubbed the "Eastern Association for Computing Machinery," the new group attracted 52 members to its first meeting at Columbia University in September of 1947 (Alt, 1962, p. 300). An early "Notice of Organization" outlined the organization's rather wide-ranging purpose: "to advance the science, development, construction, and application of the new machinery for computing, reasoning, and

other handling of information" (Alt, 1962, p. 305).[51] The broad scope of the ACM was also reflected in the composition of the group's first governing councils, which included many of the individuals introduced in the previous chapter, ranging from physicist and ENIAC co-developer John W. Mauchly to engineers such as Charles Concordia, T. K. Sharpless, and Jay Forrester. A number of computer-oriented mathematicians – such as Franz L. Alt, Hans Rademacher, and Mina Rees – also took on early leadership roles in the ACM.[52]

The group was renamed the "Association for Computing Machinery" in 1948 and it expanded rapidly thereafter, claiming roughly 250 members in April of 1948, more than 450 in early 1949, and more than 1100 in late 1951 ("News: Association for Computing Machinery," 1948; "News: Association for Computing Machinery," 1949; Alt, 1962, p. 301). Until the *Journal of the ACM* was established in 1954, *Mathematical Tables and Other Aids to Computation* served as the primary publication outlet for ACM news and articles.[53] Beginning in 1947, the group also organized its own national meetings and conferences, and it published proceedings for many of these events. In contrast to the more engineering- and industry-oriented AIEE and IRE, the ACM attracted relatively large numbers of computer-oriented mathematicians and scientists, and the group gained an early reputation for both its theoretical leanings and its reasonably close ties to the academic sphere.

In fact, the orientation of the Association toward science and comput*ing* – rather than engineers and comput*ers* – was evident in the proceedings for one of the group's national meetings in 1952. This particular volume included a Forward, authored by outgoing ACM President Franz L. Alt, that restated the group's purpose: "It is the purpose of the Association for

[51] A news item that appeared in *Mathematical Tables and Other Aids to Computation* included a statement of purpose for the ACM that was nearly identical to the one cited here, only differing in the replacement of the word "development" with "design" ("News: Association for Computing Machinery," 1948, p. 133). This modified version of the statement persisted well into the 1950s. The use of the word "science" in this statement also reflected the orientation and interests of many ACM members.

[52] Mathematician Mina Rees also served on the first ACM Council. Her early involvement with the computer field largely stemmed from her work at the Office of Naval Research from 1946 to 1953 (Green et al., 1998, p. 867). Rees' position on the ACM council suggests that the Association provided a more hospitable environment for computer-oriented mathematicians, including those that happened to be women. The AIEE and IRE, on the other hand, largely restricted membership to those who held degrees in engineering or physics. This only reinforced the homogeneity of these groups since few women had these qualifications.

[53] Founded in 1943, the scope of this journal steadily expanded from the mid-1940s on to include various topics related to high-speed calculating and computing machines. In addition to serving the early publication needs of the ACM, this journal also referenced and reviewed many of the papers published and events held by other professional societies, including the AIEE and the IRE.

Computing Machinery to advance the science of numerical computation, in particular the design, development, construction, and application of modern computing machinery" (1952).[54] As suggested by this passage, many in and around the field were beginning to frame large swaths of computing as a new type or branch of "science." And while the term "design" also appeared in Householder's remarks, there was no mention of "engineering."

In its early years the ACM enjoyed a relationship with the AIEE and IRE that was generally friendly and cooperative. By the early 1950s, however, new tensions surfaced, especially amid ongoing moves to define and clarify the orientation and jurisdiction of the major computer-oriented professional groups. As noted above, for example, in 1951 the ACM was officially listed as a "participant" in the first Joint Computer Conference (JCC), rather than as a full co-organizer. According to one more recent account, the first JCC was actually spearheaded by members of the AIEE, with the IRE accepting invitation as a joint sponsor (Armer et al., 1986, p. 226). The ACM declined a similar invitation, wishing instead to be listed as "cooperating." While larger political and financial motivations likely played a role in the ACM's guarded participation in the first JCC, more practical concerns were also afoot, including questions about the appropriate schedule and scope of the various national computer meetings. Yet as Alt explains, these types of issues were largely smoothed out by 1953, when "an unwritten compromise was worked out between JCC [Joint Computer Committee] and ACM, by which the latter would hold one national meeting per year, normally in summer … while JCC meetings would be held in the East in late fall and West in spring" (1962, p. 302).

The scheduling of conferences, however, was but one aspect of a more general process of professional and disciplinary negotiation that started to receive significant attention beginning in the early 1950s. In fact, it is somewhat ironic that Samuel B. Williams was in the middle of some of the earliest debates about the appropriate position of the ACM with respect to both the computer field generally and the computer-oriented groups of AIEE and IRE specifically. As noted in the preceding chapter, Williams was well-known for his involvement in the early design and construction of relay computers at Bell Labs. Yet he went on to serve as the ACM's Vice-President from 1950 and 1951 and President from 1952 to 1953.[55] As Akera describes it,

[54] Alt, like many other ACM leaders, held a Ph.D. in mathematics. He was also a co-founder of the ACM.
[55] As suggested by this biographical sketch, the term "hybrid actor" is appropriate for Williams. In fact, I identify a number of such actors in this chapter, and through their individual histories I bring into further relief how personal experiences and background frequently come into contact, mesh, and/or clash with

Williams "turned to the ACM as a way of retraining himself, particularly with respect to electronic computers and computing techniques" (1998, pp. 578-579). This move followed his retirement from Bell Labs in 1946 after 41 years of service, as well as a brief but generally unsuccessful stint at the University of Pennsylvania's Moore School in the mid-1940s ("Retirements: Samuel Byron Williams," 1946; Akera, 1998, pp. 578-579). Looking beyond his career trajectory, Williams' engineering background and previous experiences in computing were quite unlike those of prior ACM Presidents such as Alt, and Williams' loyalties were certainly tested as debates about the ACM's scope and proper position in the field rose in prominence.

As Akera explains, questions about the orientation of the ACM were surfacing by at least 1952, when a Policy and Planning Committee established by the ACM's governing council reported that areas of interest such as system requirements, logical design, and performance requirements should remain within the domain of the ACM (1998, p. 580). But an amendment proposed in 1953 pushed in a somewhat different direction as it called for the removal of the word "construction" from the ACM's constitution. According to Alt, this motion was intended to "reduce the overlap between the ACM and the purely engineering organizations, IRE and AIEE" (Alt, 1962, p. 305). As documented by Akera, much of the original impetus for this amendment came from IBM engineer and IRE-PGEC chair Morton Astrahan. After taking over as the editor of the IRE-PGEC's *Transactions,* Astrahan wrote a letter to Williams in which he declared, "We feel the major emphasis of ACM activities should be on the theory of computing and the applications of computing equipment in the scientific and commercial field" (cited in Akera, 1998, pp. 579-580). But in his response to Astrahan, Williams stood his ground: "I personally feel that the ACM has a very definite place in the engineering and scientific world" (cited in Akera, 1998, p. 580). In the end – and after much discussion and debate – the amendment was defeated by a slim margin of member votes.[56] At least for the time being, the term "construction" was to remain in the ACM's official statement of scope.

Yet despite both the Council's earlier recommendations and the failed constitutional amendment, from the mid-1950s onward Williams and other ACM leaders made their own

---

various technologies, social groups, institutions, and bodies of knowledge. Simply put, people frequently transcend and defy boundaries.

[56] According to Akera, this particular amendment was approved by the members of ACM's council. But as recounted by both Alt and Akera, it was ultimately defeated by ACM members (Alt, 1962, p. 305; Akera, 1998, p. 586).

strategic moves to distance the Association from the general sphere of computer hardware. In 1953, for instance, Williams authored a position piece that appeared in a new magazine titled *The Computing Machinery Field*.[57] In this short article, Williams noted that the AIEE, IRE, and ACM were all expressing "active interest in the field of computer machinery," and he asked whether the presence of these three groups was leading to overlap, duplication, and/or confusion (1953, p. 21). On the one hand, Williams explained that "[p]art of the overlap and duplication is beginning to be avoided through the activities of the Joint Computer Conference Committee" (p. 21). This was an important insight. As I note below, the joint committee and conferences were beginning to play an increasingly pivotal role in maintaining a balance between the ACM and its more engineering-oriented counterpart societies.

On the other hand, Williams also went on to argue that more could be done to partition the field between the three professional societies in question:

> But some more of the overlap and duplication may be avoided, by allocating portions of the field of computing machinery according to main interest. As between the ACM and the other two organizations, there is one area which is preeminently in the area of the ACM: "What Computers Do". This includes programming, logical design, problems to be solved, numerical and logical analysis of scientific and business problems, etc. (1953, p. 21).

Williams concluded his editorial by suggesting that AIEE and IRE publications were the primary outlets for "technical papers on machinery," and he noted that the division between these two groups would be worked out over time.[58] He also explained that ACM meetings and publications were largely focused on "what computers do." This allocation scheme clearly positioned the ACM as picking up where the more engineering- and machine-oriented IRE-PGEC and AIEE CDC left off.

Williams also made it clear that his 1953 article reflected only his personal views. However, he was soon addressing many of these same issues in his officially capacity as

[57] Established by ACM co-founder Edmund Berkeley in the early 1950s, this magazine took the title *Computers and Automation* in March of 1953. This is the name by which it is most widely known.

[58] Reflecting the dominant image associated with each organization, Williams explained that "the division [between the AIEE and IRE] will be worked out in much the same was as the division has been worked out in the past: electronic, high frequency, communication, to the IRE; electrical, low frequency, power, to the AIEE" (1953, p. 21). As suggested by this remark, the IRE seemed to have a bigger potential stake in the computer field, including in areas such as the design of electronic components and systems. In subsequent sections I discuss this point at length.

President of the ACM. In a speech that was delivered at the Association's 1953 national meeting and published the following year in the inaugural issue of the *Journal of the Association for Computing Machinery* (JACM), Williams started with a brief history of the "automatic computing field" that included a number of details about the formation of the Association (1954). He added:

> The Association has become an important factor in the field of computing machinery. Until the engineering societies became sufficiently interested to struggle with the "hardware", the Association provided a forum for all phases of the field. Now the Association can direct its efforts to the other phases of computing systems, such as numerical analysis, logical design, application and use, and last, but not least, to programming (p. 3).

And in another part of his talk, Williams continued to build on this theme by incorrectly stating that the AIEE and IRE computer committees were both established in 1951.[59]

As outlined above, the historical record reveals that electrical engineers in general – and the engineering societies in particular – were interested in computer hardware and other areas of computing by at least the mid-1940s. And while it is difficult to pin down the exact reasons for Williams' somewhat inaccurate account, by this time the ACM was clearly facing jurisdictional pressure from both the AIEE and IRE. In addition, Williams' framing opened the way for a more graceful exit to the debate, both by emphasizing the historical position of the ACM and by shifting the conversation toward a discipline-building agenda that centered on the ACM's presence in less contested areas, such as numerical analysis, applications, and programming. Yet boundary work is rarely so simple or clear-cut, and one of the topics identified by Williams – namely logical design – was an increasingly important area of negotiation in ongoing debates over the boundaries the various computing groups, a point to which I will return.[60]

---

[59] As noted in the previous chapter, a computing subcommittee was established by the AIEE in 1946, and elevated to full technical committee status in 1948. The IRE's computer committee was formed in 1948, and the PGEC in 1951. Yet even today, parts of the ACM web site include a subtitle that reads "The First Society in Computing" ("ACM: Association for," n.d.).

[60] The concept of "logical design" can be traced back to at least the work of Alan Turing in the 1930s. The term was also featured prominently in the title of a well-known 1946 report titled *Preliminary Discussion of the Logical Design of an Electronic Computing Instrument* (Burks, Goldstine, and von Neumann, 1989). As nicely summarized in a more recent volume, "The term 'logic design' refers to the process of specifying an interconnection of logic elements in digital computer hardware so that a desired

Subsequent leaders of the ACM revisted many of the issues that Williams had addressed. In late 1955, for instance, the Association's Presidency passed to Alston S. Householder, a mathematician who at the time was working for the Mathematics Panel at Oak Ridge National Laboratory. In a Presidential address delivered in 1955, Householder boasted that the membership of the group was approaching 2,000. Echoing Williams' prior remarks, he added: "Today there are active groups in the IRE and the AIEE concerned with componentry and construction, and the ACM is restricting its sphere to that of the effective use and application of those machines" (1956a, p. 1).[61] Yet in this same talk, Householder acknowledged that the field might be more aptly described as a spectrum rather than two distinct spheres:

> Design, construction, and use are but points on a continuous spectrum since clearly a designer must have a use in mind, and a user, if he is to be intelligent, must know something of the design. But ACM concerns now fall largely in the applications region of the spectrum (pp. 1-2).

And in 1956, Householder stated even more directly that there was general agreement that the ACM "should no longer concern itself with hardware" (1957, p. 2).[62] Householder's remarks hinted at the difficulties that came with bringing the activities, mission, and scope of a professional society into alignment with the major social and technical boundaries that were growing up in the computer field. Just how does one partition a "continuous spectrum," much less in ways that satisfy three professional groups? As I argue in this chapter, the concept of "sociotechnical settlement" helps us understand how this challenge was dealt with.

Further, by the late 1950s it was evident that the ACM and its members were increasingly aligned with mathematics and programming, theory and applications. The group also maintained close ties to the academic context, especially through its leadership ranks. These trends were reflected in the election of subsequent ACM Presidents, including mathematics professor and

---

function is performed" (McCluskey, 1976, p. 809). This general meaning of the phrase has remained roughly constant from the earliest days of the field to the present.

[61] Householder also indicated that engineers were extensively involved with the ACM in its early days, and he noted that "[i]n 1949 there was much discussion of componentry and design, less of techniques and applications" (1956, p. 1). By the time of his talk, however, the area of "techniques and applications" had risen dramatically in prominence and importance, both within and beyond the group.

[62] Householder raised another notable concern in this talk, namely the increasing mathematical orientation of the group: "I have heard the complaint that too many papers, in the meetings and in the Journal, are too mathematical, and that, in particular, there are not enough papers dealing with business applications" (1957, p. 2). Similar concerns have periodically resurfaced throughout the history of the ACM.

computer researcher John W. Carr III, who headed the organization from 1956 to 1958. Mathematician Richard Hamming, on the other hand, took the post from 1958 to 1960. In fact, Hamming emphasized in his inaugural Presidential address that the group had given up its interest in computing machinery, and was instead largely acting as a point of common ground for mathematicians, logicians, and users (Akera, 1998, p. 593). As documented by Akera, the ACM was also attracting large numbers of programmers to its ranks, and the group's membership rolls more than doubled from approximately 2,300 members in 1956 to more than 5,000 in 1959 (Alt, 1962, p. 301).

Perhaps not surprisingly, the ACM Conferences tilted accordingly. As reported in *Computers and Automation*, only about 15% (11 of 75) of the papers presented at the 1958 ACM National Meeting were focused on computer design, while 38% dealt primarily with computer mathematics, 20% with computer applications, and 19% with computer programming ("Is the Computer Field," 1958). As the editors noted, this data raised questions about whether the computer field "will stay together or come apart into pieces." On the other hand, evidence for the ACM's leanings toward mathematics and the sciences came in 1958, when the organization secured official representation in the Mathematical Sciences Division of the National Academy of Sciences – National Research Council (Alt, 1962, p. 304).

By some accounts, the ACM looked like a healthy and expanding organization as the 1950s drew to a close. However, it faced a growing roster of concerns. Paul Armer – who was defeated by Hamming in the group's 1958 Presidential run-off – identified many of these issues in an editorial published in early 1959. Complaining that the ACM was in "a state of complacent lethargy," Armer urged the leaders and members of the group to "think big" (Armer, 1959, p. 2-3). He also posited that the ACM might one day become "*the* professional society unifying all computer *users*" (p. 3, my emphasis). Armer went on to offer a number of more specific suggestions, including the establishment of a special interest group system and a change of venue for the ACM's national meetings from universities to hotels.[63] An ad-hoc committee led by ACM Vice President Harry Huskey made a number of similar recommendations in a 1959

[63] As Akera explains, the latter suggestion was an important challenge to the long-standing academic orientation of the group and its leadership (1958, p. 596). The idea was implemented in 1961, when the ACM's 16th annual national conference was held at a hotel in Los Angeles. This was also the first ACM meeting to feature manufacturer's exhibits, which was another significant change for an organization that had long resisted industrial or commercial influences (Huskey, 1961a).

report, including the special interest group idea. As described by Akera, this report recommended many organizational and representational changes, especially in light of the group's ongoing movement beyond "scientific computing" and into the realm of business data processing (Akera, 1998, pp. 597-598).

But even as the wheels of organizational change were starting to turn, outspoken ACM members continued to complain about the group's scope and orientation. Philip R. Bagley of MIT's Lincoln Laboratory noted in 1959, for example, that "[i]t is not at all clear to me what the ACM's actual sphere of interest is. If one were to judge from the *Journal*, it appears to be largely in mathematical techniques suitable for computers, and includes a smattering of programming techniques" (Bagley, 1959). Pointing to a disconnect between the ACM's constitution and its actual activities and publications, Bagley went on to note that many areas of possible interest to the ACM were "being staked out by other societies, principally SIAM, AIEE, and IRE-PGEC."[64] He called on the ACM to clarify its interests, and to carefully consider how these interests overlapped with other, "adjoining" societies.

Questions about the scope of the ACM were also paralleled by a sort of identity crisis among many of the group's members. Early evidence for this theme can be found in *Communications of the ACM* (*CACM*) a monthly publication that was established in 1958 as an outlet for news, notices, letters, and other materials not suitable for the more technical *Journal of the ACM*. In a letter that appeared in one early issue of *Communications*, representatives of the ACM's Los Angeles chapter asked: "What is your reply when someone asks your profession? Computing Engineer? Numerical Analyst? Data Processing Specialist?" (Editors of DATA-LINK, p. 6).[65] Noting a lack of suitable alternatives, the authors added:

> It would help our profession to be widely recognized if it had a brief, definitive, and distinctive name. This should be general enough to cover a variety of subfields – from numerical analysis to data processing, but specific enough to

[64] The Society for Industrial and Applied Mathematics (SIAM) was formally established in 1952. While largely beyond the scope of my analysis, "Looking Back, Looking Ahead: A Siam History" (2002) provides a summary overview of this organization's history.

[65] It is worth noting the intentional use of the phrase comput*ing* engineer rather than comput*er* engineer. The former suggests a concern with applications and hence computing, while the latter implies an individual who designs or engineers computers. In the following chapter I document the use of these terms as occupational designations.

> imply that computing applications are involved. Consider the solid professional sound of such terms as "Petroleum Engineer" or "Nuclear Physicist."

In addition to revealing a perceived lack of disciplinary identity and unity among many computer-oriented workers, this letter also reflected concerns about "professional" recognition. Further, the authors hinted that other domains, such as engineering or the sciences, might provide inspiration in their quest for a suitable professional identity.

Responding to this letter, ACM members put forward a number of creative suggestions. One letter defined the obvious yet awkward term "comptology" as "[t]he science of computers, computation, and computer control. Also of computer application" (Correll, 1958). The author also included a number of more specific variations of the term, including "electrical engineering comptologist." And a subsequent writer, noting inspiration from the Greek *hypologi* ("to compute") suggested "'hypologist' for the man and 'hypology' for the field," (Zaphyr, 1959). And while these terms never came into widespread use, the underlying issues were clearly important. In fact, below I discuss the emergence of some other titles that ultimately proved more successful.

In the early 1960s the ACM continued a general pattern of growth and expansion, and debates over the identity and scope of the group temporarily took a backseat to other pressing matters. In fact, many of the issues addressed around this time were at least partially skewed toward the interests and agenda of ACM President Harry D. Huskey. After serving as Vice President under Hamming, Huskey took over the ACM's top spot from mid-1960 to mid-1962. In terms of background, Huskey followed in the footsteps of other "hybrid" actors, such as the aforementioned Samuel Williams. He held M.S. and Ph.D. degrees in mathematics – an appropriate pedigree for an ACM leader (Lee, 1995, pp. 390-391). Yet Huskey was also well-known for his early work and many contributions in the area of computer design, and he maintained close ties with the IRE-PGEC, having served as Review Editor for the group's *Transactions* from 1953 to 1957.[66] Even Huskey's joint appointment at Berkeley – in 1954 he took the title Professor of Mathematics and Electrical Engineering – reflected his somewhat ambiguous position with regard to the computer field's major boundaries (Huskey, 1991, p. 294).

[66] Huskey was involved with the ENIAC project from 1943 to 1946, and he worked on the early logical design of the EDVAC. He also played a leading role in the design and construction of the well-known Standards Western Automatic Computer (SWAC) in the late 1940s and early 1950s, and he was the principal designer of the commercially produced Bendix G-15 computer (Lee, 1995, pp. 390-391).

Huskey's frequent "Letters from the President" column reveals that the early 1960s were a time when the leaders of the ACM were dealing with new publications and publication policies, new institutional membership and student chapter programs, and the formation of the American Federation of Information Processing Societies (AFIPS). The group's membership also continued to expand, with the membership count breaking through 10,000 barrier around 1962 (Huskey, 1962a). But perhaps just as importantly, changes to the ACM bylaws that were passed in 1960 opened the way for the official formation of Special Interest Committees (SICs) and Special Interest Groups (SIGs) (Huskey, 1960a; Gilchrist, 1961a). The former were smaller and more exploratory in nature, while the latter required a larger membership and were viewed as "miniature professional societies" that operated under the auspices of the ACM.[67]

As other groups such as the IRE had demonstrated, the SIG strategy could better accommodate rapid growth in the size and scope of a professional organization, especially by allowing various special interests to segment, but not secede. In fact, for the ACM the success of this structural change was reflected in the rapid establishment groups and committees. By late 1964, three SICs were active in the areas of Computer Installation Management, Computer Languages, and Digital Computer Programmer Training (Forsythe, 1964b). Five SIGs were also established by this time, including SIGBDP (Business Data Processing), SIGIR (Information Retrieval), SIGMAP (Mathematical Programming), SIGUCC (University Computer Centers), and SIGBIO (Digital Computing in Medicine).

The titles of these groups also suggest that the ACM's sociotechnical settlement in the areas of computer applications and programming – or "What Computers Do" – was reasonably well-established. The existence of the SIGUCC, on the other hand, hinted at the group's continued ties to the academic sphere. Additional evidence for these trends can be found in membership surveys that were conducted in the early 1960s. Data collected in 1961, for instance, revealed that 85.5% of members were primarily interested in "programming and using computers," while just 12.6% expressed a major interest in the "design of computers" (Gilchrist, 1961b).[68] As ACM Secretary Bruce Gilchrist concluded, "[T]he major interest of present ACM

[67] The phrase "miniature professional societies" was used in a 1961 call for members from the ACM SIG for Mathematical Programming ("ACM Special Interest Group," 1961).

[68] A follow-up survey that was conducted in 1962 provided a more detailed breakdown of these numbers (Gilchrist, 1962). It indicated that 13.7% of members were primarily interested in design, followed by 18.6% in "Systems Programming" and 37.2% in "Applications Programming." Those interested in

members is very definitely the programming and use of computers, rather than the construction and design of computers." Yet as suggested by the preceding overview, Gilchrist's remarks are not entirely surprising, as they reflected pre-existing trends that were rooted rather deeply in the history of the ACM. [69] In subsequent sections I follow these trends into the 1960s, when the ACM and many of its members became increasingly engaged with educational issues and aligned with the emergent field of "computer science." Before doing so, however, it is necessary to review the history of the IRE-PGEC and AIEE CDC from the mid-1950s to early-1960s. As I discuss in the following sections, these organizations evolved in tandem with both one another and the ACM, and an overall stability of this system of professional societies was maintained.

IRE-PGEC: The Voice of the Computer Engineering Profession

As discussed in the preceding chapter, the IRE's Professional Group on Electronic Computers (IRE-PGEC) was established in 1951, and both the group's size and range of activities ramped up quickly thereafter. In fact, by 1954 the membership of the PGEC had swelled to over 2500 members, making it the IRE's largest professional group ("News," 1954). But just who were these members? A series of surveys provide important evidence regarding the make-up of the IRE-PGEC in the mid-1950 to early-1960 period. More specifically, this data reveals the extent to which the group was largely composed of individuals who held engineering degrees and were employed in the private sector. A 1956 survey indicated, for instance, that a vast majority of the more than 2500 respondents were affiliated with private industry in either the commercial (40%) or military (37%) sectors, and 54% of all members noted that they were involved with computers as "producers" (Martin and Olson, 1957, p. 49). With regard to educational background, a vast majority of this same pool of members held engineering degrees, and a question about the "nature of work most actively engaged in" revealed three leading responses: electronic design, technical management, and logical design (p. 54). The work area labeled "programming" followed a distant fourth.

"Installations" and "Marketing" respectively made up 7.6% and 6.1% of all respondents, and "Education" was a major interest for another 2.9% of members. The remaining 13.9% of members indicated "General" or did not respond.

[69] Gilchrist also noted that "19 per cent [sic] of ACM members reporting ten years or more of experience in the computing field must mean, even allowing for reporting errors, that the Association for Computing Machinery numbers among its members a very high percentage of the people who were in the computing field in its early days" (Gilchrist, 1961b). Such remarks clearly emphasized both the historical and contemporary importance of the group.

On the other hand, individuals with interests in education, computer applications, and programming were clearly in the minority of the group's members. Only 22% of those surveyed in 1956 claimed that their primary involvement with the field was as computer "users," and approximately 18% indicated that they were actively engaged in the "applications" phase of the computer field (pp. 49; 54). Further, roughly 15% of respondents indicated "programming or mathematics" as their primary work activity (p. 54). In terms of employment, only 12% indicated that they worked for an educational institution or research group, and a mere 67 individual respondents (out of more than 2500 total) identified themselves as "educators" (p. 49) And finally, the survey revealed the extent to which IRE membership overlapped with other organizations, with about 23% and 15% of those surveyed also claiming membership in the AIEE and ACM, respectively (p. 54).[70]

These data show that professional engineers with interests in computer systems, electronic design, and related subjects largely filled out the ranks of the IRE-PGEC. A second membership survey that was conducted in 1958 reinforced the earlier findings, albeit with a larger sample size (Uncapher, 1959). One data point worthy of note centers on a revised "nature of work" question where large numbers of respondents classified their primary work activities as most closely related to "engineering" (about 65% of responding members) and "research" (about 25% of members) (p. 61).[71] Conversely, "programming/math" and "education" were respectively selected by about 6% and 5% of respondents (p. 61).

In 1959, IRE-PGEC chairman Willis Ware boasted that the group's membership had topped 7000 (Ware, 1959, p. 90). Yet even in light of this impressive growth, a survey conducted the following year suggested that the overall composition of the group was changing very little (Uncapher, 1961). Based on nearly 4000 responses, the collected data indicated that employment of members in the private sector had risen slightly to 86%, including 44% and 42% in the defense and commercial sectors, respectively (p. 84). In addition, 58% of respondents specified that their work was primarily in "engineering," while another 15% indicated "research" and 10%

[70] Perhaps even more suggestively, those respondents who indicated that they "usually" attended the ACM National and AIEE National meetings numbered a mere 7 and 18 individual members, respectively. However, approximately 34% of all responding members regularly went to the IRE National Meeting, 26% to the Eastern Joint Computer Conference (EJCC), and 11% to the Western Joint Computer Conference (WJCC) (Martin and Olson, 1957, p. 55).

[71] These percentage values are my own approximations, based on published bar graphs of the 1958 survey data. Unfortunately, this summary of the survey did not include numerical totals or percentages for most of the results.

selected “administration” (p. 83). Conversely, just 3% indicated that their primary work activities were in the area of “education,” while 4% opted for “programming/math” (p. 83) Those employed by educational institutions jumped to 7%, although this clearly still represented a rather small segment of the group (p. 84).

In the 1950s the PGEC was repeatedly framed as the organization of choice for those whose interests centered on the “the theory and practice of computer engineering” and “the allied arts and sciences” (“Constitution,” 1955). Yet the group also developed additional strategies and policies to help bolster its position in the field. An “affiliate” plan was proposed, for instance, that allowed an interested individual to become a member of an IRE professional group without first having to join the IRE, although they did need to belong to an “accredited organization approved by that group and the IRE Executive Committee” (Baker, 1957). As documented in a historical retrospective authored by former PGEC chair Walter Anderson, this idea emerged in the mid-1950s when it was increasingly clear that

> some of the best logic designers in the computer field were physicists who normally would not participate in IRE publications. The Computer Group wanted to relate to such companion professionals as these and to the mathematicians engaged in programming (Anderson, 1976, p. 48).

While the idea of affiliate membership initially received a lukewarm reception when pitched to the top leadership of the IRE, persistent lobbying helped lead to the implementation of the program in 1957.[72] And by 1960, a total of 15 societies had been approved as affiliates, including the AIEE and ACM (“Affiliate Status,” 1960). This was an important change, as it enabled greater potential participation in the PGEC by those who might not otherwise qualify for IRE membership, including many individuals who did not hold engineering degrees.[73]

---

[72] As Anderson recounts, when PGEC chair Harry Larson pitched the affiliate member concept to the IRE’s Groups Committee “it found little support from a roomful of men 10 to 20 years his senior, who gravely explained the dire effect this would have on the IRE and solemnly questioned the value of the concept” (1976, p. 48). As suggested by these remarks, professional societies often maintain membership gate-keeping functions that are difficult to change.

[73] According to Ryder and Fink (1984), the IRE was historically more liberal than the AIEE with respect to membership requirements (pp. 214-215). However, even the IRE restricted access to voting member grades to graduates of “schools of recognized standing,” while a non-voting “Associate” grade was reserved for those who had an interest – but not the appropriate educational credentials – in radio engineering. The ACM, on the other hand, had an early history of open membership, but this ended in 1965 when the group made four-year degrees mandatory for new members (Ensmenger, 2001, p. 63).

And as the winds of change started to sweep through the ACM in the late 1950s and early 1960s, so too did the IRE-PGEC enter its own phase of reflection and evaluation. In fact, by the late 1950s numerous discussions about the scope and position of the group were appearing in publications such as the group's *Transactions*. In a 1959 editorial, for instance, PGEC chair Willis Ware explained:

> The PGEC also needs to review its domestic position; with other PG's [Professional Groups] critically reviewing their areas of interest in view of technological advances and the opening of new fields, it is time for the PGEC to introspect and determine which position it wants to have in the U.S. computing fraternity and to move in that direction. It might even be desirable to consider modifying the name of the group (Ware, 1959, p. 91).

Ware's remark revealed the extent to which the scope and position of the PGEC required two types of jurisdictional negotiations, one centering on other Professional Groups within the IRE, and the other involving various organizations outside of the IRE, such as the ACM. The rapid "technological advances" referenced by Ware only further complicated these processes.

In light of these challenges, the IRE PGEC worked to both secure and expand its sociotechnical settlement. In 1961, for instance, PGEC chairman Arnold A. Cohen followed in Ware's footsteps by suggesting that the name of the group might be changed to the Professional Group on Information Processing Systems, or PGIPS (Cohen, 1961). As Cohen explained, this new title carried a broader connotation and included the important word "systems." He added:

> The combination gives recognition to the long established fact that our attention is not confined to components and techniques internal to computers. Further, it is certainly our responsibility, whether as PGEC or as PGIPS, to serve the increasing interest in system engineering of computer-centered systems (p. 845).

As suggested by Cohen's remark, it was increasingly evident that computers were being used with increasing frequency as components in larger technological systems. Hence, explicitly recognizing this new area of activity and claiming it looked like a sound strategy for the PGEC, although Cohen's letter provided little in the way of additional details about this expansionist agenda.

On the other hand, around this same time Cohen and other PGEC leaders were working on a suggestive update of the group's official statement of scope. Through

September of 1961, a statement that appeared in each issue of the *Transactions on Electronic Computers* declared that the scope of the journal "includes the design, theory, and practice of electronic computers and data-processing machines, digital and analog, and parts of certain related disciplines such as switching theory and pulse circuits" ("Information for Authors," 1961a). But beginning in December of 1961, this relatively simple declaration was replaced by a new statement of scope that identified and described five subject areas in substantial detail:

a) all aspects of design, theory and practice relating to systems for digital and analog communication and information processing;
b) components and circuits for digital and analog systems, including techniques for accomplishing the functions of logic, arithmetic, storage, control, mass data storage, input, output, and external communication in such systems;
c) relevant portions of supporting disciplines, including switching theory, symbolic logic, numerical methods, codes and number representation systems, abstract machine or automation theory, symbolic logic, bio-sciences, machine learning, pattern recognition, and other extensions of logical machine capabilities;
d) production, testing, operation, and reliability of digital and analog systems; and
e) those aspects of application, use, and programming of digital and analog computing devices and information systems that relate to their design and operation ("Information for Authors," 1961b).

While this passage was ostensibly framed as the scope of a journal, to some extent it clarified the PGEC's settlement in the computer field. In fact, and as I note below, much of this statement was incorporated into one of the group's later constitutions. This passage is also striking in that it identified and intertwined many different bodies of knowledge and types of technology, especially through the use of terms such as design, theory, practice, components, circuits, systems, and techniques. As a part of the IRE-PGEC's ongoing efforts to "critically review" its position, this statement of scope spelled out the group's settlement in rather extensive detail.

A number of additional points are worth noting with regard to this passage. First, the use of terms and phrases such as "systems," "information processing," and "information systems" reflected Cohen's prior remarks about increasing the group's presence in the area of "systems" and perhaps even renaming the PGEC accordingly. And second, some of the subject areas

outlined in this statement were clearly in overlapping areas of interest, even if many of the "core" areas had long been viewed as the province of computer designers and engineers. Section (c), for instance, acknowledged some of the areas where the settlement of the PGEC tended to overlap with other fields.

Yet the use of the phrase "supporting disciplines" rather than "related disciplines" in this statement of scope framed these other (and unidentified) fields as secondary – and perhaps even subservient – to the PGEC's major area of settlement. Further, describing this wide range of special interest areas as "extensions of logical machine capabilities" clearly emphasized the importance of the "machine," even in those phases of computing that were more application- or user-oriented. And on another closely related note, this new statement reiterated the assertion – which was first made in the early 1950s – that the areas of "application, use, and programming" were of particular relevance to the PGEC and its members when they impinged on machine design and operation.[74] Of course, this point remained significantly open to interpretation, especially given that the boundaries around design and use are rarely so clear-cut.

In the midst of ongoing efforts to clarify the scope of the PGEC, chairman Cohen also indicated that the group was considering the establishment of new "Technical Activities Committees" (or "TACs") as another way for the group to focus on special-interest topic areas. As Cohen explained, these committees might spearhead various types of activities, such as planning symposia, organizing conference sessions, reviewing papers, and cooperating with other IRE technical committees (Cohen, 1961). He added that each TAC would function as a "vigorous steering committee for organizing technical activities," especially in light of the ongoing growth of "specialties within specialties." By April of 1962, it was announced that the first two TACs would be dedicated to "analog and hybrid computing" and "logic and switching theory" (Cohen, 1962b). These developments were something of a throwback to the late 1940s, when the IRE's Technical Committee on Electronic Computers was first established. And while the IRE's technical committee and associated subcommittees gradually faded in the 1950s as professional group structure rose in prominence, this revival of the committee structure in the

[74] As indicated in the previous chapter, a 1953 editorial in the PGEC's *Transactions on Electronic Computers* explained that "[w]e may think of programming as relating to applications and being outside the sphere of interest of most computer engineers" ("Editorial," 1953). However, in light of the fact that the same journal issue included a paper that discussed a particular programming technique, the authors qualified that this topic "should be of concern to the engineer because such programs offer an alternative to designing auxiliary equipment for the same purpose."

early 1960s revealed how the PGEC was looking for new ways to cope with its own expanding size and settlement.

But despite the PGEC's increasingly expansive settlement, other prominent spokespersons reaffirmed the group's traditional identity. In a 1961 editorial, for instance, University of Michigan electrical engineer and incoming PGEC *Transactions* editor Norman R. Scott noted that the expanding page count of the group's flagship journal was "evidence not only of the growth of the computer engineering profession but also of the growth of the PGEC as a voice of that profession" (Scott, 1961). Membership rolls were also expanding, and by 1960 the group boasted about 9000 members, bringing into rough parity with the 8900 members that the ACM claimed the following year (Anderson, 1976, p. 49; Huskey, 1961b). Yet whether stability could be maintained between the ACM and PGEC – or the "computer users" society and the "computer engineering organization" – remained an open question, especially into the 1960s. Further, the PGEC was not the only engineering organization that had a stake in the computer field. I now turn to the remaining organization in this triad of professional societies.

The AIEE CDC: Committee-Bound and Power Industry-Oriented

As indicated in the preceding chapter, the larger and more influential IRE-PGEC casts a rather long shadow over the history of the AIEE's Computing Devices Committee (CDC). This tendency was only exacerbated in the 1950s by the persistent orientation of the AIEE toward power engineering, as well as the concomitant tendency of the group to focus much of its activity on the analog and application aspects of computing. Yet it is worth reviewing the history of the CDC from the mid-1950s onward, both in the interest of rounding out this system of professional systems and in order to set the stage for other developments.

To begin with, through much of the 1950s the CDC continued many of the activities that the group had initiated in the late 1940s and early 1950s. For example, CDC committee members were responsible for contributing short summaries of progress in computing devices for an annual "engineering developments" feature article, which was published each January in the AIEE's widely-read *Electrical Engineering* magazine. The committee also continued to review papers for AIEE publications and organize panels for AIEE conferences. Yet by 1954, an annual report summarized that "much of the committee's effort has been exercised through its participation in the Joint Computer Conferences" ("Report of the Board," 1954). In fact, this

same report explained that the joint conferences "provide a much needed forum for concentrated discussion of particular phases of computer activity and have the tremendous advantage of concerted action on the part of AIEE, IRE, and ACM rather than dispersing this activity in several places" (p. 774). Below I discuss in more detail the pivotal role of the JCC in maintaining stability in this system of professional societies. However, this passage clearly hinted at the perceived importance of the joint conferences and its associated committee in uniting the diverse phases of the field. In addition, the "Institute Activities" section of *Electrical Engineering* regularly covered the JCCs, providing further evidence for the perceived importance of these events, even for the AIEE writ large.

The composition and direction of the CDC remained largely consistent in the mid-1950s. In fact, many of the leaders of the group had much in common with their predecessors. Frank Maginniss, chairman of the group from 1953-1955, was as an electrical engineer in General Electric's Analytic Engineering Department ("AIEE Officers," 1954, p. 852). Like his General Electric colleague Charles Concordia – who was the first chair of the AIEE's original subcommittee on Large-scale Computing Devices – one of Maginniss' main areas of interest centered on using computers to solve engineering problems that were relevant to the electrical utilities ("New Attendance Record," 1956, p. 1111). Edwin L. Harder, who chaired from 1955 to 1957, was similarly an engineer in the "Analytical Section" of Westinghouse Electric. Harder's expertise in and orientation toward analog computing was reflected in his earlier position as chair of the CDC's Analog Computer Subcommittee, as well as his leading role in the design and construction of the well-known ANACOM analog computer in the late 1940s (Aspray, 1993). And while Harder gained familiarity with the use of digital computers through his position at Westinghouse, his expertise in the area of computer design was limited to the analog domain.[75]

Under the leadership of Maginniss, the number of CDC members hovered around 30, and the activities of the group were clustered around six active subcommittees.[76] Harder's tenure as chair, however, was accompanied by a noticeable expansion of the CDC's member rolls. By

[75] In 1991, Harder recounted his early involvement with the AIEE CDC. As he explained, "[t]here was a digital subcommittee and an analog subcommittee all in the Computer Committee. My part with it was analog at first. And as Westinghouse never really did build digital computers, why, I remained a user all my life" (Harder, 1991).

[76] A total of 31 members were listed on the committee's official roster in 1954, and 30 members were listed in 1955 ("AIEE Officers," 1954, p. 852; "AIEE Officers," 1955, p. 846). The subcommittees during this time included the Digital Computers, Analog Computers, Computer Bibliography, Digital Computer Comparisons, Analog-Digital Converters, and West Coast ("AIEE Officers," 1954, p. 852).

1956 the membership roster stood at 42, and in 1957 it listed 58 affiliated individuals ("AIEE Officers," 1956, p. 851; "AIEE Officers," 1957, p. 844). Through this same time period the CDC also appeared poised to both lead an expanded array of activities and assume a more prominent position in the AIEE. An annual report published in 1956, for instance, indicated that the CDC was reviewing its organizational structure in light of "rapid expansion in the computing devices field" ("Report of the Board," 1956, p. 752). And a 1957 committee report noted that the increasingly diverse activities of the CDC might eventually lead to the formation of an entirely new AIEE division dedicated to computing devices ("Report of the Board," 1957, p. 737).

While this was certainly an ambitious and forward-looking proposition, the group's expansion remained closely linked to member interests in computer application and use. In a 1955 report, for instance, it was noted that the CDC "may have an important part to play in a combined educational and application function to aid the industry in bringing into play the rapidly increasing power of the digital computer" ("Report of the Board," 1955, p. 730). It was reasonably clear that the use of the phrase "the industry" in this passage primarily referred to the power industry. In fact, this same report indicated plans for collaborative activities with the AIEE Committee on System Engineering, with particular emphasis on surveying how electric utilities were using digital computers to handle accounting and other business problems. AIEE interests in the *use* of computers received more formal recognition in 1956, when an Applications Subcommittee was added to the CDC. In addition to serving as a liaison with other AIEE committees with interests in computer applications, the scope of this subcommittee centered on the "treatment of all phases of the application of computers in which the dominant factors are the design, construction, selection, installation, and operation of computing and related devices" ("Report of the Board," 1956, p. 752).[77] The increasing relevance of computing and data processing in the AIEE was also reflected in the establishment of a "Joint Division Committee on Automation and Data Processing." In 1957 this committee consisted of 21

[77] As this statement suggests, the involvement of many electrical engineers in the area of computer applications remained significantly oriented toward the physical "hardware" of computing. That is, even if not directly interested in machine design, engineers might be called upon to specify, select, and install computer equipment, perhaps even as part of a larger technological system. By contrast, organizations such as the ACM were developing a reputation for their "top-down" orientation toward theory, languages, algorithms, and applications. These two very different ways of looking at computers and computing set the stage for new conflicts of culture and interest among the major sociotechnical factions of the field.

members, including the outgoing and incoming chairmen of the AIEE CDC ("AIEE Officers," 1957, p. 845).

While serving as head of the CDC Harder authored a two-part article on "The Computing Revolution" that was published in *Electrical Engineering* in 1957 (Harder, 1957a; 1957b). Harder's focus in this piece tended toward applications, as suggested by his leading remark that "[c]omputing progress in electrical engineering is an integral part of a revolution in information processing" (1957a, p. 476). And while the author reviewed some major "Advances in Machines," Harding devoted much of the article to reviewing the state of the programming art and surveying how computers were being used in research, engineering, business, and manufacturing. Other papers, conference panels, and news items revealed that interest in computing among AIEE members often clustered around two more specific areas of engineering application, namely aeronautics and power systems. Both of these areas had high demands for computational power, especially for design and simulation. In fact, the organization of an AIEE Power Industry Computer Application Conference in Toronto in 1958 revealed the rapidly expending use of computers in even this relatively conservative and old-guard province of electrical engineering ("AIEE Power Industry," 1958). In fact, the conference program indicated that many of the conference papers discussed how computers were being used to design power machinery, and to analyze and simulate power networks (p. 848).

The CDC continued a modest pattern of growth as it entered the late-1950s, and by 1958 the committee boasted 65 members ("AIEE Officers," 1958, p. 882). Even more importantly, from 1957 to 1959 the chairmanship of the CDC was taken over by Morris Rubinoff, whose background was quite unlike that of Maginniss and Harder. Affiliated at the time with both the University of Pennsylvania's Moore School and Philco Corporation, Rubinoff held academic credentials in mathematics and physics, and he had worked with Aiken at Harvard on the Mark series of computers ("Alumni: Obituaries," 2004; Rubinoff, 1971, pp. 4-6). From 1948 to 1950 he was also involved with computer design work at Princeton's Institute for Advanced Studies (Rubinoff, 1971).

Evidence suggests that Rubinoff's particular interests inflected his agenda while serving as chair of the CDC. As retrospectively noted by Willis Ware, for instance, Rubinoff was an "upstart" who was trying to change the direction of CDC, especially in terms of shifting its emphasis from analog to digital computers (Ware, 2005). In 1959 – at the end of his tenure as

CDC chair – Rubinoff similarly explained that "[t]he primary objective of the AIEE in the last few years has been to do that job of education" ("Is it Overhaul," 1959a, p. 30). Describing the success of these efforts, Rubinoff added that "in the power industries, for example, there has been a big increase in the use of digital computers in the last few years. This is because we in the AIEE went hammers and tongs at the problem of educating the power engineers to the use of computers" (p. 30). Given the early prevalence of analog computing devices in the power industry, promoting digital computers was likely one of Rubinoff's major goals as CDC chair.

Even more generally, the AIEE fell into a pattern of slowed growth in the mid and late 1950s, and by 1957 the total IRE membership for the first time surpassed that of the AIEE (Ryder and Fink, 1984, pp. 215-216). The leaders of the AIEE were increasingly concerned about the future of their Institute, and a special task force was convened in 1957 to evaluate the state of the organization and its objectives (pp. 219-220). This group concluded that the AIEE had largely failed to enter new fields, did not have sufficient appeal to student members, and did not adequately cover the whole field of electrical engineering (pp. 219-220). The task force also critiqued both the AIEE board and its technical committee structure for perpetuating these problems, and it recommended the establishment of national-level "Institute Technical Groups" (ITG). These were to be similar in form and purpose to the IRE's SIGs (p. 220).

Around the time of ITG proposal, a growing body of evidence also revealed the extent to which AIEE members maintained some level of interest in many different phases of electrical engineering, including computers and computing. For instance, in 1959 Rubinoff claimed that, of the AIEE's roughly 50,000 members, "some two or three thousand have indicated that their primary or secondary interest is in computers" ("Is it Overhal," 1959a, p. 30). Yet Rubinoff indicated that that in light of these statistics, there remained "47,000 engineers who could use computers and who should be made aware of computers if only someone would only take the trouble to do it" (p. 30). Rubinoff's comment once more revealed the extent to which the leaders of the CDC viewed the activities and future growth of their committee as linked to the use and application of computers rather than computer engineering and design.

Rubinoff's rough statistical estimates were likely based on survey and reader response data that was collected and published in the late 1950s and early 1960s. A survey conducted in 1960 and published in 1961, for example, revealed that only about 3.6% (1,608 of 44,308) of responding AIEE members indicated that their "primary interest" was in the area of computing

devices, while about 4% (1,555 of 38,515) noted a "secondary interest" ("Progress of Institute," 1961, p. 373). On the other hand, this same survey revealed the persistent dominance of the Power Division, which captured an impressive 36% of all primary and almost 38% of secondary member interests. Other significant areas of interest included electronics (4.6% primary and 6.1% secondary). In addition, various Industrial Division sub-fields, including Feedback Control Systems, Industrial and Commercial Power Systems, and Industrial Control received reasonably high response rates.

This same survey also reported on member preferences with regard to technical groups ("Progress of the Institute," 1961, p. 373). For starters, about 5% (or 114 of 2298) of responding members expressed an explicit interest in the formation of a "Computing Devices" technical group. By contrast, more than 10% of respondents called for the formation of an ITG dedicated to "Power Transmission and Distribution," and sizable numbers of members also recommended the creation of groups in areas such as Electronics (6.2%), Power Generation (5.1%), and Basic Sciences (4.5%). Other popular ITG proposals included Data Communication and Feedback Control Systems, which garnered about 5% each. In light of these results, by September of 1961 the new ITG program was being built up around eleven proposed technical groups ("Which Institute Technical Groups," 1961, p. 704). One of the ITGs was dedicated to "Computing Devices," and was situated in a Science and Electronics Division along with the Basic Sciences and Electronics groups. As suggested by this development, computers and computing were well-established areas interest for many AIEE members. In fact, the prospects for a larger and more vibrant computer-oriented technical group was likely encouraging for those who had worked hard in the 1950s to maintain and expand the scope and activities of the CDC. On the other hand, computing was still a somewhat peripheral extension of the AIEE, especially in light of the continued dominance of power and electric industry interests in the organization writ large.

But even more importantly, the initial momentum of the ITG proposal was quickly subverted as high-level moves toward an AIEE-IRE merger gained traction. By the time that the merger announcement and resolution appeared in *Electrical Engineering* in December of 1961, the technical groups plan was essentially obsolete (Chase, 1961). As many surely recognized at the time, grafting technical groups onto the AIEE was a good idea that had come too late, both for the organization generally and for those members who maintained significant interests in the area of computing devices. As summarized by Ware, Rubinoff and his successors "gradually got

things turned around a little bit, but it never got turned very markedly before the merger of AIEE and IRE" (Ware, 2005).

The AIEE Winter General Meeting in 1961 featured an impressive roster of thirty seven computer papers in seven sessions (Kagan, 1961). And in 1963, an ad-hoc subcommittee of the CDC produced a paper that outlined recent advances in the computer field. In addition to being presented at the 1963 Winter General Meeting of the newly-formed Institute of Electrical and Electronics Engineers (IEEE), this paper appeared in near-verbatim form in both *Electrical Engineering* and *Computers and Automation* (Ad Hoc Group, 1963a; 1963b). Yet these activities were clearly something of a last hurrah, as the final two CDC chairs – namely Ruben Imm (1961 to 1963) and Claude Kagan (1963 to 1964) – were primarily focused on representing the interests of the CDC in the merger of the IRE and AIEE and in the formation of the American Federation of Information Processing Societies (AFIPS).

Merger, Identity, and Scope: Forming the IEEE Computer Group

As noted above and documented by historian A. Michal McMahon, various factors contributed to persistent tensions between the IRE and AIEE in the 1940s and 1950s, including the ebb and flow of membership numbers and questions about the scope of each group's activities.[78] Yet in the 1950s, the two groups started to grow closer, including through a Joint AIEE-IRE Coordination Committee that was established in 1952 (McMahon, 1984, p. 240). Joint activities in areas such as student groups and standardization were also increasingly common through the 1950s, and in 1956 a reciprocal AIEE-IRE membership plan was established (IEEE Center, 1984). The groups continued to move closer until the AIEE and IRE Boards agreed to work toward a merger in 1961 (McMahon, 1984, p. 241). Of course, there remained countless details to hash out, in areas ranging from publication outlets and the format of conferences to the appropriate geographic scope of the organization (i.e. American or International). Further, melding the AIEE's technical committee structure with the IRE's professional group system loomed as a particularly large challenge as the merger progressed.

Ultimately, a professional group structure – which was such a pivotally important ingredient in the IRE's post-war growth and vitality, and which also inspired the AIEE to

[78] My historical overview of the merger is largely a summary of the accounts developed by McMahon (1984, pp. 239-243) and Ryder and Fink (1984, Ch. 12).

propose its own technical group system – quickly emerged as a defining feature of the new organization. However, these groups were rechristened as "professional *technical* groups," thereby representing the legacy of the AIEE technical committees. But as McMahon explains, "[s]ome of the AIEE's Technical Committees were immediately absorbed into Groups; others retained their committee status, to be later merged into the Groups system" (1984, pp. 242-243). As suggested by this summary, the technical committees were quickly engulfed by and integrated into the professional groups, which were larger in size and largely self-governing.

Following a member vote that approved the AIEE-IRE merger by a relatively large margin, the completion of the merger at the "headquarters" level was officially complete by January of 1963, resulting in the official establishment of the Institute of Electrical and Electronics Engineers, or IEEE (Ryder and Fink, 1984, p. 225).[79] Ongoing efforts to merge the various committees and groups of the prior parent organizations took somewhat longer to complete. In fact, this process proceeded with some variability from group to group, as reflected in the unique amalgamation that brought together the AIEE CDC and the IRE PGEC. One early step in this process involved the late 1962 formation of a four-member Joint Study Committee, consisting of representatives from both the AIEE CDC and the IRE PGEC. As retrospectively explained by Anderson, the merger process required the working out of many subtle and not-so-subtle differences in the traditions and preferred approaches of the two groups, ranging from the processes by which leaders were selected to the use of different rules for peer review (Anderson, 1976, p. 49).

Following a mandate from IEEE headquarters, by early 1963 the PGEC was officially renamed the IEEE Professional *Technical* Group on Electronic Computers, or "PTGEC" (Anderson, 1963b). The Joint Study Committee subsequently prepared a plan for the merger of the group and the committee. This plan was approved in mid-1963, leading to the formation of a four-member Constitution of and Bylaws Committee (Anderson, 1963c). By February of 1964 PTGEC Chairman Walter Anderson noted the "careful design work" thus far involved with the merger, an apt description given that the process was being led by engineers (Anderson, 1964a).

[79] Ryder and Fink describe this process under a heading that reads "The IEEE is Born" (1984, p. 225). However, I prefer the more nuanced – and perhaps more accurate – framing presented by McMahon: "The makers of the IEEE had drawn copiously on its tangible past and, so, in a real sense, this new national engineering society was formed, not founded. Its technical fields, publications, and convention habits were only the most obvious components of a rich and detailed inheritance" (1984, p. 243).

And while Anderson also complained that the merger was progressing with "more rigor than vigor," in April of 1964 the merger of the CDC and PTGEC to form the "IEEE Computer Group" (or "IEEE CG") was officially announced (Uncapher, 1964a). As incoming Chairman Keith W. Uncapher pronounced, "The event marked the culmination of a long effort to create an effective organization whose service to its members and to computer technology would be greater than the sum of the prior independent contributions of the CDC and PTGEC" (Uncapher, 1964a, p. 184).

One major aspect of the merger processes that appears to have advanced relatively smoothly involved the development of a committee structure for the two groups. As noted above, the AIEE CDC was organized around a number of subcommittees, and the IRE-PGEC launched its own technical committee structure in the early 1960s.[80] In fact, PGEC Chairman Anderson announced the formation of the first TAC in 1962 (1962). Dedicated to the area of Logic and Switching Theory, from the start the new group was recognized as a joint AIEE-IRE committee. By early 1965 the twelve Technical Committees of the newly-formed Computer Group were listed in the *IEEE Transactions on Electronic Computers* ("IEEE Computer Group," 1965).

Yet a closer examination of the merger reveals some key variations in the framing of the new group's identity and scope. As noted above, PGEC chair Arnold A. Cohen proposed in 1961 that the PGEC be renamed the Professional Group on Information Processing Systems (PGIPS), which tended to emphasize the group's broad interests in all phases of information processing. On the other hand, during the lead-up to the merger Cohen emphasized that "the combined computer wings of the two Institutes contain the ingredients for a strong, effective computer engineering organization" (1962a). Cohen reiterated this statement verbatim in a second letter, also published in early 1962 (1962b). As merger activities continued to ramp up, ongoing discussions about the name of the group hinted at further questions about the extent to which the group's interests extended beyond the domain of "computer engineering."

In a 1963 letter published in the PTGEC's *Transactions*, Louis Fein picked up where Cohen's prior PGIPS suggestion had left off (Fein, 1963). Fein started by asserting that any new name for the organization should be broad enough to cover its full range of activities and interests, yet not so broad that it "includes the principal activities and interests of other groups of

[80] Through the 1950s and into the early 1960s, the IRE continued to maintain technical committees at the top level of the organization, largely outside the purview of any single professional group. However, these were primarily dedicated to standards (Chase, 1961, p. 912).

engineers and scientists." As I discuss in more detail below, by this time Fein was acutely aware of the professional and disciplinary politics that were in play. Further clarifying his view of the PTGEC's scope, Fein posited that the membership was primarily concerned with "the theory and practice of the design, construction, test, operation and maintenance of reliable components, circuits and equipment to be used by itself or as part of a larger system." He also emphasized the importance of selecting a name that included or implied terms such as "equipment," "systems," and the associated "purposes" thereof. After declaring that a variety of existing and new names were inadequate, the author ultimately proposed "Professional Technical Group on Synnoeta."[81] Referencing his own 1961 article that proposed the establishment and development of a new field that he dubbed "synnoetics" – or the "computer-related sciences" – Fein explained that the Greek-derived term "synnoeta" referred to the use of systems or equipment in the performance intellectual tasks. He concluded his letter by noting that organizations such as the ACM might also "get around to changing their names to more adequately describe their scope of interest and activity."

While Fein's proposal generated little in the way of follow-up commentary or discussion, the leaders of the PTGEC were nonetheless toying with other naming possibilities during this important transitional period. By early 1963, for example, PTGEC Chairman Anderson started using the simple phrase "Electronic Computer Group" to refer to the organization (1963a), and in June 1963 he both commented on the group's "alphabet soup" name and suggested that it might be more appropriate to adopt a more simple title such as the "Computer Group" (1963c). He even signed his letter as the chairman of the "IEEE Computer Group," even though the group was still officially known as the PTGEC. In early 1964, however, Anderson noted that moves were afoot to adopt the name "*Society* for Electronic Computers," which was intended to reflect the organization's true size and scale (1964a). And indeed, many commentators around this time were quick to point out that the organization's membership had topped 10,000.

But as the merger progressed, yet another possible name surfaced. In an April 1964 letter, chairman Anderson explained that the newly drafted constitution included "a new name for the group, the *Society for Computer Sciences*" (Anderson, 1964b). And the following month, the

[81] The other names discussed by Fein included the Professional Technical Group on Intellectronic Systems (with the term Intellectronics borrowed from Simon Ramo), Professional Technical Group on Information Processing Systems (with an explicit reference to Cohen's PGIPS suggestion), and Professional Technical Group on Automata (Fein, 1963).

trade magazine *Datamation* carried a news item that referenced this same name ("SJCC Society Gleanings," 1964). Later in the year, ad hoc Secretary and former PGEC chair Arnold A. Cohen cautioned that the group should be referred to as the IEEE "Electronic Computer Group" until the name change was approved more formally (1964). And by the end of 1964, Uncapher indicated that the official name of the organization was the "IEEE Computer Group," but he added that the name was subject to revision in light of a more general overhaul of the IEEE's naming conventions that was planned for 1965 (Uncapher, 1964b). At least for the time being, the IEEE's interests in the computer field would remain under the purview of a group rather than a more autonomous society.

Published at the beginning of 1965, the group's new Constitution and Bylaws proclaimed that "[t]he name of this organization shall be the Computer Group of the Institute of Electrical and Electronics Engineers" ("IEEE Computer Group Constitution and Bylaws," 1965, p. 2). Yet the phrase "computer sciences" was also featured prominently in this document. In contrast to earlier PGEC constitutions – which emphasized the group's orientation toward "computer engineering and allied arts and sciences" – the new constitution explained that "[t]he group shall strive for the advancement of the theory and practice of computer sciences" (p. 2). As I discuss in more detail below, the term "computer science" was coined in the late 1950s, and it entered wider circulation in the 1960s. As suggested by the Computer Group's new constitution, the phrase had broad appeal among computer professionals, including many computer-oriented engineers. In fact, it was so appealing that it displaced any reference to engineering in the Computer Group's statement of objective.

The group's scope, as outlined in the second article of the new Constitution, also hinted at possible shifts and expansions in the organization's agenda ("IEEE Computer Group Constitution and Bylaws," 1965, p. 3). In most general terms, this declaration closely followed the statement of scope that was established in 1961 for the PGEC's *Transactions on Electronic Computers*. In fact, the first four sections of the new scope mirrored items (a) through (d) of the prior statement in near verbatim form. The fifth section, however, was suggestively revised. As noted above, part (e) of the original statement declared that "application, use, and programming" were within the scope of the journal, but only as long as these subjects were related to issues of "design and operation." Yet the new statement declared that the group's scope more generally encompassed "[a]pplications, use, and programming of digital and analog computing devices and

information processing systems and the use of computers in electrical and electronic engineering" (p. 2). While this statement placed particular emphasis on computer applications in electrical engineering – perhaps not surprising given the historical orientation of the parent Institutes – the first part of this passage removed the "design and operation" qualifier. At least hypothetically, this change expanded the computer field's settlement to cover an even wider swath of sociotechnical territory.

The Constitution more clearly spelled out certain parts of the group's settlement via specific statements of scope that were written for each of eleven technical committees.[82] In fact, many of these statements spelled out both "included" and "excluded" subjects, topics, and technologies, suggesting that the committees were being positioned with respect to one another, as well as other groups within and beyond the IEEE. And while many of these committees were dedicated to areas that had long been associated with computer design and engineering – to name a few, Logic and Switching Theory, Computer Systems, Computer Elements, and Reliability – others were in more contested areas. In fact, the committees that impinged most directly on other fields and groups were described in rather strategic terms. The scope of the Programming committee, for example, was framed as including:

> Treatment of the theory and development of generalized computer programs, especially those falling in the categories of assembly programs, compiler programs, executive programs and processors for problem-oriented programming languages which are used in the writing of working programs by a broad segment of users (p. 6).

As suggested by this passage, the interests and activities of this committee were described as reaching only so far into the domain of programming. More specifically, both the categories listed and the use of phrases such as "generalized computer programs" reveal that the committee's scope included "system software" and "programming software," but largely stopped

[82] While a total of twelve "Technical Committees" were listed in the *IEEE Transactions on Electronic* Computers in early 1965, the "Standardization" group was reclassified in the Constitution as one of ten "Standing Committees." The eleven technical committees included Logic and Switching Circuit Theory, Computer Systems, Computer Elements, Programming, Reliability, Applications in Management Data, Applications in Automation Processes, Design Automation, Data Acquisition and Transformation, Design Evaluation and Simulation, and West Coast Committee.

short of end user applications. This statement of scope therefore revealed a point of overlap and perhaps even conflict with the ACM, which maintained wide-ranging interests in software.[83]

The formation of the IEEE Computer Group was a significant development, as it combined the activities of the two major electrical engineering groups that maintained settlements in various areas of the computer field. Further, the establishment of this group provided its members and leaders with opportunities to revisit and refine the relation of electrical engineers, computer technology, and computing. From a more systems-oriented perspective, the formation of the Computer Group was a potential source of disciplinary and professional instability. That is, bringing computer-oriented engineers together in a single organization demanded a renegotiation of settlements with other groups, such as the ACM. As I discuss in the following sections, unifying bodies such as the National Joint Computer Committee and its historical successors played important roles in maintaining a measure of stability in a dynamic and rapidly evolving system of professional societies.

Stabilizing the System: The Joint Computer Conferences and Committees

As discussed in the preceding chapter, the early Joint Computer Conference series provide early evidence for the emergence of a distinct field of "computer engineering." Yet through the mid and late 1950s, the joint conferences were an increasingly important point of intersection for all of the major phases of the computer field. In addition to being both well-known and well-attended by a reasonably wide variety of computer professionals, the JCCs tended to cover an array of topics, even if many of the individual conferences were topically skewed toward particular phases of the field. Further, it is worth emphasizing that these conferences were not "co-located." Rather, these were true joint meetings, organized, executed, and attended by members of the ACM, AIEE, IRE, and later the IEEE. Taking a closer look at the history of these events reveals the role they played in helping to balance the computer field's competing forces of integration and fragmentation.

To begin with, the orientation of the Joint Computer Conferences toward engineering and design topics remained particularly strong in the early years of the event. In a summary review of

[83] The Constitution also identified and described committees that were dedicated to specific domains of application, such as "Applications in Management Data" and "Applications Automation Processes." Yet these groups were also qualified accordingly. The former, for example, was framed as being primarily focused on the selection of "desirable equipment and systems characteristics" ("IEEE Computer Group Constitution and Bylaws," 1965, p. 6).

the 1953 Eastern Joint Computer Conference (EJCC), for example, Householder noted that the joint conferences tended to emphasize computer design and construction, while ACM meetings were often focused on applications (1954). Yet he also observed that the computer field "is on the way to disintegration into a hundred little field of specialization" (p. 6), and he argued that future ACM and JCC meetings might be organized in ways that countered this tendency. Indeed, the official addition of the ACM as a full sponsor and participant in the joint conferences in 1953 looked like an important step toward greater collaboration and cooperation among the AIEE, IRE, and ACM. The three-way composition of the joint committee was codified in one of the group's first official organizational documents. Titled "Organization of the Joint Computer Committee" (1954) and published in the proceedings of the 1954 EJCC, this document explicitly specified that the JCC was "jointly and equally" sponsored by the ACM, AIEE, and IRE.[84] This same document also made it clear that the principle activity of the group centered on planning and running the joint conferences, thereby mitigating against the possibility that the joint committee might take over the territories or activities of the three participating societies.

This same statement of organization also specified a fifteen-member governing body for the committee, composed of five members from each of the participating groups. This composition clearly favored the engineering organizations over the ACM.[85] The tilting of the joint committee toward engineering was also reflected in this document. A rather suggestive "statement of object," for instance, indicated that "[t]he Committee shall aid in the promotion of close co-operation and co-ordination in the activities of the sponsoring societies related to the field of computer engineering and allied arts and sciences" (p. 91). And while the "general scope" of the Joint Computer Committee was described as principally stemming from the scope of each participating society, a subsequent passage on the committee's "major interests" suggestively declared: "The major field of interest of the JCC shall be the engineering aspects of the design, development, manufacture, and use of computers, but shall also include an interest in

[84] However, subsequent commentators recognized the secondary position of the ACM in the early years of the joint conferences. For example, in 1959 Paul Armer explained that, with respect to the early joint conferences, "the ACM was a second class citizen for a period of time" (Is it Overhaul," 1959b, p. 19).
[85] More specifically, the fifteen-member committee included three ex-officio members, namely the president of the ACM, chairman of the AIEE CDC, and chairman of the IRE PGEC. Four additional members from each group filled the remaining twelve spots. Further, each group's four representatives were evenly divided between East and West Coasts.

the various activities that contribute to this field or utilize the products or techniques of the field" (p. 91). Such statements clearly place primary emphasis on engineering over applications.

By the mid-1950s, however, shifts in the orientation of the joint conferences were becoming more evident. In a foreword published with the proceedings for the 1955 Western Joint Computer Conference (WJCC), for example, conference manager William L. Martin noted that "[t]he Conference has changed in character from a meeting of small groups of specialists discussing problems of mutual interest to large meetings involving people from many phases of engineering, management, business control" (1955, p. 1). The 1955 EJCC, on the other hand, was topically dedicated to "Computers in Business and Industrial Systems," and conference chair John G. Brainerd explained in a keynote address that this topic area "should be of major interest to both [computer] creators and users" (1955, p. 6). Brainerd also noted in his talk that the Joint Computer Committee was dedicated to organizing conferences and carrying out other relevant work, "so as to avoid duplication of effort on the part of its sponsors" (1955, p. 6).[86]

The pendulum swung back to the "engineering phases of computers" for the 1956 WJCC (Whitby, 1956), and in mid-1956 ACM President Householder generalized that the joint conferences "relate mainly to hardware" (1957, p. 1). Yet when the joint computer committee revised its statement of organization in 1956, a number of minor revisions revealed ongoing changes in its orientation and identity ("Organization of the National," 1956). In addition to being renamed the National Joint Computer Committee (NJCC), this document included a revised statement of "major interests" that omitted the phrase "engineering aspects" (p. 1). Instead, it indicated that the "major field of interest of the NJCC shall be the design, development, manufacture and use of computers, but shall also include an interest in the various activities that contribute to this field or utilize the products or techniques of this field" (p. 1). On the other hand, this same document reiterated that the committee was dedicated to promoting cooperation and collaboration among those sponsoring societies "related to the field of computer engineering and allied arts and sciences" (p. 1). This revised document once again placed subtle emphasis on "computer engineering" over computer use and applications.

In the late 1950s, industry trade magazines periodically carried commentaries about the computer field's major professional societies and conferences. In 1958, for example, *Datamation*

[86] More than three decades later, IBM's Morton Astrahan similarly recalled that one of the primary, original purposes of the NJCC was "[t]o promote cooperation instead of duplication of effort" ("Reflections on a Quarter-Century," 1986, p. 228).

published a series of candid remarks from attendees at the most recent WJCC. Commenting on the ongoing shift of the joint conferences away from issues of engineering and design, Eric Weiss explained that "there are fewer design people and many more 'users' at this conference. … Unfortunately, there's not much in the way of new things here for engineers" (quoted in "Post Conference Feedback," 1958, p. 20). And while many factors likely contributed to this trend, others were quick to note that the increasingly commercialized and competitive computer field was having an impact on the content and climate of the joint conferences. In fact, Weiss himself noted the "ultra cautious" attitude of many conference speakers, and he added that "[t]hese conferences could be much more effective if a freer exchange at panel discussions and technical sessions were possible" (p. 20). Another conference attendee noted the possibility that many companies were holding back information for competitive reasons. He went on to describe the conference presentations as "a kind of game played with information. On the one hand, a firm doesn't want to tip its hand but in a year they would like to say that a year ago they presented a paper on a new development" (p. 20). As these remarks reveal, the close relation of computer-oriented engineers with industry was having a negative impact on their ability to participate fully in some professional activities.

A 1959 *Datamation* editorial expressed further concerns about the increasingly fragmented character of the joint conference series. Authored by Rand Corporation employees Keith Uncapher, Malcolm Davis, James Babcock, and Shirley Marks – the former two explicitly identified as engineers, the latter as "programmers" – the article argued that "[t]o stress the joint aspect of a joint computer conference, subjects should transcend the divisions between engineers, programmers, and users of computers" (Uncapher, et al., 1959). These authors clearly recognized the potential for the joint conferences to help unify the field. In fact, the authors noted in this same piece that a number of speakers at the recent December 1958 EJCC had clearly "recognized that professional personnel, occupied in diverse branches of the computing industry, must one day come together in a unified effort." While this call stood in tension with many of the sociotechnical schisms that were growing up in the field, other outspoken commentators expressed similar views, a point to which I will return. Further, the types of concerns expressed in this editorial revealed that many questions remained about the future potential of the joint conferences to promote what the authors called a "unity of purpose" for the computer field.

In subsequent years the JCC pendulum continued to swing between two ends of a sociotechnical spectrum. According to *Datamation*, some complained that the 1959 EJCC "seemed too hardware oriented with not enough emphasis on applications" (Grems, et al., 1960, p. 25). Yet countervailing tendencies prevailed at the 1960 EJCC, as reflected in one commentator's claim that "the relative emphasis on usage problems as opposed to hardware papers was greater than at any EJCC since the 1955 Boston Conference" (Heising, 1961 p. 36). Attempting to explain these trends, the author suggested that "new usage techniques are required to keep pace" with advances in computer design and construction generally, and faster machines specifically (p. 36). As suggested by this commentator's remark, the destabilizing forces of rapid technological change were making it difficult for the joint conference organizers to plan programs that spanned the broad sweep of the field.

Still other conference review articles revealed the relative extent to which the JCCs were attracting attendees from each of the main participating societies. Registration data for the 1960 WJCC, for instance, revealed strong attendance from ACM and IRE members, who respectively made up about 38% and 26% of all registrants (Barnard, 1960). However, those who maintained affiliations with the AIEE and neither the ACM nor IRE made up only about 3% of registered attendees (p. 23).[87] The 1961 EJCC, on the other hand, was clearly dominated by ACM members. According to a post-conference report by ACM President Harry Huskey, "Of those who registered, 60 per cent [sic] were from ACM, 30 per cent from IRE, and 3 per cent from AIEE" (1962b). Not only do these data highlight the relatively marginal role of AIEE members in these events, they also suggest that the ACM and its members were moving to the forefront of a joint conference series that was originally initiated and dominated by engineers.

In summary, the NJCC and its conference series were a well-established institution by the late 1950s and early 1960s, despite the ebb and flow of individual conferences. The three participating organizations had settled into stable and effective patterns of cooperation, and the joint conferences were characterized by expansive programs, large crowds, and healthy financial returns. In fact, by 1959 RAND's Paul Armer quipped, "If the JCC meetings get any larger there

[87] While these statistics excluded those AIEE members who were also members of the IRE and/or ACM, the data nonetheless suggests AIEE members were relatively sparse at many of the joint conferences.

will be damn few places where they can be held just because of their size" ("Is it Overhaul," 1959a, p. 33).[88]

On the other hand, this period of apparent stability and vitality was threatened by at least three major challenges. First, concerns persisted about the purpose and scope of the joint conferences, especially in light of ongoing tendencies toward fragmentation and deepening specialization, within both the participating societies and the field writ large. Second, other professional groups with interests in the broad domain of "information processing" were clamoring to participate more formally in the joint committee and conferences. For example, the National Machine Accountants Association (NMAA) – which later became the Data Processing Management Association (DPMA) – lobbied for this type participation through a 1959 presentation to the leaders of the NJCC, but with little success ("Reflections on a Quarter-Century," 1986, p. 235). Members of Simulation Councils, Inc. also tried to become formal participants in the NJCC in the 1950s, and they were similarly rebuffed ("Reflections on a Quarter-Century," 1986, p. 235). Using terminology originally developed by Callon (1999), the joint conferences and committee had become an "obligatory passage point" that was exclusively controlled by representatives of the AIEE, IRE, and ACM (Latour, 1987).[89]

A third major challenge to the stability of the field stemmed from growing demands for a single and more cohesive organization that could represent all phases of the computer field, especially on the international stage. By the late-1950s, these and other undercurrents were threatening to upset this system of professional societies, as well as the stabilizing agent known as the NJCC. For additional perspective and background on these trends, I turn to a well-known symposium that was held at the RAND Corporation in 1959.

---

[88] Armer was well-qualified to make such a remark, as around this time he served both as an ACM representative to the NJCC and as a vice-chair of the NJCC.

[89] That is, achieving full participation and recognition in the computer field required participation in the joint conferences, but the joint committee made moves to block the expansion of the committee beyond the original three groups. In fact, former NJCC members have recounted how the representatives of the NMAA were treated "incredibly rudely" when they made their 1959 request to join the joint committee. As Armer summarized, the NMAA "wasn't considered a professional society. They were punched-card people, and we looked down our noses at them" ("Reflections on a Quarter-Century," 1986, p. 235). Rubinoff noted yet another important reason for why the NMAA was turned away: "[T]here was already enough internecine warfare" ("Reflections on a Quarter Century," 1986, p. 235).

“Is It Overhaul or Trade-In Time?”: The 1959 Rand Symposium

In 1958, the RAND Corporation hosted the first in a series of invitational symposia that were scheduled in tandem with the annual WJCC. These events typically involved small groups of prominent computer professionals who engaged in candid discussions about some of the pressing issues that faced their field. The second such event, held in March of 1959, brought together seventeen individuals to explore a number of topics that are particularly relevant to the present analysis. Many of these persons were affiliated with commercial interests, but four hailed from universities or university labs, one came from the Department of Defense, and three were employed by RAND. The group also included at least one individual with close ties to each of the major organizations in the computer field, namely the NJCC, ACM, IRE, and AIEE.

Not only was the resulting discussion provocative and wide-ranging, it was recorded, transcribed, and partially published in *Datamation* (“Is it Overhaul,” 1959a; 1959b). And as these transcripts revealed, the participants spent significant time on one specific agenda item that queried, “What Can Be Done to Increase the Effectiveness of our Professional Organizations?” The *Datamation* editors more creatively captured the essence of the discussion via their own headline: “Is It Overhaul or Trade-In Time? Perennial Professional Society Question Worked Over by Computer Specialists” (1959a, p. 24). As participant Paul Armer later explained, the attention heaped on this particular topic was significantly stimulated by ongoing questions about whether and how to allow other professional groups to join the NJCC (“Reflections on a Quarter Century,” 1986, p. 230). Armer also noted that questions about how to represent U.S. interests in information processing at the international level also loomed large, a point to which I return.

In most general terms, few could disagree with one participant’s assertion that the major groups in the computer field were united by a common interest in “that big box of stuff sitting in the middle of the room” (“Is it Overhaul,” 1959a, p. 24). Yet in discussing how interest in this boundary-object was – or *should* be – parceled up among the relevant professional societies, the attendees painted a rather discordant picture. Participant Herb Bright – at the time a Westinghouse engineer who was also affiliated with the ACM– characterized the relation of the AIEE, IRE, and ACM as an “impossible jurisdictional mess,” and he added that the three organizations “pretend to be working together when actually they’re fighting each other tooth and nail” (“Is it Overhaul, 1959a, p. 30). And as aptly summarized by well-known computer pioneer and instigator Herb Grosch, “we have two warring hardware groups and one poor

moribund user's group, all trying to work together in this JCC farce" ("Is it Overhaul," 1959b, p. 26). Even

Providing further insights regarding these apparent jurisdictional conflicts, other participants hinted at persistent tensions between the "gentlemanly" ideals of computer professionals and the expansionist agendas of the major computer groups. For example, aforementioned Bright explained: "Each one of them [the ACM, IRE, and AIEE], in a way which it piously hopes is gentlemanly and forward-looking is trying to cover some of the same ground as the other two" ("Is it Overhaul," 1959b, p. 25). Pointing to the seemingly unique nature of the situation, Bright added the three groups were "trying to make believe their interests don't conflict" ("Is it Overhaul," 1959b, p. 25). And in a later part of the discussion that was focused on the persistent schisms in the field between hardware and applications, Willis Ware put a slightly different spin on the issue when he explained that "[t]here seems to be a gentlemen's agreement between the ACM and the PGEC to keep the division that way" ("Is it Overhaul," 1959b, p. 23).[90] These statements reveal that ongoing jurisdictional negotiations between the ACM, IRE, and AIEE were often characterized by an outward appearance of stability and civility but with many persistent undercurrents of instability and conflict.

Still other participants expressed more specific concerns about the relevant professional societies, while also proposing various approaches to reorganizing the field. Philco Corporation employee and outgoing AIEE CDC Chairman Morris Rubinoff, for instance, noted that a relatively small fraction of IRE and AIEE members had a primary interest in computers, and he concluded that "the IRE and the AIEE should recognize that they are primarily electrical engineers with a side interest in computers as such" ("Is it Overhaul," 1959b, p. 26). And at another point he noted that those interested in "hardware" should get involved in the AIEE or IRE, "which should be merged in any case" (Is it Overhaul, 1959b, p. 19).[91] And while Rubinoff clearly recognized the role of designers and engineers in the field, he also put forward the idea that "all computer activities should filter through the ACM. All other groups should then affiliate

[90] At the time, Ware was employed by the RAND Corporation and serving as chair of the IRE-PGEC.
[91] In another suggestive passage, Rubinoff noted that the IRE and AIEE tended to publish the same types of material. Indicating that he was serving on editorial boards for both groups, he explained, "If a paper is submitted to me in my capacity with Society X then I'm going to place it in the publication of Society X. But, actually, the man [sic] really doesn't know where to send it, because we haven't defined what is going to be published" ("Is it Overhaul, 1959a, p. 27).

with the ACM" ("Is it Overhual," 1959b, p. 26).[92] Indeed, many participants noted that the ACM appeared best-suited to take on such a role, especially given its size, scope, and independence.

However, this proposal stood in tension with Rubinoff's own assertion that the ACM had largely failed to take a leadership role in the field, and he suggested that "the ACM should have some new blood injected intravenously" ("Is it Overhaul," 1959b, p. 26). Rubinoff also quite correctly noted that if the ACM moved to the forefront of the field, "[a] lot of people are going to feel that they have had the rug pulled out from under them" ("Is it Overhaul," 1959b, p. 26). In light of these and other concerns, Rubinoff's idea gained little traction. Saul Gorn, who had ties to the ACM, was also sympathetic with the idea of "reformulating" the Association so that it could take on more of a leadership role. Yet even Gorn acknowledged the inflexibility of the group's constitution, and at one point he added that the "ACM is in the throes of trying to find out the extent of its own amorphousness and decide what its justification is" ("Is it Overhaul," 1959b, p. 19). Others offered even more forceful critiques. UCLA's Curtis B. Tompkins, for one, suggested that the Association "doesn't have any status goals that are adequate at the moment" ("Is it Overhaul," 1959a, p. 24). Grosch, on the other hand, referred to "[t]he void left by the ACM" ("Is it Overhaul," 1959a, p. 25), and he later disparaged the Association and its leaders for focusing on the scientific and academic side of computing while ignoring the growing importance of computers in the business sphere ("Is it Overhaul," 1959a, p. 33).

In light of such concerns, participants put forth a number of additional ideas for reorganizing the field. One approach centered on expanding the role of the NJCC, although participants acknowledged that the committee's form and charter limited its ability to do anything but organize conferences.[93] Other options proposed at the symposium included the creation of either a new membership-based organization or an "Institution of Societies." Ultimately sympathetic toward the latter idea, the discussants proposed the formation of an "American Association for the Advancement of Computing" (AAAC), with explicit reference to the American Association for the Advancement of Science (AAAS) as a prototype organization

[92] While Rubinoff's sympathy with this idea may appear unusual, the interests of the AIEE CDC in the 1950s often tilted toward the application of computers in various areas of engineering. And the CDC – unlike the IRE-PGEC – was a relatively small group that stood to lose relatively little if the ACM rose in prominence. But as noted below, control over the lucrative joint conferences was a sticking point for all of the groups involved, including the AIEE.

[93] As Grosch noted, the NJCC was "reaching for responsibilities which their charter really prohibits. … Most of the people who get in there really want more responsibility, but this is withheld from them by the nature of the committee charter" ("Is it Overhaul," 1959a, p. 25).

("Is it Overhaul," 1959b). As suggested by some attendees, such an Association could accommodate broad participation by users groups, manufacturers, and a wide range of professional societies, including those with both direct and indirect interests in the field.

The discussants clearly recognized the challenges and difficulties that came with bringing such an Association to fruition. As Bright summarized, "Each of these organizations wants a solution which fits the area which it has carved out for itself, which overlaps the other areas. You've reached a real impasse here" ("Is it Overhaul," 1959a, p. 27). But in an important sense, commentators such as Bright failed to recognize the extent to which the stability of this system of professional societies was maintained not via rigid jurisdictional claims, but rather through the ongoing negotiation of sociotechnical settlements. Hence, the overlapping and interpenetrating interests of these groups were not by definition problematic, as evidenced by the ability of these organizations to work together and co-evolve through much of the 1950s. However, maintaining balance and stability both within and between these professional societies was made more difficult by the many currents of change that increasingly pervaded all aspects of the field.

And in the end, the participants in the 1959 RAND symposium offered little in the way of hard and fast recommendations regarding how this system of professional societies might be reformed or reorganized. Yet the publication of parts of their conversation clearly drew attention to many of the issues that were play, and some of the ideas discussed at the meeting were later realized in the early 1960s transformation of the NJCC into the American Federation of Information Processing Societies (AFIPS).

From the NJCC to AFIPS: Preserving Stability in the System

Understanding the ultimate fate of the NJCC demands that we take a step back to examine the formation of the International Federation of Information Processing Societies (IFIPS). And indeed, tracing out this development requires that we look at some of the earliest efforts to organize an international conference on information processing, an idea that can be traced back to at least 1955 (Auerbach, 1986a, p. 180). After the concept for such a conference was presented to the NJCC by Isaac Auerbach – an engineer who at the time was working for Burroughs Corporation – an ad hoc committee composed of AIEE, ACM, and IRE

representatives was formed to pursue the idea (p. 180).[94] This led to the 1957 submission of a proposal to UNESCO (United Nations Educational, Scientific, and Cultural Organization) that called for the organization of a conference "to promote a freer exchange of technical information among leading scientists and engineers of many nations" (p. 181), especially in the area of information processing systems. UNESCO support was garnered through a series of meetings involving an array of international experts in many areas of computing and information processing, and two important developments followed. The first of these involved the organization and execution of a series of international conferences, as originally envisioned by Auerbach. The inaugural event, held in Paris in 1959, was dubbed the International Conference on Information Processing (p. 184). As recounted by Auerbach, it attracted 1,800 participants and attendees from 38 countries and 13 international organizations.

A second important development involved the formation of the International Federation for Information Processing (IFIP) in 1960. This so-called "society of societies" gained primary fame for organizing and executing subsequent international conferences, although the group was also involved in other important activities, such as in the area of standardization (Auerbach, 1986a, pp. 186-187).[95] The Federation has also been credited with helping to usher in the new terminology of "information processing," which gradually started to displace alternative terms such as "computers" and "computing." But even more importantly for the present analysis, the initial formation of IFIP required that just one professional society from each participating nation could join the Federation. Hence, the process for approving the formation of IFIP in the U.S. was rather convoluted, with the ACM, AIEE, and IRE separately approving the IFIP statutes and authorizing the NJCC to report these decisions back to IFIP. As Auerbach explains, "all hell broke loose" when the statutes started to make their rounds for approval, as many claimed that

[94] According to Auerbach, this committee consisted of Samuel N. Alexander of the National Bureau of Standards representing the AIEE and Alston S. Householder of the Oak Ridge National Laboratory representing the ACM. Auerbach, who at the time was employed by Burroughs Research Laboratory, represented the IRE.

[95] The formation of IFIP significantly trailed the 1956 founding of the International Association for Analogue Computation (AICA). While initially focused on analog computing, the scope of the group expanded in subsequent years, and in 1976 it was renamed the International Association for Mathematics and Computers in Simulation (IMACS). The AICA founding was also shortly followed by IFAC in 1957 (Automatic Control), IMEKO in 1959 (Measurement), IFORS in 1959 (Operations Research), and IFIP in 1960. The activities of the five groups were coordinated under the purview of FIACC (Five International Associations Coordinating Committee), which was established in 1972 with UNESCO support ("Call for Papers," n.d.).

the NJCC possessed neither the adequate legal status nor authority to deal with an international body ("Reflections on a Quarter-Century," 1986, p. 230).

The statutes were ultimately approved by the ACM, IRE, and AIEE, but the process helped push the leaders of the computer field to develop a more unified national voice. An NJCC committee chaired by Harry Goode was charged with addressing the major issues in play, such as determining how to involve other professional groups in the NJCC or its successor, as well as establishing representation for the United States in IFIP (Auerbach, 1986b, pp. 258-259). Goode and his committee approached their task with both rigor and caution, and by 1959 they had developed a "Proposed Constitution for a Federation of Information Processing Societies." This document that directly modeled on the constitution for another Federation, namely the American Institute of Physics ("Reflections on a Quarter-Century, 1986, pp. 235-236). After various adjustments were made to this document, support was garnered from each of the participating societies, and at a meeting in mid-1961 the NJCC was officially replaced by the "American Federation of Information Processing Societies" (AFIPS).[96]

In general, the orientation, scope, and purpose of AFIPS pointed in a number of important directions. For instance, the organization's new constitution indicated that "[t]he purpose of this Federation shall be the advancement and diffusion of knowledge of the information processing sciences and for literary and scientific purposes … These sciences include, but are by no means restricted to, the computer sciences and their applications to Society" (252). This statement stood as yet another reflection of the increasingly widespread use of the term "science" – rather then "engineering" – to describe wide swaths of activity in the computer field. This was also a significant departure from the NJCC's prior statement of organization, which emphasized the group's orientation toward "the field of computing engineering and allied arts and science" (1956, 1). Further, AFIPS was conceived so that its activities could potentially extend beyond the organization of joint conferences. As Ware later explained, AFIPS was intended as "*the* preeminent national single spokesperson for the

[96] While the influence of the aforementioned RAND Symposium on this process is not entirely clear, at least three of the participants were in some way involved with the drafting and approval of the AFIPS constitution. And one of these individuals, namely Willis Ware, was elected the first Chairman of AFIPS. More recently, Armer has claimed that the NJCC committee recommendations – which in turn led to the AFIPS constitution – were "essentially the consensus of the participants in the Rand symposium" ("Reflections on a Quarter-Century," 1986, p. 231).

computer," and to some extent its constitution was tailored accordingly (1986, p. 304, author's emphasis).

Yet in other ways, the formation of AFIPS involved a rather conservative transition from the NJCC. For example, AFIPS was established as a "society of societies" that was headed by twelve directors, four each drawn from the ACM, AIEE, and IRE. The structure of AFIPS was also largely synergistic with IFIP, its international counterpart. As succinctly described by Robert Rector, who was closely involved with the formation of AFIPS, "IFIP provided a working model of a federation, and the NJCC was a readily available structure to do the building" (1986, p. 262). Of course, many other factors impinged on the ultimate form and function of AFIPS. Ware, for instance, has noted that time pressures were a major issue, especially for those who preferred the much more difficult task of either forming a new membership-based professional group or moving an existing group to the forefront of the field (1986, pp. 303-304).

It is also worth noting that AFIPS was intentionally organized in a "non-threatening" manner. That is, the constitution of the group largely preserved the delicate balance of power that had been worked out by the three major constituent groups over roughly a decade-long period. In fact, Rubinoff has noted that the ACM was particularly "nervous" about the extent to which AFIPS might take over its "turf" ("Reflections on a Quarter-Century, 1986, p. 231). This feeling of vulnerability was likely heightened by the different characteristics of the ACM – which was largely composed of programmers, computer users, and academics – and the IRE and AIEE, which were large organizations with membership rosters and activities that extended well beyond computing (p. 231). If AFIPS somehow displaced the IRE-PGEC, for example, the IRE would surely persist. But if the ACM was similarly challenged, it might ultimately face decline and even dissolution. As Rubinoff has argued, "The organization that stood to lose most was the ACM" (p. 231).

As a result of such concerns, various protections were written directly into the AFIPS Constitution. It stated, for example, that "the Federation shall do nothing that is in direct competition with the activities of its member societies" ("Reflections on a Quarter-Century," 1986, p. 231). This particular issue was further mitigated as a result of the AIEE-IRE merger, which led to the replacement of the eight total AIEE and IRE slots on the AFIPS board with just

four IEEE positions, in parity with the four already held by the ACM (Rector, 186, p. 263).[97] In addition to reducing the chances of an AFIPS-led coup, the structure and constitution of the Federation also helped mitigate against outside encroachments. This issue was of particular salience given both the steadily increasing financial magnitude of the joint conferences and the numerous prior requests from outside organizations to participate formally and officially in the NJCC and its associated conferences.[98] In order to address these issues, the AFIPS constitution mandated that the "full members" of AFIPS – namely the AIEE, IRE, and ACM – retained full control over the joint conferneces. Organizations approved as "affiliate members," on the other hand, remained locked out of the JCC finances, but they were given a vote on the AFIPS board. In 1962, The Simulation Councils, Inc. was approved as the first such affiliate member ("AFIPS Appoints," 1962; Ware, 1963, p. 42).

Many of these themes were evident in a 1964 article that outlined both the position of, and prospects for, AFIPS. As explained by Willis Ware, who at the time was serving as the chair of the Federation, "AFIPS represents the intellectual activity of the entire field of information processing. There is no other organization with such a universal goal" (Ware, 1963, p. 42). The author went on to use the emergent discourse of "software" and "hardware" to outline the unique territories that had been claimed by the two major constituent societies: "The IEEE is largely the hardware population of the computing field, and the ACM, largely the software population which has grown into information processing through scientific computing" (p. 42). Ware also cautiously described the prospective future role of AFIPS. For instance, he indicated that it might be appropriate for AFIPS to take the lead in certain areas, such as in coordinating educational matters, interfacing with the public and other disciplines, or acting as a clearinghouse on standardization issues. Yet he noted that "AFIPS must serve its member societies," and "AFIPS activities will be fully coordinated with and agreeable to is members" (p. 43).[99] Like the NJCC

[97] As further evidence for the relatively good relationship between the ACM and AFIPS around this time, in 1964 a small ad-hoc committee that was charged with reviewing the structure and purpose of the ACM concluded that "[n]o substantial change in ACM-AFIPS relations seems to be called for" (Perlis, 1964, p. 508).

[98] More recent accounts have identified a number of groups that at one time or anotherwanted to join the NJCC, including the NMAA (National Machine Accountants Association, which later became the DPMA), The Simulation Councils, and the Society for Industrial and Applied Mathematics (SIAM) ("Reflections on a Quarter-Century," 1986, p. 235).

[99] Ware's commentary also revealed ongoing anxieties about the appropriate role of other organizations in the information processing field. He indicated, for instance, that The Simulation Councils were

before it, AFIPS was clearly positioned at the center of a "system of professional societies" that had been carefully nurtured for more than a decade. Yet AFIPS was ultimately designed to preserve the stability of this system, and to protect the sociotechnical settlements that had been carved out by its constituent societies. As a result, the ability of the Federation to promote a more thoroughgoing "integration" of the field was limited.

Conclusion

In this chapter I have documented the historical trajectory of a number of professional societies that maintained major interests in the computer field. More specifically, I analyzed the internal development of these groups by discussing ongoing efforts to negotiate their respective identities and sociotechnical settlements. I also tracked trends the composition and activities of these groups, and I emphasized how they interacted with one another in a larger "system of professional societies." Through the 1950s the Joint Computer Conference series and its associated joint committee came to play centrally important roles in this system, especially as the JCCs shifted from being a locus for computer-oriented engineers to a common point of intersection for diverse phases of the field, including the members and leaders of the ACM, IRE-PGEC, and AIEE CDC. Hence, one of my main arguments in this chapter centers on the claim that the joint conferences and committees helped stabilize this system of professional societies, despite persistent tendencies toward specialization and fragmentation. Further, I contend that the JCCs in part reflected and enabled the overlapping and interpenetrating character of the sociotechnical settlements claimed by each of its constituent groups.

My analysis also highlights the impressive overall stability of this system of professional societies. In fact, this stability was preserved through the late 1950s and early 1960s, which was a period marked by rapid technological development, major increases in the number of professionals working in the field, and ongoing expansions in relevant bodies of knowledge. This system also weathered a number of major organizational changes, including the merger of the

---

recognized as an "affiliate" member of AFIPS through their coverage of the "analog and mixed analog-digital aspects of computing" (p. 42). The author also identified two other major organizations with relevant interests, namely the Business Equipment Manufacturers Association (BEMA) and the Data Processing Management Association (DPMA). The former represented manufacturers of office and data processing equipment, while the latter was a membership-based organization composed of individuals (such as accountants) who were interested in computer applications in business (p. 42). Ware acknowledged that the DPMA, as a "society of individuals," was eligible for membership in AFIPS and "may, one day, decide to join" (p. 42).

AIEE and IRE and formation of AFIPS. Yet these changes were also gradually reconfiguring the field in important ways, and by the mid 1960s we find the emergence of new points of parity between the ACM and the IEEE Computer Group. These two professional societies claimed roughly equal numbers of members, they were evenly represented in AFIPS, and they had a balanced stake in the lucrative Joint Computer Conferences.

Further, the dominant image of the ACM had largely coalesced around mathematics, programming, and applications, while the Computer Group was primarily aligned with engineering, design, and "hardware." In many ways, this balance both reflected and reinforced the larger social and technical boundaries that pervaded the computer field. In subsequent chapters, I document how the boundary between computer engineers and other computer-oriented professionals evolved in the commercial sector and educational sectors. I also discuss the ongoing evolution of the divide between hardware and software. As I contend, these "mirrored dichotomies" helped perpetuate and stabilize the unique and evolving structure of the computer field in the United States.

It is also worth noting that there were tentative signs that the ACM was starting to move into a more prominent position in the field, especially in the early and mid 1960s. As noted above, for example, the ACM members and interests were gaining prominence in the JCCs, and in early 1963 the editors of *Datamation* quite directly asked, "Are the Joint Conferences overly software oriented to a point of diminishing returns for the hardware registrant?" ("The Great Conference Debate," p. 25). They went on to note that attendees affiliated with the "software-oriented" ACM were beginning to "far outweigh representation for the IEEE at the JCC" (p. 26). The editors pointed to a number of factors that were contributing to these trends, including rapid growth in the "programmer population" and an expanding gulf between engineers and applications (p. 25). The hardware-oriented sessions at the JCCs were therefore bringing in relatively few attendees, while other engineering conferences seemed to be attracting many of the better hardware papers.[100] Still another editorial that appeared in the upstart trade magazine

[100] J. Don Madden, the chairmen elect of AFIPS, responded to this concern in a follow-up piece, also published in *Datamation*. As Madden argued, more could be done to tailor conference papers, sessions, and programs to mixed audiences of engineers and programmers. As Madden explained, "The software and hardware aspects of computers are becoming so closely interrelated that it is increasingly important for specialists in one area to understand the other" (1963, p. 45). Yet as my analysis suggests, realizing these reciprocal types of understanding proved perennially difficult in actual practice, even though similar calls for "intercommunication" had been surfacing since at least the early 1950s.

Computer Design complained about the lack of "'hardware-oriented' design engineers among the attendees" of the 1963 SJCC, as well as the overall tilt of the program toward "software" and applications over design techniques (Sacks, 1963). This same editorial called for the joint conferences to become a common meeting place for designers and users.

These were interesting developments indeed, especially given that the early joint conferences were strongly linked to both engineers and the electrical engineering societies. And while some of the subsequent joint conferences started to tilt back toward "hardware," it was clear that the landscape of computing had changed dramatically since the field's early years. The Computer Group that emerged out of the AIEE-IRE merger occupied a prominent position on this landscape, especially in light of its large membership and somewhat expansive agenda. Yet the leaders and members of the IEEE-CG also faced an increasingly segmented field and ambitious ACM. As I discuss in subsequent chapters, a host of developments and negotiations helped preserve an overall balance of forces in the computer field from the mid-1960s onward, but not without implication or cost. Before tracing this history forward, however, it is necessary to step back to fill out the rest of this segmentation story. As I argue below, reaching a better understanding of the historical trajectory of the "system of professional societies" described in this chapter requires engagement with other contexts of sociotechnical negotiation, including various places of employment, a number of different educational arenas, and even the sphere of computer technology itself.

Chapter 4

Dichotomous Developments in the Early Computer Field: Profession, Technology, and Education, c. 1955-1963

> "Although there is considerable mutuality of concern in their ultimate objects, 'the advancement of computer technology and application,' hardware personnel and their software peers have long been widely separated by geography, education and interest, and all that is written and said has not as yet made one head out of Humpty and Dumpty. … Perhaps, when the seemingly insurmountable hurdles are charged for the last time, it may suddenly appear that Humpty Dumpty is after all, a single entity and must be fitted properly together to continue sitting high on the wall."
>
> ("A Datamation Staff Survey," 1961, p. 36)

As indicated in the epigraph above, in 1961 the editors of the trade magazine *Datamation* cleverly described the state of the computer field with an analogy to Humpy Dumpty, the well-known nursery rhyme. And despite its whimsical character, this editorial remark carried more than a grain of truth. As discussed in the preceding chapter, the pervasive divide between the so-called "hardware personnel and their software peers" was both reflected in and reinforced by the "system of professional societies" that emerged and evolved through the 1950s and into the 1960s. Yet these groups and their relation stand as pieces in a much larger historical puzzle. In the first part of this chapter, I analyze how the major social and technical dichotomies of the computer field were mirrored and partially stabilized in other sites and contexts, including within both corporate worksites and the sphere of technology itself. My analysis also leads into a discussion of how various actors called into question the dichotomies that separated the field, leading them to articulate alternative visions for the future of computer technology and computer-oriented professional work that were more "integrated" than fragmented.

The second major part of this chapter is focused on still another context – namely the educational sphere – where computer-oriented courses and degree-programs remained largely in flux through the 1950s and into the 1960s. In addition to tracing out the early emergence of new

discipline-building movements in the computer field, I document the evolving role of electrical engineering departments in various spheres of computer design and use. The latter parts of this chapter begin to point to the rising importance of university faculties and departments in ongoing debates over the appropriate relation of the Humpty and Dumpty of computer technology and application, while also hinting at the importance of the academic context as a crucially important site for the formation and development of new disciplines.

Part I – Mirrored Dichotomies: Hardware/Software and Engineer/Programmer

While one can overemphasize the historical evolution of specific words and phrases, tracing out the development of the term "computer engineer" and its variations provides important insights into a number of major currents and trends. In fact, such an analysis reveals how these terms both reflected and reinforced the links between various educational pathways, professional and disciplinary identities, and jurisdictions of knowledge, work, and technology.[101] As noted in Chapter 2, calls for "computer engineers" and "computer designers" started to appear in private-sector employment ads by 1952. In the present section, I follow the use of the term "computer engineer" through the remainder of the 1950s and into the early 1960s. In doing so, I look at how job advertisements for computer-oriented engineers proliferated in the 1950s, especially as the computer industry expanded. My analysis shows how "computer engineer" and a number of closely related terms went through a period of "interpretive flexibility" before moving toward a more stable set of meanings. In doing so, I also emphasize how formal education in electrical engineering was increasingly viewed as a prerequisite for computer design work, thereby promoting the dominant image of computer engineering as both a branch of the engineering profession generally and subfield of electrical engineering specifically. In summary, my account reveals the establishment and normalization of a distinct professional jurisdictions and set of educational requirements for computer designers and engineers.

In subsequent sections, I use a variety of advertisements and other sources to discuss both the divisions of labor that were expanding *within* the domain of computer engineering and the growing schisms *between* computer engineers and other types of computer-oriented workers.

---

[101] As Stuart Shapiro quips, "What's in a name? Plenty." He goes on to argue that many of the labels that have been associated with various aspects of computers and computing "have not been applied as merely post-facto descriptions but as prescriptive models for shaping the IT [Information Technology] profession" (1994, para. 7). This tendency to use disciplinary labels in a prescriptive manner is perhaps most evident in the academic context, but Shapiro's point has salience in the commercial sector as well.

These themes provide a convenient segue to a more general discussion of the computer field's evolving sociotechnical boundaries. More specifically, I juxtapose the fragmentary tendencies of the emergent "software/hardware" dichotomy with a variety calls for the "integration" of the computer field's major divisions of labor, technologies, and bodies of knowledge.

Computer Engineering Identities: From Interpretive Flexibility to Stable Jurisdictions

In Chapter Two I discussed how early employment openings for computer engineers were frequently described as involving design-oriented engineering work in circuits, logic, and systems. Yet in tandem with this trend, variations in the use of the term "computer engineer" started to surface. A 1954 ad from Bendix Aviation Corporation, for instance, called for a "computer engineer" with an engineering degree who was "capable of handling programming in the simulation and study of jet and reciprocating engine fuel systems, and aircraft shock strut and brake systems" (Bendix, 1954). Quite contrary to the "computer designer" role suggested by other ads from around this time, this opening clearly demanded a worker who could effectively *use* computers in the solution of engineering problems. As additional evidence for this interpretation, the ad provided an appropriate disclaimer: "no maintenance ability necessary."

Bendix announcements in 1955 and 1956 for "Senior Computer Engineer" and "Computer Engineer" positions further muddied the waters, as the former called for experience with "analog computers with control applications," and the latter demanded a "digital computer programmer" (Bendix, 1955b; 1956). This occupational nomenclature was further refined in a 1955 Bendix ad that was headlined "Analog Computer Engineers." This posting briefly described three work positions, namely for "Senior Analog Computer Problem Analyst," "Senior Computer Problem Engineer," and "Analog Computer Problem Engineer" (Bendix, 1955a). Once again, the ad revealed that these positions involved wide-ranging job responsibilities and educational requirements, and they were focused on analog rather than digital technology.[102]

A further canvas of the literature reveals that while many companies were using variations of the term "computer engineer" around this time, the aircraft industry led the way. A 1956 ad from Northrop Aircraft, for example, explained that "applied mathematicians and

[102] The first of these positions involved the most extensive experience and responsibility in areas ranging from problem analysis to computer set-up and operation. The latter two openings, however, were focused on computer set-up and operation, suggesting that the company was making rather broad use of the term "engineer." In fact, one of these positions stipulated a degree in math or physics, and the other in math, physics, or electrical engineering.

engineers are needed as computing analysts" (Northrop, 1956). The ad listed a series of more specific job titles, ranging from "computing engineers" and "computing analysts" to "electronics engineers" and "applied mathematicians." And while the use of terms such as "computing engineer" drew on established distinctions between "computers" and "computing," advertisements such as this one were vague about how the listed job titles were linked to specific types of work, expertise, or educational prerequisites. The body text for this same ad also indicated that technicians, electronic engineers, and mechanical engineers were needed for design and development work in Northrop's Computing Center. Companies such as Northrop and Bendix were clearly seeking employees with a wide variety of backgrounds for computer-oriented work. And since more precise classifications for these types of employees had not yet emerged, these companies crafted their own partially unique sets of terminology.

Along similar lines, a 1956 series of announcements from Autonetics – a division of North American Aviation, Inc. – indicated many opportunities for "engineers and scientists" to fill openings as "computer specialists," "computer programmers," and "computer application engineers" (Autonetics, 1956). And Temco Aircraft Corporation announced in 1957 that it was looking for "analog computations engineers" who possessed analysis and programming experience (Temco, 1957). Douglas Aircraft Company, on the other hand, called for those with formal training in mathematics, science, or engineering to work as "expert programmers" or "computing engineers" (Douglas, 1957). While this laundry-list of positions was clearly oriented toward programming and applications, still other openings blurred the boundaries between computer design and use. A 1957 ad from Westinghouse-Baltimore, for example, explained that "Digital Computer Engineers" were needed "[f]or the extensive application of present and future digital techniques to military problems. Applicants with background and interest in digital coding, digital programming, and in the necessary hardware to implement such systems" (Westinghouse-Baltimore, 1957).

The many variations of the term "computer engineer" surveyed here can be accounted for in a number of ways. To begin with, many of these examples are from aircraft and aerospace companies, which by the mid-1950s had established major interests in both the *design* of special-purpose computers for major aerospace projects and the *use* of general-purpose computers for solving a wide variety of engineering problems. Hence, terms such as "computer engineer" were sufficiently flexible to capture professional work involving both the design and application

dimensions of computer-oriented work. And as reviewed above, number of more descriptive and specific variations helped clear up some definitional ambiguities, at least until better alternatives emerged. In addition to the aforementioned examples of "computer problem engineer," "computing engineer," and "computations engineer," other companies invoked terms such as "computer applications specialist" (Hughes, 1954), "computer programmers" (Autonetics, 1956), and "computing analyst" (Northrop, 1956).

On the one hand, these latter terms – which avoided the engineering appellation altogether – appeared with increasing frequency through the 1950s. On the other hand, many of the companies highlighted above were probably using variations of the term "computer engineer" for good reasons. For example, a number of these advertisements clearly pandered to computer-oriented engineers who might otherwise avoid openings that carried titles such as "programmer" or "analyst." Further, engineers were likely a preferred pool of employees for many of these companies, for at least two major reasons. First, there was a growing perception around this time that training an engineer or other specialist how to use a computer for problem solving was far easier than the reverse approach, namely teaching a programmer or analyst the necessary knowledge to undertake domain-specific work in engineering and design. And second, employer demand and average salaries were spiraling upward for computer programmers, operators, and applications experts, and new questions were being raised about both the quality and motivations of the diverse individuals who were taking these types of positions. Engineers, by contrast, were a well-established occupational pool with comparatively high stability, predictability, and homogeneity (i.e., white and male).[103]

A further canvas of employment listings from the mid-1950s onward reveals other relevant trends. First, the use of the term "computer engineer" in its more design-oriented sense

[103] A 1957 ad from Burroughs Corporation provided a rather lengthy description of the company's "ideal" prospective engineer. Headlined "That Certain Man," the ad copy explained that "good engineers" often shared a number of common characteristics: "He's ambitious, he's inquisitive, and if he's still a young man he's been out of college only a few years, has a wife and possibly one or two children. He likes his job and the company he works for … but he's a little restless. He knows he is a good engineer but wants a chance to prove it. In many cases, he's bogged down with too much paper work, – not enough responsibility. Or perhaps doing the job of a trained technician. He needs a change of pace. He needs creative work to still his restlessness and prove his ability. He wants recognition, and a chance to advance" (Burroughs, 1957). Weaving together themes of masculine socialization and identity, this ad stands as a potent example of the "dominant image" of the ideal engineer. The emergent identity of the computer programmer or analyst, on the other hand, was comparatively vague, ill-defined, and of lower professional status.

persisted throughout this period, and by the late-1950s most of the ambiguities evident in earlier ads were starting to fade. A 1955 Republic Aviation ad, for example, complemented its call for an applications-oriented "Senior Computing Engineer" with an opening for a "Computer Engineer" to "supervise maintenance and to design special circuitry for computers" of either the digital or analog variety (Republic, 1955). Notices from National Cash Register (NCR) published in 1956 prominently displayed the terms "Computer Engineers" and "Digital Computer Engineers," as well as more specific associated titles, such as "Senior Electronics Engineers," "Transistor Circuitry Engineers," and "Senior Digital Computer Engineers" (National Cash Register, 1956a; 1956b). According to the copy for the second of these two ads, the "Senior Digital Computer Engineer" position involved "advanced computer design, development, and application," and required a "thorough knowledge of digital computer logic and circuitry, input-output devices, programming" (1956b). Noting that employees would "enjoy the freedom of a small, select research group – operated *by* engineers *for* engineers," the ad hinted at NCR's ideal prospective employee, as well as the company's dominant culture of research and design.

NCR's depiction of the digital computer engineer – which tended to emphasize hardware and design over applications and programming – captured both the formative image and jurisdiction of this emergent professional domain. And from the late-1950s onward, other uses of the term that both drew on and reinforced this image appeared with increasing frequency. "Experienced analog or digital computer engineers" were needed at North American Aviation in 1956 (North American Aviation, 1956), while a 1957 ad from Librascope – a "computers, controls, components" company – called for "Digital Computer Engineers," including in sub-fields such as "Logical Design" (Librascope, 1957). And throughout 1958, Hughes indicated immediate openings for engineers in many areas, including "Computer Engineering" (Hughes, 1958a) and "Digital Computer Engineering" (Hughes, 1959b).

A 1960 ad from Hughes, on the other hand, displayed the labels "circuit designers," "logical designers," "systems analysts," and "programmers" beneath a larger heading that read "digital computer engineers" (Hughes, 1960). In 1960, the Kearfott Division of General Precision, Inc. listed similar types of work under the banner of "Digital Computer Engineers," including "Digital Circuit Design and Development" and "Computer System Synthesis and Logic Design" (Kearfott, 1960). NCR also returned with a 1960 posting for "Digital Computer Engineers" that called for applicants with EE degrees who were experienced in areas such as

logic design, circuit design, product engineering, and systems engineering (National, 1960). The major areas of expertise outlined in these ads – which had been tentatively linked to computer engineering earlier in the decade – were becoming more widely recognized as core domains of knowledge and practice for the field. To put it another way, these advertisements both described and prescribed a preferred educational background for computer designers and engineers, as well as the types of work tasks, bodies of knowledge, and technologies that were within their purview.

Further evidence for these trends can be found in advertisements from International Business Machines (IBM), which through the 1950s emerged as a dominant player in the commercial computer industry. To begin with, many early IBM ads emphasized the role of engineers and engineering in the development of the company's best-known machines. A 1954 ad for "electronic and electro-mechanical engineers," for example, pitched that prospective employees would "be working with the great terms of engineers that created and developed the world's most advanced digital computers" (IBM, 1954). And in 1955, one personnel ad described IBM as an "outstanding engineering organization," and it encouraged applications from "[m]en with BSEE degrees and some experience in design" (IBM, 1955a). And later in the same year another IBM posting carried a headline that read "The Challenge of Creative Engineering" (IBM, 1955b). Text that appeared directly beneath a picture of two men – who were surrounded by electronic equipment, and who were presumably engineers – explained: "At IBM, engineers are continually exploring the frontiers of man's knowledge in the expanding field of electronics." In addition to framing IBM as an engineering organization, these advertisements portrayed cutting-edge work in electronics and computer development as a jurisdiction that was claimed by engineers and engineering.

Through the 1950s IBM also regularly advertised openings for electrical, electronic, and "electro-mechanical" engineers. And while these ads generally did not make general reference to "computer engineers," more specific variations of the term were plentiful. A 1957 ad for the company's military products division, for example, listed openings for "computer circuit design engineers" and "computer logical design engineers," as well as "systems evaluation engineers," "systems engineers," and "systems analysts" (IBM, 1957). Each of these five positions was accompanied by a profile of an existing IBM employee, and all but one of these individuals held

an electrical engineering degree.[104] In 1961 IBM similarly announced immediate opportunities for "Systems Engineers" and "Computer Engineers," as well as for various programmers, analysts, and other specialists (IBM, 1961). As suggested by these ads, IBM followed many other companies in linking the profession of computer engineering with both formal education in electrical engineering and design-oriented expertise in circuits, logic, and systems.

Still other sources suggests that what was happening at IBM was largely the norm in the commercial sector. In a 1959 *Computers and Automation* article on the demand for college-level computer training, Penn State's Frank Hartman concluded that "[t]raining in computer design is at present deemed to be almost entirely the prerogative of the electrical engineers" (Hartman, 1959, p. 13). As additional evidence for this claim, Hartman presented data on the demand for computer-oriented personnel based on survey results from 325 companies (Hartman, 1959, p. 13). With regard to a total of 191 employees who were hired in 1958 with a "[b]ackground in the engineering problems associated with computers (maintenance, modification, design, etc.)," a total of 155 of these individuals (or 81%) had backgrounds in electrical engineering. The vast majority of the remaining employees in this category (34 of 191, or 18%) were trained as mechanical engineers. Similarly, these same companies reported a total of 44 vacancies for individuals who had this same type of background, and it was expected that 43 of these positions would be filled with graduates of electrical engineering programs. While pockets of definitional ambiguity surely persisted, by the early-1960s the assumed educational prerequisites – as well as the professional identity and jurisdiction – of the "computer engineer" had largely stabilized.[105]

Divisions of Labor and Hierarchies of Design: Bounding and Segmenting Computer Engineering

In addition to revealing the historical trajectory of the term "computer engineer" in the commercial sector, many of the personnel advertisements that appeared in the mid 1950s to early

[104] In fact, the exceptional individual on this list did hold an engineering degree, but in the area of mechanical rather than electrical engineering (IBM, 1957, p. 79A).

[105] One pocket of definitional ambiguity was evident in a series of 1960 postings from Philco for "computer engineers." In addition to associating these positions with rather typical types of computer design and engineering work, these ads called for "[e]xperienced computer engineers … to install, start up, and maintain large-scale, high speed digital computer systems" (Philco, 1960). The required qualifications these positions were not entirely clear, but an associated heading that read "Customer Service Engineers" suggested that lower-status technical work was involved. However, my survey suggests that these types of ambiguities were increasingly rare in subsequent years.

1960s period reveal two closely related trends. The first of these involves the various divisions of labor that were emerging and expanding within the domain of computer engineering, especially as computer development became increasingly complex, commercialized, and even routinized. A second relevant trend centers on the ongoing demarcation of computer engineering, design, and related activities from other domains of knowledge and practice. In addition to documenting and analyzing these trends in the context of various worksites, in subsequent sections I follow these themes into other contexts, including the sphere of technology and the domain of education.

As indicated above, 1950s era personnel advertisements for "computer engineers" were frequently accompanied by more detailed lists of subject areas, job responsibilities, and occupational titles. These ads hinted at the extent to which the development of computer systems and related equipment during this time period involved expanding "hierarchies of design," to use terminology developed by Vincenti (1990). One early discussion of these divisions of labor can be found in a 1955 talk by Charles W. Adams, who identified component design, logical design, system design, and the "development of automatic coding techniques" as some of the main areas that fell under the larger umbrella of computer design (Adams, 1957, pp. 139-140).

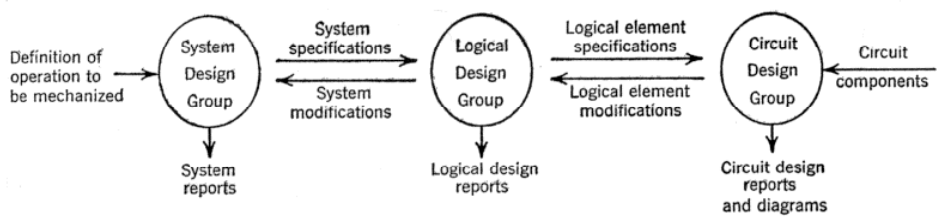

Figure 4.1 – Functions and Responsibilities of Computer Design Groups (Phister, 1958, p. 3)

In his 1958 textbook titled *Logical Design of Digital Computers*, electrical engineer Montgomery Phister worked along similar lines.[106] Early in the book, Phister outlined how computer system design projects were frequently carried out through the cooperative efforts of three major engineering groups, as shown in Figure 4.1. The "system analysis" or "system design" group, to begin with, was largely responsible for developing system specifications in

[106] Phister was well-qualified to write on these divisions of labor. In fact, this textbook grew out of his experiences teaching graduate-level logic design courses to electrical engineers at UCLA in the mid-1950s. Phister's perspective was also enriched by his first-hand experiences as an engineer in the 1950s, including at Hughes Aircraft and Ramo-Woolridge (Phister, 2005).

light of the intended applications of a particular machine (p. 2). The circuit design group, on the other hand, was primarily concerned with using basic circuit elements – such as resistors and vacuum tubes, diodes and wires – to develop various sub-components that could perform specific operations (p. 2-3). And finally, the logical design group was positioned between the other two groups, and was charged with assembling sub-components into larger subsystems and systems in order to realize desired machine specifications and functionality.

While Adams and Phister provide us with a glimpse of the design hierarchies that were emerging within computer design groups in the mid and late 1950s, additional support for these depictions can be found in personnel ads from this same time period. The aforementioned 1957 ad from IBM's Military Products division, for example, showcased the activities of five different types of engineers that roughly fit into Phister's categories. More specifically, these job classifications included: Computer Circuit Design Engineers, Computer Logical Design Engineers, System Evaluation Engineers, Systems Engineers, and Systems Analysts (IBM, 1957). And while this particular ad tended to emphasize the importance of systems analysts and designers, by the early 1960s other companies were framing logical designers as a pivotal part the commercial computer development equation. In fact, Phister explained that "the experienced logical designer is a Jack-of-all-trades," whose expected knowledge often spanned from machine operation, maintenance, and application to understanding different approaches to the design of systems, subsystems, and components (1958, p. 3).

A 1963 Honeywell advertisement, on the other hand, explained that "[c]omputers are born in the mind of the Logic Design Engineer" (Honeywell, 1963). In addition to claiming that logical design work was "engineering in the truest sense," this same ad noted that the responsibilities of the Logical Design Engineer cut across several disciplines.[107] And a 1965 Honeywell employment posting added that "[s]ome of the most challenging engineering being done at Honeywell is hidden behind the job title Logic Design Engineer" (Honeywell, 1965). Yet by 1966, this same company was pandering to circuit and systems engineers in a similar manner. "Computers are realized in the mind of the Circuit Design Engineer" (Honeywell, 1966a), one

[107] It is also worth noting that this Honeywell ad appeared in an early issue of *Computer Design*. Established in 1962, this trade publication was oriented toward both industry generally and digital circuit and system designers specifically. Clearly aimed at practicing engineers, an editorial in the first issued indicated that the magazine would publish articles and reports that would help "bridge the gap between textbook theory and the practical rule-of-thumb principles that guided designers to a successful product" ("Editorial Prospectus," 1962, p. 3).

advertisement proclaimed, while another explained that "[c]omputers are conceived in the mind of the system design engineer" (Honeywell, 1966b). While these statements may initially appear contradictory, each holds a grain of truth. That is, circuit, logic, and system design engineers all played important roles in a design hierarchy that had emerged and coalesced through roughly the first decade of commercial computer development. Further, the evidence presented above suggests that all of these positions were located within the domain of computer design or computer engineering, with each claiming a jurisdictional sub-segment of the larger field.

On the other hand, there were clearly other types of actors who were to some extent involved in computer design, such as programmers, applications specialists, and even end users. This issue points to a second major theme that is evident in many of the advertisements published during this time period, namely the growing divide between two major professional jurisdictions, one focused on design and the other on applications. This tendency was nicely summarized by mathematician Franz Alt in his 1958 textbook: "At least two such fields of specialization have come into prominence: computer machine engineers, concerned with the design, construction, and maintenance of these machines, and the programmers and numerical analysts, who prepare problems for them" (1958, p. v). In fact, by the late 1950s many companies were more frequently making employment pitches that were exclusively aimed at "computer programmers." As a result, ambiguous terms such as "computing engineers" and "computations engineers" were displaced, and "programmers" were increasingly associated with a partially distinct assortment of work locations, skill sets, bodies of knowledge, educational backgrounds, and technologies.

Evidence for this theme can be found in a number of advertisements. A 1959 posting from System Development Corporation (SDC), for example, was explicitly and exclusively aimed at "Computer Programmers." And by 1962, IBM was running a series of appealing advertisements for "Programmers." One of the first ads in this series described programmers as part of a "young but rapidly growing profession," and it went on to note that "programmers are creating new concepts in software, and contributing to the design of new systems" (IBM, 1962). Such advertisements described and prescribed the emergent occupational niche of the computer programmer, just as prior ads had contributed to the establishment of the dominant image and jurisdiction of the computer engineer. However, ads for programmers were often comparatively

vague about educational prerequisites, as evidenced by one IBM ad that simply sought applications from those with "experience in computer programming" (IBM, 1962).

A 1962 Honeywell advertisement provided further evidence for the divides that were growing up between programmers and engineers. A tall vertical pane on the left side of a split-page spread carried the headline "Engineers," and it provided an overview of both the system specifications and performance benchmarks of the company's new H1800 computer system (Honeywell, 1962c, p. 10). The same vertical pane also listed professional opportunities at Honeywell for circuit designers, logical designers, electrical engineers, and product designers. A similar box on the far left of this layout – which was separated from the rest of the ad by two columns of unrelated magazine content – was headlined "Programmers" (p. 11). This panel listed job openings in areas such as automatic programming, operational programming, compiler development, and systems analysis (p. 11). This advertisement provided a potent visual metaphor for the computer field's major sociotechnical boundaries. That is, engineers and programmers were framed as being interested in different aspects of computer technology, and they were sought for distinct types of professional positions.[108]

A further canvas of ads reveals that the major divisions of labor and hierarchies of design outlined above were firmly established by the mid-1960s. Advertisements for computer engineers frequently required electrical engineering or physics degrees, and were often focused on circuit design, logic design, and systems engineering. Programmer positions, on the other hand, often stipulated an education in mathematics or science, and involved work in areas such as numerical and systems analysis, systems and applications programming, and software development. Of course, questions remained about the extent to which existing educational programs provided adequate preparation for these and other types of computer-oriented professional work, a point to which I return below. But before doing so, it is necessary to analyze how the negotiation of these professional and disciplinary boundaries became deeply intertwined with another emergent dichotomy, namely that of "hardware" and "software."

[108] In 1962, the personnel consulting firm Dataman Associates similarly split out its listing for engineers and programmers, each of which appeared in separate advertisements on separate pages (Dataman, 1962a; 1962b).

The Hardware/Software Ensemble: Constructing and Questioning the Dichotomy

On the surface, one might presume that the distinction between computer hardware and software is largely or even wholly a technical matter, especially in light of contemporary, popular uses of these two terms. Yet in this section, I use a discourse-oriented approach to assert that ongoing efforts to both define the meaning of these two terms and delineate their relation was – and remains – a multi-dimensional and sociotechnical process. Drawing on concepts and terminology developed by Paul Edwards, the body of discourse surrounding hardware and software can usefully be viewed as a "heterogeneous ensemble" that melds technologies, social identities, practices, and bodies of knowledge (1996, p. 40). Further, framing hardware and software in this manner helps shed light on their dichotomous yet relational character.

It is first worth briefly reviewing some important background details that were introduced in preceding chapters and sections. While the term "hardware" was first used in reference to computing machinery in the 1940s, the term gained momentum through the 1950s as convenient shorthand for the "physical components of which computers are made" ("Editorial," 1953, p. 1).[109] And as noted above, the jurisdiction and identities of computer designers and engineers became closely linked to the physical machinery of computing, especially in the 1950s. On the one hand, these developments reveal the extent to which the emergence of new professional and disciplinary identities is often sociotechnical, in that can involve intertwined social markers, bodies of knowledge, and domains of technology. Yet this process was also significantly relational, in that it involved the definition and negotiation of interrelated terminology and concepts. Through much of the 1950s, for example, the term "hardware" was frequently juxtaposed with other terms that described the more ethereal "internal" aspects of computers, such as the digital bits and bytes that ultimately comprised all programs and routines.

By the late 1950s, however, commentators such as Paul Armer were complaining that the term "program" and its many variations were overused. As Armer explained, "[O]ur field is badly in need of a new set of generic terms. In particular, we need replacements for all forms of the word 'program'" (1959, p. 3). Snidely adding that "it's even possible these days to discuss 'the Dynamic Programming programming program,'" Armer suggested that this issue might be

[109] According to the *Oxford English Dictionary Online*, the term "hardware" was first used in the context of computing machinery in Douglas Hartree's 1947 book, *Calculating Machines* ("Hardware," 2006; Hartree, 1947).

addressed via a contest or committee. Yet such measures were ultimately unnecessary, as the term "software" surfaced with increasing frequency in the early 1960s. Credit for coining this term often goes to scientist John W. Tukey, who in a 1958 journal article explained:

> Today the 'software' comprising the carefully planned interpretive routines, compilers, and other aspects of automative programming are at least as important to the modern electronic calculator as its 'hardware' of tubes, transistors, wires, tapes and the like (quoted in Shapiro, 2000, p. 69).

While Tukey followed prior commentators by juxtaposing "programming" and "hardware," he attached the former to an appealing new term. "Software" was a clever catch-all expression that captured an increasingly expansive sub-domain of computing, and its general meaning was easily discerned by those already familiar with the term "hardware."

By the early 1960s it was clear that the definition and use of this new term often varied significantly from author to author and text to text. In a 1962 letter, for example, mathematician Bernard Galler noted that the meanings associated with the term "software" were proliferating. He also tried to clarify matters by explaining that "[t]o each user of a computer, the total computing facility provided for his use, other than the hardware, is the software" (1962).[110] As suggested by Tukey's definition, the precise definition of the term was perhaps not as important as its relation to "hardware." Around this same time, still others were predicting the demise of the amorphous term. In an interview published in *Datamation*, for example, IBM executive Warren C. Hume suggested that "the term software – as a catchall word – is going to become less and less meaningful as time goes on" (Bergstein, 1962, p. 35).

Commentators such as Hume clearly underestimated the valuable discursive niche that this term filled. In fact, many other computer companies were eagerly embracing the dualistic discourse of software and hardware. IBM rival Honeywell, for instance, provided prospective customers with a basic definition for software in a 1962 advertisement: "Software is a new and important addition to the jargon of computer users and builders. It refers to the automatic

[110] Further hinting at the flexibility of the term, Galler noted that its meaning could shift as a function of both time and user. He more specifically explained that "[t]o the systems programmers of an installation just receiving a computer, *software* means that which the manufacturer supplies which is not actual hardware. … to a user of that same computer one year later, *software* means the system available to him, including all of the additions to the library, new translators, utility programs, etc., which his systems group has added to the delivered software" (p. 6). As this passage reveals, the meaning of a given technical term can be highly dependent on context, especially when the term tends toward generality.

programming aids that simplify the task of telling the computer 'hardware' how to do its job" (Honeywell, 1962a, p. 46).[111] And in another advertisement published the same year, Honeywell emphasized that "Good software makes good hardware better – and vice versa" (Honeywell, 1962b, p. 2-3).[112] This was a timely pitch, as a growing roster of pundits was pointing out that hardware development had largely settled into stable patterns of ongoing, incremental improvements in reliability and speed, while major advances in the realm of programming were comparatively sparse. Further, proclamations about the so-called "complimentarity" of software and hardware were surfacing more frequently, although they often appeared in tandem with critiques of the software-hardware relationship. I revisit this theme in more detail below.

Other formal definitions for "hardware" and "software" appeared in print in subsequent years, revealing the extent to which these terms had quickly become a widely recognized part of the computing lexicon. For instance, a glossary that was originally developed by an ACM committee and published in abbreviated form in the *CACM* put forward these definitions:

> hardware
>
> > The physical equipment such as the mechanical, magnetic, electrical and electronic devices from which a computer is fabricated; the material forming a computer, as distinct from the routines. Contrast with: *software* (Fritz, 1963, p. 155)
>
> software
>
> > The totality of programs and routines used to extend the capabilities of computers, such as *generators, compilers,* and *operating systems.* Contrast with: *hardware* (Fritz, 1963, p. 157).

Once again, hardware and software were defined as counterparts, with the former referring to the "material" or "physical" aspects of the machine, and the latter representing the associated "internal" programs and routines.

In the early 1960s, the trade magazine *Datamation* also picked up this new terminology and ran with it. The always provocative Herb Grosch, for instance, authored a 1961 editorial titled "Software in Sickness and Health." And a 1961 survey article on computer components

[111] This same ad also identified three sub-categories of software, namely assembly systems, compiler systems, and operating systems (pp. 46-47).
[112] The ad went on to explain that "Honeywell software is designed to capitalize on, and complement the advanced capabilities of Honeywell hardware. Each extends the power of the other" (p. 3).

suggestively juxtapositioned the terms "software" and "hardware" in order to comment on the computer field's major boundaries, which were framed as neither simply nor merely technical:

> Although there is considerable mutuality of concern in their ultimate objects, 'the advancement of computer technology and application,' hardware personnel and their software peers have long been widely separated by geography, education and interest, and all that is written and said has not as yet made one head out of Humpty and Dumpty. … Perhaps, when the seemingly insurmountable hurdles are charged for the last time, it may suddenly appear that Humpty Dumpty is after all, a single entity and must be fitted properly together to continue sitting high on the wall (p. 36).

While it is not clear whether the "Humpty Dumpty" analogy used in this passage referred to computer systems, computer-oriented workers, or perhaps even the computer field as a whole, the allusion was particularly effective because it hinted at the full range of "sociotechnical" dynamics that were in play at the time. That is, this editorial remark linked two general spheres of technology – denoted by the terms "hardware" and "software" – with two distinct classes of computer professionals, who often worked in different locations and possessed different educational backgrounds and interests. [113] Yet the editors also challenged these boundaries by hinting at the benefits of somehow unifying or integrating these two sociotechnical spheres.

Many period advertisements from major computer companies similarly invoked the software/hardware schism while simultaneously calling it into question. A 1962 ad from Burroughs Corporation, for example, queried: "When will a computer manufacturer design a system so that hardware and software – including operating system, programming languages and compilers – are completely integrated?" (Datamation, August 1962, p. 14).[114] The 1964

[113] In a 1961 *Datamation* editorial, Grosch worked in similar directions when he noted that "a few miles between software and hardware boys is healthy, but a hundred is too much" (1961, p. 33). While it was clear that Grosch was referring to the increasing geographical distance between software and hardware experts – who were increasingly segmented in different corporate divisions and even different companies – it also suggested a more general schism between the domains of software and hardware.

[114] On a closely related note, a 1959 advertisement from System Development Corporation (SDC) carried a large headline that queried: "Computer Programmers: Seen any new horizons lately?" (System Development Corporation, 1959). The ad copy outlined a number of major SDC research projects, one of which centered on the "investigation of computer design from a standpoint of programmability rather than engineering." In addition to revealing the ongoing expansion of a divide between the perspectives of computer programmers and computer programmers, this ad once more suggested that the computer industry was trying to design computers that were more responsive to end-user needs and applications.

advertisement from Mesa Scientific Corporation shown in Figure 4.2, on the other hand, even more suggestively declared that "Mesa Men now come in two convenient types: Software… and Hardware!" (Mesa Scientific Corporation, 1964). In addition to calling attention to a perceived gulf between these two domains, the ad emphasized the ability of Mesa's "integrated software/hardware team" to "reduce software/hardware interface problems" and "optimize software/hardware trade-offs." While this passage echoed other calls from around this time for more "integrated" approaches to computer development, it also forcefully revealed the extent to which the boundaries around software and hardware were as much about "men" as they were about machines. In addition, the rhetoric presented in these ads revealed a fundamental tension, namely that if software and hardware were truly integrated, the terms might not be needed.

"Software" and "hardware" also appeared with increasing frequency in conference programs and in the remarks of leading figures in the computer field. At the FJCC in 1963, for instance, separate panel sessions were dedicated to "Software for Hardware Types" and "Hardware for Software Types." These events, which were reportedly well-attended, hinted at the extent to which the discourse of software and hardware were becoming closely linked to pre-existing professional and disciplinary identities, worksites, bodies of knowledge, technologies, and cultures of design.[115] Along similar lines – and as noted in the previous chapter – in 1964 Willis Ware suggested that the IEEE largely represented the "hardware population" of the computer field, and ACM the "software population." Yet as suggested by many of the examples cited above, others were questioning the apparent and ongoing tendency of the field to cleave into two major parts. In the section that follows I take a closer look at some of these more critical alternative perspectives.

[115] A pre-published description of the "Software for Hardware Types" session was especially revealing. It noted, for instance, that "[t]he role of programming and the programmer in the computer field is growing rapidly in recognition and importance but is still widely or poorly misunderstood" (53). The same overview noted the historical tendency for programmers to hold a "second class status" to hardware people, and they emphasized that programmers were becoming more widely accepted as "partners" in the planning and design of computers.

Figure 4.2 – "Mesa Men" (Mesa Scientific Corporation, 1964)
Advertisement Furnished Courtesy of Northrop Grumman Corporation

Artificial Barriers versus Integration: Carr and Gorn on the Boundaries

As noted in the preceding chapter, by at least the late 1940s a handful of commentators were calling for improved approaches to the development of computing machines and systems. In a 1949 conference presentation, for instance, Jay Forrester advocated additional research in the area that he called computer "systems engineering." Still other writers – such as Lehmer, Mauchly, and Hopper – extolled the benefits of close cooperation and open communication between computer designers and programmers. As these authors argued, increasing the cross-talk between these two groups could lead to relatively small design changes that would greatly improve the functionality and usability of computing machines. Yet these types of calls for cooperation were largely confined to a small circle of thoughtful critics, many who happened to lack engineering credentials. In the present section I focus on John W. Carr III and Saul Gorn as two important actors who helped carry this tradition of critique through the 1950s and into the 1960s, especially as they discussed the justifications for – and implications of – the computer field's major sociotechnical divides.

I begin with Carr, who in the 1950s surfaced as an outspoken proponent for expanded university involvement in computer-oriented research and education, especially in areas such as computer design. After earning a Ph.D. in mathematics at MIT in 1951, Carr spent much of the 1950s as a professor and research mathematician at the University of Michigan (Lee, 2001). From 1959 onward, he assumed a variety of academic posts at both the University of North Carolina and the University of Pennsylvania's Moore School of Engineering. On the surface, Carr might appear a somewhat unlikely commentator on the topic of computer design, especially given his background in mathematics and computer applications. However, Carr's research interests and experiences provided him with a nuanced understanding of the computer field's evolving social and technical landscape. In fact, he stands in a longer line of mathematicians and programmers whose in-depth familiarity with the first generations of computing machines provided them with the ability to insightfully comment on and critique the contemporary state of computer design and engineering.

Early evidence for Carr's engagement with these types of issues can be found in the Proceedings of the first JCC, which included a summary report by Carr on a series of hastily-convened conference sessions that were focused on various "problems of programming" (Carr, 1952, p. 113). These short reports suggest that the meeting sessions provided opportunities for

Carr and other participants to discuss the evolving relation of machine design and operation at an event that was ostensibly and more narrowly focused on the "engineering aspects" of computer design and construction. Carr went on to engage with many related issues at the EJCC 1956, where he took advantage of his role as "conference summarizer" to develop a rather forthright commentary on the contemporary state of education, research, and employment in the computer field. More specifically, he highlighted three intertwined problems. The first of these centered on "the problem of manpower" (Carr, 1956, p. 147). In light of impressive growth in both the total population of computers and the number of different system models, Carr asked: "Where are the people to come from who will develop, maintain, and use these new monsters, devourers of both information and personnel?" (p. 147). This issue was receiving a great deal of attention around this time from a growing roster of commentators, and I discuss some of their proposed solutions in more detail below.

A second and closely related problem discussed by Carr centered on the "preservation and rehabilitation of the universities in the area of computer circuits, design, and logic" (p. 147). On the one hand, Carr acknowledged that a handful of schools remained active in computer design research, yet he described these as "isolated cases with tenuous futures" (p. 147).[116] As further evidence for this claim, Carr noted that the EJCC at which he was speaking featured few presentations from university researchers, which was a marked change from prior joint conferences. And at another point in his talk, Carr complained that high salaries in industry were luring many professors and graduate students into the commercial sector, and he suggested that government funding for university research in computer design and development was being neglected, especially in areas such as "over-all systems design" (pp. 147-148). In light of these challenges and trends, Carr suggestively asked, "In the area of computer design, are we letting the wells run dry at the source?" (p. 147). He also warned his audience: "When university research in computers disappears, university teaching in that area crumbles" (p. 148).

The third major point of concern discussed by Carr centered on the so-called problem of "intercommunication." As the author explained, various "artificial barriers" were isolating computer users from designers, as well as "logical program designers" from the "logical hardware designers" (pp. 147-148). Carr claimed that this problem was more serious than ever,

[116] More specifically, Carr identified computer design and development activities that were being carried out at Purdue, the University of Michigan, the University of Pennsylvania, and the University of Illinois.

as evidenced by the weak coverage of programming topics at the very EJCC at which he was speaking (pp. 147-148). And while these types of concerns clearly echoed the prior comments of Muachly and Hopper, Carr pushed into new territory when he asked: "How is a discipline organized so that it can intercommunicate?" (p. 148). In response, Carr framed the computer field as a single discipline, and he emphasized potential commonalities and possible points of contact between hardware designers and programmers.

Such comments might make it look like Carr was out of touch with existing commercial and professional realities. Yet he went on to argue that universities and their associated personnel and students could play a pivotal role in ameliorating all three of the major problems identified in his talk. He noted that university professors, for example, tended to approach the task of intercommunication as a "labor of love," and he explained that they frequently "pass the discipline on" through various activities, including through the development of various textbooks, glossaries, and "treatises" (p. 148). Carr also suggested that university professors and researchers were uniquely positioned to act as both critics of existing computers and sources of imaginative new machine designs, even if they lacked the resources to build their own components, much less entire systems (pp. 147-148). And finally, he argued that establishing new "professional" graduate programs could provide the types of employees that the marketplace was demanding.

Toward the end of his talk, Carr discussed some of the specific ways in which the computer field's extant social and technical boundaries might be blurred. In the technical sphere, he described how cutting-edge computers such as the Univac-Larc and IBM STRETCH were being designed as "integrated systems," from the "outside in" (p. 149). As Carr explained, this "integrated systems approach" took "the external language of communication as the starting point," and used "automatic programming techniques in carrying the language into the middle of the machine" (p. 149). This alternative model of computer design – which was driven by applications and higher-level programming languages – was quite unlike the mode of computer development that had become dominant in the commercial sector, where manufacturers were increasingly adept at building faster and more reliable general-purpose, stored-program computers, while failing to realize more significant or imaginative changes in overall machine design and functionality.

With regard to the social aspects of the field, Carr revisited the issue of intercommunication. He more specifically recommended the organization of small meetings that brought together diverse types of computer-oriented workers. As Carr explained, events composed of roughly ten to thirty persons might be scheduled as complements to larger meetings such as the JCCs. Further, he suggested that these small meetings could potentially transcend extant organizational and occupational boundaries by bringing together "logical designers with programmers, circuitry personnel with automation specialists, language specialists with programmers, and so on" (p. 150). It is no stretch to describe Carr's remarks as an argument for the "integration" of the field's social and professional spheres, just as he had described and championed "integrated" approaches to computer system design.

When Carr took over at the President of the ACM in 1956, he used his inaugural address to hint once more at the theme of intercommunication. As Carr explained:

> The A.C.M. stands as a common meeting ground for a variety of interests. … We must continue to interpret our many interests one to another – administrators to mathematicians, programmers to logical designers, educators to members of industry – all linked by this common use of a remarkable set of machines of which we have not yet seen the final limitations (Carr, 1957, p. 7).

While Carr's rhetoric was rather optimistic, the preceding chapter revealed that the ACM continued to tilt toward the needs and interests of only some of these factions – namely mathematicians, programmers, and educators – through much of the 1950s and into the 1960s. The IRE and AIEE, on the other hand, already key centers of activity for large numbers of computer engineers and "logical designers," as well as many members of industry.

Through this same time period mathematician Saul Gorn of the University of Pennsylvania's Moore School of Electrical Engineering also called into question the computer field's major boundaries, although he placed particular emphasis on the relation of "machines" and "programs." In a 1958 letter that appeared in the newly-established *CACM*, for example, Gorn noted the "equivalence of hardware and programming" (p. 2). And in a 1959 conference paper pre-print, he similarly explained:

> The point of view expressed in this paper makes more tangible two principles accepted intuitively by many programmers and logical designers. They are a) the

> equivalence of formal languages and machines, b) the equivalence of programming and hardware (Gorn, 1959, p. 25-1).

Yet despite this hypothetical equivalence, Gorn acknowledged an important associated design question, namely: "how much [structure] should be in the hardware and how much the job of programs?" (1958, p. 3).[117] Indicating his preference for more flexible machine structures, Gorn provocatively added:

> Since it is a user's world, the combination of machine and compiler is the "machine" we are really interested in. The designers of automatic coding systems must therefore be considered among the machine designers, and should be involved before the hardware designers have finished their plans. The pioneers in automatic coding were well aware of the identity of compilers and machines. Others need constant reminding (1958, pp. 3-4).

Gorn's argument for explicitly including the development of compilers within the province of the "machine" clearly challenged the dominant position of engineers as vanguards of computer design. Further, his remarks suggested that shifting from a "machine-oriented" to "user-oriented" perspective might demand accompanying revisions in the computer field's major social and technical boundaries.

As I discuss in the following chapter, Gorn's views on machine-program equivalence clearly informed his early efforts to champion a new discipline that he dubbed the "Computer and Information Sciences." But for the present analysis it is worth returning to Carr, who resurfaced in the 1960s with an updated critique of the computer field and its technological state of the art. In a 1962 article titled "Better Computers" – which appeared in both in an early issue of *International Science and Technology* and in the German journal *Elektronische Rechenanlagen* ("*Electronic Computers*") – Carr started with a pointed assertion: "Today's mass-produced general-purpose digital computers are being designed and used almost wholly without imagination" (1962b, p. 157).[118]

---

[117] As noted in Chapter 2, this particular line of questioning had already been in play for at least a decade. As Mauchly noted in 1948, for example, "A decision must be made as to which operations shall be built in and which are to be coded into the instructions" (p. 205).

[118] It is worth noting that Carr's critique appeared in two publication outlets that were relatively obscure and marginal at the time, at least for U.S. readers. However, it is not clear whether Carr ever tried to publish his this article in a more mainstream professional publication.

Yet Carr also acknowledged the efforts of the early computer pioneers, whose "early daring" helped lay the foundations for the field. In fact, he credited "electrical engineers and physicists" such as Aiken, Wilkes, Eckert, and Forrester for successfully "modeling in hardware" the theoretical concepts that had been developed by various "mathematicians and philosophers," including Turing, Mauchly, von Neumann, Goldstine, and Burks (p. 157). While this type of historical framing was certainly oversimplified, it framed computer development in hierarchical terms, where engineers and scientists realized the ideas of mathematicians and philosophers. Further, this type of characterization was strategic in that it implicitly bolstered Carr's legitimacy as a critic of computer design, given his own background as a mathematician.

Much of the remainder of the article was dedicated to critiquing the computer field for failing to move beyond a general framework of machine design and application that originally developed in the 1940s. More specifically pointing to the problem of "designer conservatism," Carr complained that computer users were "restricted almost completely to the original limited concepts of problem-solving capabilities bestowed on the machines by their designers, rather than a more global view of the problem" (p. 158). Suggesting that this problem was exacerbated by the tendency for computer designers and programmers to be working in very different physical locations, Carr added:

> Hardware specialists often propose solutions to important technical problems which involve a relatively small effort by the logical designer, but leave the bulk of actual implementation to the programmer. These men [sic] may never have met, and probably don't even belong to the same organization (p. 159).

Such complaints strongly echoed many of Carr's earlier remarks. Yet in this particular paper, the author also stepped forward with a critique of programmers. He argued, for example, that the "vested interests" of programmers often led them to oppose changes in machine configuration that might threaten their job security. In other words, making machines easier to use might eliminate much of the detailed analysis and coding work that was at the heart of the programmer's occupational niche. Once again, the problems identified by Carr clearly involved intertwined social and technical factors, ranging from the dominant model of computer design to the differing interests and worksites of various computer professionals.

In a 1965 article published in *Computers and Automation*, Carr revisited and extended many of these same themes. Gorn's influence on Carr is also evident in this piece. In addition to

referencing a 1961 article by Gorn on the topic of "Mechanical Languages," Carr repeatedly trotted out phrases such as "machine-programmed systems" (p. 15) and even the "man-machine computer combination" (p. 15). And later in this same article he critiqued both computer programmers and designers for failing to recognize that "programming is *equivalent* to (not 'analogous to' or 'similar to') building a machine, and not only that, to building a machine in a certain orderly fashion" (p. 16, my emphasis). Invoking a rather suggestive metaphor, Carr echoed his prior writings when he noted that future research and development was needed to meet the challenge of combining "stored algorithms (programs) and equipment algorithms (machines)" into a more "organic" whole (p. 17). He also issued complaints about problems of "intercommunication," although he placed particular emphasis on the gulf that often separated theories of computer programming from its actual practice.

In the early and mid 1960s, other commentators were raising related issues about the expanding gulf between programmers and machines. In a 1961 editorial, for example, Robert L. Patrick noted ongoing and dramatic improvements in the reliability and speed of computer hardware (Patrick, 1961). However, he complained that "it appears as though the hardware types are outstripping the programming types," and he added: "[W]e have no new senior, machine oriented, programmers coming along (due to the emphasis on higher level languages)." For Patrick, this trend was especially problematic for computer installations, where diagnosing and troubleshooting machine faults and "bugs" required types of expertise that were in short supply. And five years later, this same writer trotted out a similar complaint:

> In the last few years we have begotten a whole new generation of programmers who have never come into intimate contact with a machine. They have programmed in a higher order language and have been insulated from the hardware by a solid phalanx of operations managers, machine operators and monitor programs (Patrick, 1966).

In the remainder of this editorial, Patrick expressed further concerns that many programmers were developing software that was simply not in tune with the characteristics and capabilities of particular machines, and he claimed that performance often suffered as a result. And in the same year, a short and humorous *Datamation* piece further hinted at the extent to which good programming practice remained wedded to an in-depth understanding of various facets of

machine design. In his list of "Thirteen Programming Paradoxes," writer Peter D. Jones quipped: "The world's best programmer is also the world's top computer engineer" (Jones, 1966).

In 1962, Christopher J. Shaw came at this issue from a somewhat different angle in a short editorial that was aptly titled "Programming Schisms" (1962). After somewhat nostalgically noting that computer programmers and designers had often been one and the same in the early days of computing, Shaw went on to note the long trend toward specialization in the field. He placed particular emphasis on the growing schism between "the system programmers – who must tame the beast the computer designers build – and the applications programmers – who must then train the tamed beast to perform for the user." The author also predicted that computer designers and systems programmers "will probably amalgamate into one, fairly homogeneous professional group," while more user-friendly computers would contribute to the replacement of applications programmers with domain-experts working as so-called "problem specialists." The author explained that such changes would "bring the computing profession back almost to its pristine beginnings, back when there was only one professional type: the all-around, computer specialist."

While reforms advocated by commentators such as Shaw and Carr may appear overly romantic and idealized, they clearly tapped into widespread concerns about the computer field's major sociotechnical schisms. Cultivating more "integrated" or "organic" approaches to developing computer systems, applications, and even experts was therefore an increasingly appealing prospect in the early and mid-1960s. Still others were calling more specifically for radical innovations in computer structure, especially in terms of moving beyond the decades-old "von Neummann" style of the stored-program machine design. Engineer Lowell Amdahl, for example, used the phrase "gothic computer architecture" to describe the state of the art circa 1965, while Franz Alt complained: "What revolutionary changes in computer design are ahead? Unorthodox answer" (Amdahl, 1965; Alt, 1965, p. 11).

Yet the barriers that stood in the way of realizing such visions were formidable. Amdahl, for example, noted tendencies in the field toward "security and complacency." And while neither he nor Alt clearly identified the underlying reasons for these trends, there were certainly commercial and competitive reasons for maintaining the historical status quo, especially given the dominance of a few big computer makers such as IBM. In fact, Alt noted that innovative machines such as the Univac LARC and IBM Stretch were not commercially successful, leading

manufacturers to retreat back to more conventional designs. Other pressures were also likely afoot. Major computer users in the military and business sectors, for example, have at times tended to preference reliable, standardized designs over more radical and potentially destabilizing innovations. In fact, Steven Usselman nicely summarizes that the history of American computing has been regularly punctuated by periods of relative stability, where "[t]he potential for more rapid and more radical change at the technical frontiers of the industry has been sacrificed in exchange for the perceived benefits of standardization" (1996, p. 30). As Usselman further elaborates, these tendencies were historically enabled by a distinct American political economy, as well as through the intersecting actions and policies of computer manufacturers, the government, and end-use consumers of both the commercial and military stripe.[119]

Returning to the subject at hand, the rise of both the "hardware/software ensemble" and the designer-programmer schism from the mid-1950s to mid-1960s also revealed the extent to which the computer field was increasingly pervaded by major sociotechnical dichotomies that reached deeply into worksites, professional societies, and even technology itself. Yet many of the aforementioned commentators hailed from the academy, where the social and technical boundaries of computing were not quite so sharp, and perhaps easier to call into question. In addition, many writers such as Carr argued that universities in general and computer-oriented faculties and departments in particular could help stimulate the development of a new generation of computer professionals who possessed a less myopic outlook, and who could produce more innovative technologies.

In his 1965 article, for example, Carr discussed how various curricular developments and reforms might provide students with a more in-depth understanding of the inter-relationship of computer programs, machines, and even the so-called "man-machine interface" (pp. 17, 54). As I discuss below, Gorn emerged in the 1960s as a champion for the emergent computer-oriented discipline that he called the "Computer and Information Sciences." Yet these and many other reformers likely underestimated the extent to which both the dominant structure of the computer market and other types of sociotechnical dichotomies – such as engineering versus science or

[119] As evidence for these themes, Usselman places particular emphasis on the initial emergence and growing dominance of IBM in the commercial computer market, including through the 1950s and 1960s.

even profession versus discipline – stood as formidable barriers on the way to realizing a more integrated or unified field of computing.

Part II – Education and Discipline: (Re)Negotiating the Boundaries of Computing

In order to frame the emergence of a variety of discipline-building projects in the computer field, the second part of this chapter takes another step back to analyze the historical development of computer-oriented educational programs, especially from the early 1950s to early 1960s. I place particular emphasis on the role of electrical engineering departments, which were a major source of formal training for the first generations of computer designers and engineers. My analysis helps set the stage for tracing out the efforts of a growing cadre of actors to establish a variety of computer-oriented departments and degree programs at various universities, especially from the late 1950s forward. And as suggested by my discussion of historical trends in other contexts of activity, the outcome of these efforts were largely suspended between two poles of possibility. On the one hand, the educational sphere was potentially a site where the sociotechnical dichotomies that had come to pervade the workplace and the professional societies would be reproduced. In fact, the bifurcated jurisdictional claims and expanding divisions of labor that increasingly came to characterize the commercial computer field in the 1950s suggested that this type of reproduction was an altogether likely scenario.

On the other hand, the prospect of developing a more "integrated" or "organic" approach to the training of computer-oriented professionals was an appealing alternative vision for many actors, especially as they pondered how this might stimulate more imaginative computer designs or new types of applications. Might the educational arena emerge as a setting where the Humpty and Dumpty of hardware and software could be put back together again? While this question clearly transcends the bounds of this chapter, the analysis that follows begins to reveal the pivotal importance of the educational arena in ongoing efforts to (re)negotiate the computer field's major social and technical boundaries.

Computer Education: An Inchoate Early Assortment of Courses and Curricula

The development of computer-oriented courses and curricula can be traced back to the earliest days of the field. As documented by Aspray (2000) and noted in the preceding chapter, by the mid-1940s five major universities were significantly involved in computer development

and research, namly Columbia, Harvard, MIT, the University of Pennsylvania, and Princeton. By the early 1950s, research and educational activities at these and other schools were frequently focused on topics related to computer design and construction, although there were also a number of pockets of interest in application-oriented subjects such as numerical analysis. Yet from the mid-1950s and into the 1960s, questions about how to train or otherwise educate a wide range of experts in computer applications, programming, and related areas emerged as a pressing concern. In fact, this period was marked by the rapid expansion, diversification, and commercialization of the computer field, and few could ignore the rising demand for workers with many different types of computer-related expertise.

The aforementioned John W. Carr III authored one of the earliest commentaries on the computer field's looming employment and educational challenges. Titled "Who Will Man the New Digital Computers?" and published in the upstart trade magazine *Computers and Automation* in 1953, Carr started his brief article by explaining that the computer field was entering a stage of "runaway growth," and he noted an "immediate and pressing need for people to man the machines" (Carr, 1953, p. 1). As suggested by these remarks and the article title, the author's primary focus was on both the demand for and education of the various "engineers, mathematicians, and associated trained technical personnel" who were needed to operate and maintain a rapidly growing number of computer systems and installations (p. 1). Carr also noted rising demand for trained professionals in the area of machine design and construction, yet this was clearly a secondary issue for the author.

In addressing the paper's central topic, Carr identified some of the major types of computer training that were either in use or in development, including short courses offered by computer manufacturers, on-the-job training at computer installations, and intensive summer courses offered at universities (pp. 1-2). But Carr ultimately emphasized the value and importance of a fourth type of training, namely "regularly scheduled graduate and undergraduate courses and programs in universities" (p. 2). He added that such programs could produce "a steady stream of mathematicians, computer engineers, assorted scientists, accountants and business school graduates, all trained in several or many aspects of automatic digital computers" (p. 2). On a closely related note, this same article reveals Carr as one of earliest advocates for the establishment of graduate-level programs that would "give specific degrees in computation or else to give degrees in older fields with specialization in the use or design of computers" (p. 3),

and the author added that mathematics and electrical engineering departments might play a leading role in such programs. For Carr, the former were optimally positioned to teach subjects such as numerical analysis, while the latter were best able to focus on "machine design, logic, and construction" (p. 3). The topic of "machine programming," on the other hand, was described as an area that frequently involved several different kinds of experts, including mathematicians and engineers. Here we find an important early attempt to map some of the computer field's major domains of knowledge and work onto the pre-existing disciplinary structure of the academy. Further, the authors' boundary-spanning depiction of programming revealed some of the difficulties that came with such a project.

A host of closely related themes received considerable attention at the First Conference on Training Personnel for the Computing Machine Field, held at Wayne University in June of 1954 (Jacobson, 1955a). With co-sponsors including the ACM and the Detroit chapter of the IRE-PGEC, the event brought together an impressive array of individuals from the commercial, governmental, and educational sectors.[120] The chairman of the meeting, Wayne University mathematician Arvid W. Jacobson, explained that the idea for the conference went back at least three years, when it was first becoming apparent that developments in the computer field were bringing about fundamental changes and challenges with regard to education and employment (Jacobson, 1955b). He added that a major goal of the gathering was to "find out about the manpower requirements of all areas relating to automatic computing and data processing" (Jacobson, 1955c, p. 3). In addition to assessing prerequisite skill levels and overall demand in this new employment sector, Jacobson stated that the participants at the event were charged with reviewing existing training programs and probing the relationship between educational needs and existing curricula (1955c, pp. 3-4).

As Jacobson explained, the computing machine field embraced "many basic sciences and all manner of practical arts" (1955c, p. 3). Yet other attendees followed Carr by specifically emphasized the role of electrical engineering and/or mathematics departments in computer research and education. The aforementioned Harry Huskey, for example, based his conference remarks on one of the first surveys of university computer education, which was conducted by

[120] Attendees included many well-known computer pioneers and personalities, including Howard Aiken, Franz Alt, John Brainerd, Grace Hopper, Alston Householder, Harry Huskey, and John Mauchley, to name a few (101-104).

the IRE PGEC in 1953 and 1954 and first published in 1955.[121] While the original survey reported that at least ninety schools had some kind of computer facilities or associated courses, Huskey summarized that roughly thirty of the responding universities were offering "regular" training in analog and/or digital electronic computers, mostly at the graduate level (Huskey, 1955, p. 23). Further, he explained that electrical engineering departments were the principal site for this activity, with a total of twenty-nine EE departments offering one or more classes, and nine offering three or more classes (p. 24). Mathematics departments were a distant second in this regard, offering one or more computer-related courses at a total of ten different schools.[122]

Huskey went on to note that a common pattern had emerged at many institutions, where early classes in computer design and closely related subjects were followed by the development of new courses in computer application and use. This trend allowed electrical engineering departments to assume a prominent role in the early development of computer education. Yet given that the interests of electrical engineering faculty and departments tended to skew toward particular subjects, mathematics and other departments stepped in to both fill in gaps in coverage and move into new areas. In fact, Huskey noted that math department offerings were largely clustered around topics such as numerical analysis, logical design, programming, and applications (p. 23). Hence, the educational sphere was beginning to look like another site where the computer field's emergent sociotechnical dichotomies were being reproduced, at least in part.

Further, much of the 1954 conference reflected these divides in that it was significantly tilted toward the development of research activities and educational programs in rapid-growth areas such as computer programming and applications. Throughout the event, discussions about the training of machine programmers and operators frequently overshadowed relatively sparse references to educating computer designers and engineers. Surveying the conference proceedings reveals at least three major reasons for this orientation. First, presenters such as G. T. Hunter of IBM claimed that there were relatively few employment slots in the initial planning and design of computers and other electronic systems (Hunter, 1955, p. 17). He also noted that workers in

[121] Huskey was closely affiliated with the IRE-PGEC at the time, especially through his position from 1953 to 1957 as the Review Editor for the IRE's *Transactions on Electronic Computers*. The survey results that Huskey referenced in his talk were also presented by Goode (1955) in the *IRE Transactions on Electronic Computers*.

[122] It is also worth noting that regular computer courses were being offered outside of the math and EE departments at a total of eight schools (Huskey, 1955, p. 24). The original survey also indicated that eight schools were offering advanced degrees in computing, and eleven had computer-oriented assistantships (Goode, 155, pp. 50-51).

these areas were not coming from computer-oriented educational programs, but were rather moving up through the ranks of industry (p. 17). As Hunter explained, the benefits of this career pathway centered on the range of experiences that it provided for up-and-coming designers. This line of reasoning implied that formal educational needs in the specific area of computer design and engineering were minimal.

A second and closely related theme discussed by Hunter centered on the idea that existing electrical engineering programs provided an adequate level of preparation for a wide range of professional work in computer design, manufacturing, and maintenance (p. 17). Echoing this argument, conference participant James L. McPherson of the U.S. Bureau of Census framed computer design and maintenance as the "engineer's side" of computer equipment, and he added that "[e]ducation in electronic engineering has been, and will continue to be, the way men capable of fulfilling this need are developed" ("Manpower Needs," 1955, p. 33). And mathematician Albert A. Bennett argued at the same conference that it was undesirable for engineering students to focus too narrowly on electronic computers and associated technologies: "The engineering of design, servicing, and testing an electronic machine is too specialized to be a typical topic for the candidate for a degree of Bachelor of Science in Engineering" (Bennett, 1955, p. 41). Here we find noteworthy parallels with the historical development of other engineering subfields and specialties such as "radio engineering," where general types of training in electrical or electronics engineering often prevailed over educational programs that were more narrowly focused on a specific domain of technology.

Casting a wider net reveals that many engineering educators followed Bennett's conservatism with regard to engineering education. In a talk at the 1955 WJCC, for example, Cal Tech engineer F. C. Lindvall spoke on the topic "Computers Challenge Engineering Education" (1955). As Lindvall explained, many colleges of science and engineering were being "urged to offer courses in computer fundamentals, logic, design, components, applications, and use, not to speak of complete curricula leading to degrees in computer engineering" (p. 41). And while he acknowledged that schools with particularly strong research agendas in various areas of computing might be justified in developing such offerings, Lindvall's larger message was far more cautious. He actively argued against "detailed specialization" and "specialized training" in new and emerging areas of interest such as computing, and he instead promoted an educational agenda that was grounded in basics, fundamentals, and generalizations (p. 41).

At the EJCC in 1956, mathematician H. T. Engstrom's keynote remarks suggested that Lindvall's position was widely held, including by many university administrators. Referring to the development of early centers of computer research at schools such as Harvard, Princeton, and the University of Pennsylvania, Engstrom explained that "[u]niversity management was not convinced, and in some cases still remains unconvinced, that the field of logical structure design of computing devices was one with proper academic stature" (Engstrom, 1956, p. 3). Such concerns about the academic legitimacy of computer-oriented research and education clearly stood as formidable barriers for those who favored the development of courses and programs in the area of computer design and engineering.

And finally, the Wayne University conference revealed a third challenge to the ongoing development of courses and programs in the area of computer design and engineering. In summary, there was growing sentiment by the mid-1950s that improvements in computer design and performance were largely outpacing progress in programming and applications. As summarized by conference participant W. H. Frater of General Motors, "Our troubles, at the moment at least, are not the mechanical or logical design of equipment. We already have equipment which we cannot fully exploit" (Frater, 1955, p. 22). And Ralph E. Meager – who at the time was serving as both chief engineer of the Digital Computer Laboratory at the University of Illinois and editor of the IRE's *Transactions on Electronic Computers* – worked in similar directions in a paper given at a 1956 symposium that was organized by the IRE. As Meager explained, "the computer engineers have a tendency to feel now that the main job in computer design work has been completed, that the basic ideas are known" (quoted in "Symposium on the Impact," 1956, pp. 147-148).

As noted above, fundamental innovations in the area of computer design did seem to be leveling off through the 1950s, especially as computer research and development activities became increasingly commercialized and routinized. Further, the general challenges and educational demands associated with computer programming and applications were clearly rising in urgency and importance. Yet many commentators argued to the contrary that research and development activities in the area of computer hardware and systems sorely needed fresh sources of imagination and innovation. Meager, for instance, called on engineers to adopt a "far-reaching attitude," especially in ongoing efforts to develop "integrated systems" (p. 148). Such remarks

strongly echoed Carr's aforementioned concerns about the state of computer design during this period.

Valuable additional commentary on the state of computer-oriented research and education also surfaced in a 1956 article by mathematician and ACM President Alston Householder that was appropriately titled "The Position of the University in the Field of High Speed Computation and Data Handling." After summarizing the prominent early role of universities in computer development and construction, the author noted that it was increasingly feasible for universities to acquire computers rather than build them. "In some cases," Householder explained, "[computer] construction may be worthwhile as a research or a training project. But it seems to me that in the future the universities can contribute the most in applications and training" (Householder, 1956b, p. 8). Householder added that the shortage of technical manpower was a problem that extended well beyond the boundaries of the computer field, and he argued that the ability of universities to relieve the problem was limited. Further, he followed in the footsteps of earlier speakers by warning against the dangers of "narrow specialization" in technical training (p. 8).

However, Householder was willing to promote mathematics departments as a locus of activity for university education and research in some phases of the computing field, even going so far as to state that he was "virtually advocating that the entire mathematical curriculum be oriented toward numerical analysis" (pp. 9-10). As further support for his recommendations, Householder noted that the problems faced by the programmer or numerical analyst were ultimately "mathematical in character" (p. 10). Yet Householder – like Carr before him – also recognized the importance of collaboration across extisting disciplinary boundaries, especially when dealing with the non-scientific applications of computers, such as in the business sphere. The "most rapid progress," he explained, would happen when

> mathematicians, engineers and business experts can be persuaded to join together in arriving at a common understanding by which to differentiate the primary needs of business from the incidental byproducts of established procedures, and then to devise the hardware and routines for achieving the real objectives (p. 10).

He concluded the article by asking, "Where could such teams form more readily than in a university?" (p. 10). It is worth noting here that Householder was reiterating his own prior

remarks – as well as those of others – in pushing the idea that intended applications should be a primary driver behind the development of both programming routines *and* hardware designs.

At a 1955 conference on "The Computing Laboratory in the University," a number of speakers offered candid remarks on the future role of the university in computer-oriented education and research. And indeed, many of these individuals expressed further skepticism about the ability of universities to the forefront of this domain. Jay Forrester, for example, forecasted that the "[t]he university will no longer be the primary training ground for computer experts" (Forrester, 1957, p. 18), and at another point he explained that on-the-job training was the main mode of professional training for those entering the electronic data-processing fields (p. 17).[123] Herb Grosch, on the other hand, noted at the same conference that "[l]eadership in systems engineering and in programming techniques may have to continue with the manufacturer and the industrial users of this equipment, and it may be difficult for the universities to continue to contribute powerfully even in components" (1957, p. 90). Such remarks provide further evidence for the rapidity with which computer research and development activities had shifted from universities and government research labs to the private sector in the first half of the 1950s, thereby impairing the ability of universities to develop and/or maintain their own research centers and educational programs.

Grosch also outlined some of the moves that universities needed to make if they wanted to assume a more prominent position in the computer field. In addition to encouraging "new thinking" and "more adventurous thinking," Grosch called on universities to "assemble their talents, cross departmental lines, build a few fires under some of the more mulelike faculties, and obtain financial support" (p. 90). Grosch's comments hinted at an even wider range of impediments – ranging from pre-existing departmental and disciplinary boundaries to faculty conservativism and financial pressures – that were seriously hampering the development of computer research and educational activities on many campuses.

Still other speakers at the conference discussed how engineering schools and departments might make strategic moves into certain phases of computer-oriented education. Applied mathematician and computer programming pioneer Forman S. Acton, for example, echoed many

[123] It is worth noting that Forrester's own research interests were beginning to moving away from computer design and development by the mid-1950s. His arguments for the declining role of the university in computer-oriented research and education may have therefore been skewed by his own movement out of these areas.

of his colleagues when he described a looming shortage of problem analysts, programmers, and coders. Explicitly noting that he was not interested in developing curriculum for "the small group who would design better machines or even devise better general codes" (p. 123), he went on to briefly describe a new "mathematical engineering" option that had been established in Princeton's School of Engineering (Acton, 1957, pp. 124-125). In addition to listing a series of associated courses in computation and mathematics, Acton noted that the students in such programs gained an understanding of the "the proper formulation and numerical solution of engineering problems" (p. 125). And elsewhere, he referred to this type of work as "computational engineering" (p. 122). As noted above, demand for this type of expertise was rising around this time, especially in computationally-intense industries such as aeronautics.

On the other hand, electrical engineer and University of Wisconsin-Madison faculty member Vincent C. Rideout was the only conference speaker to explicitly focus on the topic of "computer engineering curriculums" (Rideout, 1957, p. 156). More specifically, he noted that electronic engineering departments were a particularly appropriate location for training the "'triple-threat' men so eagerly desired in industry today – men who are soundly versed in mechanics, in electronics, and in computing" (p. 156). In contrast to Acton, Rideout emphasized that engineering students specializing in this new field should be well-versed in both compu*ters* and compu*ting*, and described a variety of computer-oriented courses that might be offered at the undergraduate and graduate levels – many of which were pulled directly from his school's catalog. He also suggested that the power and communications specializations typically offered to senior-level students might be supplemented by a computer option, and he listed a series of required and elective course for a computer-oriented master's degree program.

While Rideout's recommendations were slightly skewed toward analog computing, his published remarks stand as one of the first attempts to outline what a computer engineering curriculum might look like at both the undergraduate and graduate levels. In fact, he was well ahead of his time in proposing a computer-oriented degree option for undergraduate electrical engineering students. On the other hand, computer-oriented courses and programs in electrical engineering departments were being established rather slowly, and the developments that were underway at the University of Wisconsin appeared more an exception than a rule. Program reformers and developers such as Rideout clearly faced a host of challenges and barriers, such as those described by Grosch. In addition, university research in computer design was in an overall

state of decline by the mid-1950s, and the demand for analysts and programmers was far outpacing the need for computer designers. In summary, the field of computer design and engineering remained closely wed to – and largely overshadowed by – other and more historically dominant forms of research and education in electrical and electronics engineering.

Yet through the latter half of the 1950s, other developments were beginning to impinge on the academic landscape of computing. For example, there seemed to be growing demand for those with computer component or system design expertise in both the commercial realm generally and the defense sector specifically. Membership surveys revealed, for example, that the number of IRE-PGEC members affiliated with commercially-oriented firms in the private sector increased from 1011 (or 40% of members) in 1956 to more than 1600 (or about 42% of members) in 1960, while those affiliated with defense-oriented firms increased from 944 (or 37.5% of members) to more than 1700 (or about 44% of members) during this same period (Martin and Olson, 1957, p. 49; Uncapher, 1961, p. 84). Other evidence for these trends can be found in many of the aforementioned advertisements. IBM, for example, established a Military Products division in 1955, and was aggressively seeking to fill it with new engineers by at least 1957 (IBM, 1957).

By the late 1950s a handful of outspoken commentators were also starting to discuss how a new and more independent field or discipline of computing might be brought to fruition in the academic context. Not only did many of these proposals call for the crossing of departmental boundaries, they also recommended the establishment of entirely new computer-oriented institutes, departments, programs, and even schools. It is worth discussing these discipline-building projects in more detail, especially given the potential of this movement to impinge on the evolving relation of electrical engineering, computers, and computing.

Toward a Scientific Discipline of Computing

In order to frame the initial emergence of new discipline-building movements in the computer field, it is necessary to situate this story against a larger historical backdrop, especially the post-war ascendancy of science and the concomitant tilt of the engineering fields toward the so-called “engineering sciences.” While this general trend has been well-documented elsewhere, here I use a series of advertisements from IBM to show how scientific rhetoric and imagery

started to pervade the computer field, especially in the latter half of the 1950s.[124] In fact, IBM ads from as early as 1955 started to frame the company's work environment and employment opportunities through the rhetoric of science and scientific progress. These ads stood in marked contrast to many of the examples mentioned above, which had framed IBM's cutting-edge work in electronics and computer design as largely or even wholly within the province of engineers and engineering.

Some of the earliest evidence for this shift can be found in a December 1955 employment posting for electronic engineers that was dominated by a large banner that read: "The legacy of the scientist is the highest achievement of his predecessors. Only if he has ideas and ability can he reach greater heights" (IBM, 1955c). A 1956 IBM ad that sought engineers, on the other hand, was headlined with a suggestive quote from nineteenth-century British journalist and economist Walter Bagehot: "Nine-tenths of modern science is...the produce of men whom their contemporaries thought dreamers!" (IBM, 1956). The imagery for this particular ad included various busts and figureheads who were presumably famous scientists or intellectuals. Building on these thematic elements, the ad copy went on to explain that "IBM … has always sought in engineers and scientists that one source of all scientific achievement – *the ability to think fearlessly!*" Prospective employees were also informed that an IBM engineer "has every opportunity to make important and rewarding contributions to scientific progress."

These ads revealed a major change in imagery and rhetoric, especially as compared to IBM ads that had rung just a year or two before. Even more importantly, these ads suggested the emergence of two competing conceptions of computer research and development, one based on images of science and other on engineering. And while later IBM ads backed off on the scientific gusto, the examples highlighted here were a harbinger of things to come, especially after terms such as "communication science," "information science," and "computer science" entered circulation in the late 1950s and came into more widespread usage in the 1960s.

In fact, MIT electrical engineer Jerome B. Wiesner can be credited with one of the earliest attempts to identify and describe an emergent disciplinary domain that both encompassed large swaths of the computer field and carried a scientific moniker. Per Wiesner, the "communication sciences" were largely focused on the study of complex communication and

[124] On the early rise of the engineering sciences, see Seely (1999). On the continued ascendancy of science, scientists, and the engineering sciences in the in the post-Sputnik period, see Lucena (2005).

computing systems, with particular emphasis on "mathematical methods, computational techniques, and general understanding of communications" (1958, p. 268).[125] He also explained that problems involving "the processing, storage, and transmission of information" were both at the heart of this new science and centrally important in linking a wide range of existing disciplines, ranging from mathematics and the sciences to engineering and beyond.

Yet in describing this new field, Wiesner argued that reaching an adequate understanding of complex communication and information systems involved the development and use of theoretical and mathematical tools that were largely beyond the reach of engineers. As he cautiously explained:

> I don't want to underestimate the role of the engineer or the inventor in this field, because I think it will always be true, especially in a field as complex as this, that innovation and invention will probably outrun the theory, but they cannot outrun the theory very far if one hopes to have continued growth and development (1958, p. 270).

As suggested by this characterization, the dominant image of engineering around this time framed the field as only tenuously based on theory, especially when compared to the sciences. Wiesner also went on to counter a potential critique of the so-called "communication sciences" by explaining that it would be different from other scientific disciplines. More specifically, he clarified that this new area of research was largely focused on the "organizational" or "structural" rather than "physical" properties of complex systems (p. 269). Following this line of reasoning, he argued that work in the emergent field extended not only into the realm of computing machines and communication networks, but also into the domain of biology, such as in relation to the study of the nervous system. Perhaps not surprisingly, Wiesner supported this argument by referring to theoretical pioneers such as Claude Shannon and Norbert Wiener, who had laid important prior groundwork at the intersection of information theory, electronics, and communications.[126]

---

[125] Wiesner's article, which was published in IBM's *Journal of Research and Development*, also discussed the activities of MIT's newly established Communication Sciences Center.

[126] Wiesner explicitly referenced Wiener's *Cybernetics* (1948) as making important contributions in areas such as feedback and control theory (p. 274). Further, Wiesner framed the study of error-correcting and other types of feedback systems as an important component element in his larger vision for a field of communication science.

While Wiesner described a rather broad and far-reaching disciplinary project, other commentators placed more explicit emphasis on computers and computing. Louis Fein, for instance, is often credited with coining and promoting the term "computer science."[127] With a background that included stints as an engineer at both Raytheon and Computer Control Company – where he gained extensive experience in the design and development of high-speed digital computers and related devices – Fein launched a career as an independent consultant in 1955 (Fein, 1979). One of his early consulting jobs involved a study, commissioned by Stanford University, on the role of universities in computers and data processing, with particular emphasis on both the status of existing educational programs and the development of new curricula.

As recounted by Fein, his research led him to conclude that computing increasingly looked like a collection of emergent disciplines and sub-disciplines, and by 1957 he was widely using the term "computer sciences" to describe this federation of topics and activities (Fein, 1979). His earliest published remarks on the topic appeared in three papers, each bearing the same title: "The Role of the University in Computers, Data Processing, and Related Fields." The first of these papers was Fein's report for Stanford, the second was delivered at the Western Joint Computer Conference in 1958, and the third was published in the *CACM* in 1959 (Fein, 1961a, p. 167). As suggested by this review of the literature, Fein's ideas were widely distributed in the computer field, especially via the latter two publications.

With regard to the 1959 *Communications* article – which was probably the most widely read of these three pieces – the author's evaluation of existing university research and education in computing was almost wholly negative. After noting that some 150 universities and colleges were "engaged in some kind of activity in the fields of our concern," he nonetheless argued that there was a profound lack of "distinguished academic centers of computers," and a dire need for "integrated" approaches to computer research and education (Fein, 1959, p. 9). Fein also provided a laundry list of topics and courses in computer design and applications that were receiving some attention at colleges and universities, although he simultaneously bemoaned the field's rather meager theoretical foundations. Echoing Wiesner's concerns, Fein explained that

[127] Historian Paul Ceruzzi, for instance, points to Fein's 1959 article in the *CACM* as the origin of the term "computer science" (Ceruzzi, 1989, pp. 266-267). In this particular piece, Fein repeatedly places "computer sciences" in quotes, suggesting some uncertainty over the phrase. At various points he also refers to both the plural "computer sciences" and the singular "computer science." Fein has more recently claimed that he first adopted the term in the middle of 1956 (Fein, 1979, p. 7).

"the fields of computer *theory*, application *theory*, model *theory* do not yet appear to have been successfully attacked" (p. 10, author's emphasis).

Fein responded with a series of recommendations aimed at moving universities into a more prominent position with regard to computer research and education. In most general terms – and as noted by commentators such as Ceruzzi (1989) – Fein argued that the development of the "computer sciences" first and foremost demanded sound organizational and administrative footing. Fein's rather imprecise and tentative outline of the new field suggested that its definition and scope would become more clear over time, especially once the appropriate support structures were in place. Following this line of reasoning, he promoted the creation of a graduate school of computer science, composed of five new academic departments and a computation center.[128] Fein added that these new schools would likely enter into collaborative relationships with a range of existing departments, while also working to "develop the new disciplines" (p. 12) and to establish "integrated" programs for students pursuing advanced degrees.

As suggested by this overview, Fein described the computer sciences as a kind of theoretical "supra-discipline," somewhat akin to mathematics, which was similarly linked to various offshoots and fields of application. On the other hand, he made it clear that this new field would both draw from and inform work in other domains, such as engineering, business, and the sciences. However, engineers and engineering received little in the way of special mention in Fein's analysis. In fact, he echoed other commentators when he argued that there was "little reason" for universities to build their own computers, especially given the ready availability of commercial equipment. And as he emphasized the importance of the more mathematical and theoretical dimensions of computing, he went so far as to note that "an excellent integrated program in some selected fields of the computer sciences should be possible *without* any computing equipment at all" (p. 11).

In the early 1960s, some of Fein's vision was on its way to being realized, including through the increased use of terms such as "computer science," as well as via ongoing moves to establish new computer-oriented university departments and programs. Yet by this time Fein was

[128] Fein described a rather general "Computer Department," as well as departments dedicated to Operations Research, Information and Communication, Systems, and Philosophy of Organization (1959, pp. 12-13). And while the overall description of the five departments tilted toward theory, applications, and programming, Fein associated a number of "hardware" topics – including computer organization, component and circuit research, and systems research – with the proposed "Computer Department."

promoting an even more ambitious disciplinary vision. In a 1961 article that was published in both *American Scientist* and *Datamation*, Fein creatively imagined and described the emergence of a new science, circa 1975, that he dubbed "synnoetics" (Fein, 1961a; 1961b).[129] While the details of Fein's article are largely beyond the scope of my analysis, the author's explicit reference to the "Communication Sciences" continued themes discussed by prior commentators such as Wiesner. More specifically, Fein described synnoetics as a "supradiscipline," and he noted that the "computer sciences" were a branch of synnoetics that focused on "the theory and practice of the design, programming, and application of computers" (p. 151). The author also borrowed Simon Ramo's term "intellectronics" to describe another sub-branch of synnoetics that was closely associated with engineering schools and focused on the "the implementation of synnoetic systems by electronics" (p. 151).[130] Fein's proposal also anticipated tensions regarding the academic status of the computer sciences. Forecasting what the academic landscape might look like in 1975, he explained: "The academic community did not acknowledge that the study of the design, programming and applications of computers constituted a discipline in the classical sense" (p. 160). Reiterating his earlier remarks, Fein also spoke out against programs that were centered on and dominated by computing equipment, rather than higher-level theoretical and "supra-disciplinary" foundations.

While commentators such as Wiesner and Fein put forward ambitious disciplinary agendas that framed computers and computing as important elements in a much large disciplinary framework, significant barriers stood in the way of realizing their vision. For instance, both of these reformers likely underestimated the challenges that came with shifting the dominant, unifying image of the computer field away from the boundary object of the computer and toward a milieu of theory that remained substantially ill-defined and inchoate. Further, Fein

---

[129] As explained by Fein, "Synnoetics is the science treating of the properties of composite systems – consisting of configurations of people, mechanisms, plant or animal organisms, or automata – whose main attribute is that its ability to invent, to create, and to reason – its 'mental' power – is usually greater than the 'mental' power of its components" (150).

[130] Simon Ramo, the originator of the term "intellectronics," described the field as "[t]he science of extending man's Intellect by Electronics" (1960, p. 6). He outlined his views on this new "science" at the 5th National Communications Symposium in 1959, and a brief excerpt from the talk was reprinted in *Computers and Automation.* (Ramo, 1960). In subsequent years he promoted the concept and field of intellectronics in a various ways, including via lectures and through his leadership position at aerospace company Thompson-Ramo-Woolridge Corp. (later TRW). It is also worth noting that both Wiesner and Ramo were scheduled to speak in a 1958 AIEE conference session dedicated to "Computing Devices and Research – Thinking Machines of the Future" ("AIEE Winter General Meeting," 1958, p. 78).

apparently received many negative responses when he started presenting his discipline-building ideas to larger audiences in the mid and late 1950s.

At a small seminar at Berkeley, for instance, Fein made one of his first calls for the establishment of separate university departments of "computer science" (Fein, 1979). A number of well-known mathematicians and computer pioneers were present, including Derek Lehmer, Edward Feigenbaum, and Julian Feldman (Fein, 1979, p. 10). As recounted by Fein, the reaction of the attendees was almost wholly negative, with most viewing the proposal as unrealistic at best, and "crazy" at worst (pp. 10-11). Subsequent encounters were similarly fraught with tension and resistance. Fein has retrospectively indicated that Jack Herriot, a Stanford mathematician and early head of the Stanford Computation Center, was particularly hostile to the idea. At one meeting, Herriot apparently made a comment along the lines of: "What you want it [sic] pie in the sky and you can't have pie in the sky!" (p. 11).

Tracing out the details of this story reveals two additional actors of note. The first was Harry Huskey, the aforementioned Berkeley Professor of Mathematics and Electrical Engineering who attended Fein's seminar at Berkeley. According to Fein, Huskey was another early naysayer who "saw no need whatever for having a separate department. He was doing computing in engineering" (Fein, 1979, p. 10). Another important actor was George Forsythe, a Stanford mathematician who was one of only a handful of individuals who had mixed feelings about the merits of Fein's ideas, especially as applied to Stanford. According to Fein, the Stanford mathematics department was strongly opposed to a new computer-oriented department, yet Forsythe was "equivocal; he could go for it or not go for it" (p. 11). The reactions of Huskey and Forsythe are particularly significant in light of subsequent events. First, Berkeley and Stanford were two important sites where the development of computer-oriented curricula proceeded along two very different pathways. And second, both Huskey and Forsythe served terms as ACM Presidents. I revisit these themes below.

As I discuss in the following chapter, calls for the establishment of new university computing departments gained significant momentum in the early 1960s, often in tandem with the increasing use of the "computer science" moniker. Yet the appeal of this new terminology was also soon reflected in various commercial employment ads. Both Philco and MITRE, for example, indicated employment opportunities for "computer engineers and scientists" in the early 1960s (Philco, 1960b; MITRE, 1961). And around the same time, a series of

advertisements from the General Motors Research Laboratories described exciting career possibilities for mathematicians, engineers, and physicists "at the edge of computer science" (General Motors, 1960; 1961a). A similar 1961 ad from General Motors, on the other hand, dropped all reference to engineering, and was aimed instead at "applied mathematicians" and "programmers" (General Motors, 1961b). The associated ad copy also indicated a need for "Research Mathematicians" and "Senior Programmers" to work on "advanced computer applications."

Once again, these advertisements reveal the segmentation of the computer field into two major spheres of activity, the first focused on mathematics, programming, and "computer science," and the second on engineering and design. By the early 1960s, however, questions remained about the extent to which it was desirable or even possible for electrical engineering departments and faculties to claim various areas of computer-oriented education and research. In the following sections, I review some of the 1950s-era discussions about the role of computers, computing, and related topics in the realm of electrical engineering education. My analysis is organized around two major sections, the first focused on educating computer-using engineers, and the second on educating computer engineers and designers.

Educating Computer-Using Engineers

As noted above, through much of the 1950s computer-oriented topics and courses were filtering rather slowly into the electrical engineering curriculum at the majority of schools, despite the rapid growth of the computer industry and widespread use of computers in diverse organizational settings in both the public and private sectors. Yet by the late 1950s and early 1960s it was increasingly apparent that those university engineering departments that chose to ignore or downplay new computer technologies and computing techniques did so at their own peril. Further, electrical engineers were doubly challenged by these trends given their unique position as potential computer users *and* designers. On the other hand, this section demonstrates that general discussions about familiarizing engineers and engineering educators with computer-based problem-solving methods at times tended to overshadow the ongoing development of courses and programs in more specialized areas such as computer design and engineering.

In fact, discussions about teaching computing skills to engineers surfaced with increasing frequency in outlets such as the *Journal of Engineering Education* (*JEE*). In a 1959 article, for

example, Bruce Gilchrist – who was serving at the time as Director of Syracuse University's Computer Center – discussed the current and prospective position of universities with regard to teaching the "use" of computers (or "computing") to students, engineers included (Gilchrist, 1959). In addition to outlining existing computer programming courses at Syracuse, Gilchrist proposed a new undergraduate major in "mathematics-computing" that was largely focused on numerical analysis skills and designed to produce the "expert programmer and coder" (p. 344). Yet in contrast to other commentators, the author argued against the creation of new departments for computer-related instruction, and he instead claimed that programming courses should be offered by existing units, such as mathematics departments. As Gilchrist explained:

> While computers are not solely mathematical, there is so much about them, both in respect to design as well as to use, which is essentially mathematical, that it would seem difficult to find a more appropriate department to offer the fundamental programming and coding courses (p. 345).

Yet the author also acknowledged that it was appropriate for Electrical Engineering Departments to offer courses in computer design and construction, and for Business Schools to develop offerings in areas such as data processing.

Gilchrist's remarks were followed over a year later by a special issue of the *Journal of Engineering Education* (*JEE*) that was dedicated to the topic of "Computers in Engineering Education." In line with Gilchrist's remarks, two of the articles in this issue were focused on educating students in the area of computer "use" or "application." Contributor Adolph Katz, to begin with, suggestively titled his article "Do Computers Belong in the Engineering Curricula" (1960). Answering in the affirmative, the author explained that extant "cut and try" approaches to engineering design were quickly being replaced by more complex analytic and mathematical methods, many of which depended on the use of analog and digital computing devices (p. 835). By surveying college catalogs, Katz concluded that a majority of university courses in "computer application" were being taught by electrical engineering departments, but he added that mathematics departments offered many similar courses (pp. 835-836). Per Katz, this trend could in part be explained by the fact that EE departments had a long housed and maintained many different kinds of electronic equipment and related devices, including computers.

Katz went on to conclude that the universities were doing a good overall job of teaching computing techniques, but he recommended that existing engineering courses needed to move

beyond the subjects of computer components and circuits to more thoroughly cover related mathematical foundations and numerical techniques. As Katz argued, “It is the responsibility of the university to teach the modern engineering student more of the fundamentals of the field, and to present the scientific laws and mathematics required to enable the engineer to use the modern tools available to him for the practical solution of engineering problems” (p. 838). Katz’s comments clearly hinted at a long-standing challenge in engineering education that involved the delicate balancing of disciplinary “fundamentals” with more “practical” types of tools, skills, and knowledge. This challenge was exacerbated by post-war trends in engineering education that led to an increasing emphasis on fundemantals, theory, and the “engineering sciences,” thereby leaving little room in the crowded curriculum for more specialized topics.

In another article in the same special issue, mathematician William F. Atchison of the Georgia Institute of Technology more specifically discussed the rising importance of numerical analysis in engineering problem solving (Atchison, 1960). Pointing to the need for close cooperation between existing mathematics and engineering departments in training students in this area of expertise, Atchison also noted the “fairly strong chorus of voices calling for a new department … devoted to coordinating all the various aspects of computing” (p. 857). And while the author mentioned “Information Processing” as a likely choice for the name of such a department, most of his remarks were limited to promoting the use of computers as problem-solving tools among engineering faculty and students.

In a lengthy follow-up letter that was published in the wake of the special issue, William F. Luebbert of the Department of Electrical Engineering at the United States Military Academy added important insights regarding the manifold effects that computers and “information processing techniques” were having on electrical engineering. For starters, he described the historical “evolution” of electrical engineering education by outlining a gradual shift in emphasis from electrical power to communications and electronics, especially in the core courses of the discipline. Even more importantly, he noted that topics related computers and information processing – which appeared at least as significant as earlier developments in the field – were often relegated to special dedicated courses. As explained by the author, the rapid growth of the computer field “has tended to set computers off as a distinct technical field or specialty” (Luebbert, 1960, p. 134).

Luebbert went on to note that the suggestion of establishing "Information Processing Departments" tended to come from "those educators with a less predominantly electrical viewpoint," and he concluded that such ideas were "startling," "radical," and "thought-provoking" (p. 136). He also wrote them off as "neither feasible nor desirable," especially when "more evolutionary" approaches were possible (p. 136). As one such alternative, the author argued for the extensive integration of computers and information processing topics throughout the electrical engineering curriculum, and especially in the foundational core courses that electrical engineers were required to take at most institutions.[131] On the one hand, Luebbert's remarks reveal a rising anxiety among engineers as the idea of separate computer-oriented departments and programs gained traction. On the other hand, his call for a thorough infusion of computing in all phases of electrical engineering education foreshadowed various curriculum reform efforts that gained traction beginning the mid and late 1960s. I discuss these movements in more detail in the following chapter.

Another noteworthy article appeared that appeared in the *JEE* in 1961 was topically titled "Engineering Students Must Learn Both Computing and Mathematics" (Forsythe, 1961). Authored by Stanford mathematics professor George Forsythe, this piece paralleled many of the aforementioned articles in that the author made a general call for improving the mathematical and computer-related aspects of engineering education. Yet Forsythe also came out as a strong advocate for the so-called "computer sciences." At the very beginning of the article, for instance, Forsythe noted with slight derision that "[computers] are developing so rapidly that even computer scientists cannot keep up with them. It must be bewildering to most mathematicians and engineers" (p. 177). In addition to downplaying the position of mathematicians and engineering in the area of computing, the author's rather matter-of-fact reference "computer scientists" suggested that the identity of this group was well established. Yet at another point in this same article, Forsythe was more cautious in describing the "computer sciences":

[131] Evidence suggests that the types of reforms suggested by Luebbert were being proposed and discussed more widely. In 1959, for example, the Electrical Sciences Committee of the American Society for Engineering Education (ASEE) published a "Report on the Engineering Sciences, 1956-1958" (American Society for Engineering Education, 1958). In addition to revealing a continued emphasis on science in engineering education, the report also, in Luebbert's words, "advocated the complete reorganization of electrical science curricula into an energy processing portion and an information processing portion" (Luebbert, 1960, p. 136).

> In spite of the diversity of the applications, the methods of attacking the different problems with computers show a great unity, and the name of Computer Sciences is being attached to the discipline as it emerges. It must be understood that this is a very young field whose structure is still nebulous (p. 177).

Embellishing this depiction, the author described the "theory of computer programming" as a key area of activity in the emergent field, and he also identified numerical analysis, the study of data processing, and computer system design as "computer sciences" (p. 178). Framing computer design as a branch of "computer science" revealed the wide range of topics – including many that were traditionally connected with engineering – that many commentators claimed such a field should encompass. And while Forsythe stopped short of explicitly calling for the founding of new computer science departments in universities, he hinted in this direction when he suggested that introductory computer courses should be taught by "computer scientists." Further, he noted that these types of courses might be developed more rapidly if the associated instructors were "not judged primarily by the standards of any existing department" (p. 180). One might infer that this could only be accomplished through the establishment of new departments that were linked to a new discipline.

In the following chapter, I trace the historical trajectory of the "computer sciences" thought the 1960s and beyond. And in order to round out the present discussion, it is necessary to review a handful of developments beyond the pages of the *JEE*, including an ambitious and well-known project that was launched at the University of Michigan in 1959 to more broadly stimulate and accelerate "the use of computers in engineering education." Sponsored by the Ford Foundation and directed by Donald Katz – a Professor of Chemical Engineering at the University of Michigan – the impact of the project extended to dozens of other universities through the participation of over two hundred participating faculty (Katz et al., 1963, p. v).[132] As explained in one final report, one major original impetus for this project stemmed from the general observation that "experiences with computers in research were filtering down very slowly into undergraduate engineering education" (Katz et al.,1963, p. iii). Building on the premise that

[132] In fact, evidence suggests that the overall impact of the Ford Foundation project was indeed quite significant. As University of Michigan computer scientist Bernie Galler explained in a 1991 interview, "Computers were really quite new, and it was recognized that they were very useful in engineering, but they hadn't really gotten into the curricula. I know people who told me that in their opinion, this project advanced the use of computers by at least five years throughout this country" (Galler, 1991, p. 32).

"[a]ll graduating engineers of the future must have a knowledge of computers, just as they have a knowledge of mathematics" (Katz and Organick, 1960, p. 184), the project aimed to "study the feasibility of broad scale integration of electronic computer use into the educational process" (Katz et al., 1963, p. iii). In working toward this goal of integration, much of the project was dedicated to developing appropriate teaching materials, sample problems, and appropriately trained faculty. It also covered a range of engineering disciplines, and much of the associated work was accomplished via summer training programs and workshops.

While the Ford Foundation project may appear somewhat tangential to the mainline development of computer engineering, the project reports provide important evidence about the general position of computers in electrical engineering education, especially in the early and mid 1960s. In fact, a 1966 report on the role of computers in electrical engineering design education explained that "[e]lectrical engineers have an interest in computers not only as devices for performing computation but also as systems of interest in themselves" (McMahon et al., 1966, p. IV-1). Yet somewhat paradoxically, this same report concluded that "the computer as an aid to engineering design has not been emphasized in electrical engineering to the extent that it has in other engineering disciplines" (p. IV-1). Far from conjecture, this tendency was widely documented. A survey published in 1963, for instance, indicated that just over half (54%, or 22 of 41) of all responding electrical engineering departments were teaching computing courses, while only 45% (or 21 of 47) of surveyed EE departments required its undergraduate students to take one or more computer courses (Cook, 1963, pp. 5; 9).[133] The authors of the 1966 Ford Foundation report offered two explanations for these trends. First, they pointed to a general neglect of design-oriented education in most engineering fields, which limited the extent to which computer-based techniques could be incorporated into the curriculum. And on a second and closely related note, they suggested that the use of computers among electrical engineers was limited by the fact that "many of the problems with which electrical engineers deal are fairly tractable mathematically" (McMahon et al., 1966, p. IV-1).

[133] Interestingly, the report noted that industrial engineering departments were offering the most computer courses for engineering students (58%), followed by electrical engineering (54%). In addition, 77% of industrial engineering, 47% of aeronautical engineering, and 47% of chemical engineering departments were requiring computer courses for graduation, as compared to 45% for electrical engineering departments.

As I note below, groups such as the COSINE Committee emerged in the mid-1960s and in part picked up where the Ford Foundation project left off, especially with regard to urging electrical engineering educators to incorporate computers into all phases of the EE curriculum. Yet unlike the participants in the Ford project, COSINE members also raised questions about the extent to which it was desirable for electrical engineering departments to cultivate student and faculty expertise in a wider array of subjects, ranging from computer design and systems engineering to programming and even the so-called "computer sciences." Yet well before the COSINE effort got off the ground, a handful of commentators were both raising questions about and making tentative recommendations for how electrical engineering departments might expand their interests in computers and computing generally, as well as in the area of computer design and engineering more specifically.

Educating Computer Engineers and Designers

As described above, by the mid-1960s a growing cadre of engineers and other interested actors were increasingly focused on teaching future generations of engineers how to use computers, albeit with mixed results. A somewhat smaller pool of individuals was approaching questions about the appropriate role of computers and computing in engineering education from other angles. As noted above, a 1957 article by Rideout was one of the first to discuss the potential development of "computer engineering curriculums," and within a few years a handful of authors were pursuing related themes. In the 1960 *JEE* special issue on Computers in Engineering Education, for example, MIT civil engineer C. L. Miller and electrical engineer W. W. Seifert discussed the position of engineering schools with respect to a full range of "computer know-how" (Miller and Seifert, 1960). In most general terms, these authors argued that "the faculty-computer relationship must embrace the entire faculty – that the ideal 'computer faculty' includes the entire faculty of the engineering school" (p. 839). They also went on to identify four constituent areas of expertise, namely design, communications, mathematics, and applications (p. 841).[134]

[134] The authors used the term "communications" to refer to "man-machine communications." This area of activity included topics associated such computer programming and languages. With regard to "applications," the authors were principally referring to the use of computers as a "problem solving tool of modern engineering" (p. 841).

With regard to computer design, Miller and Seifert framed engineering as a rather obvious locus of activity and education for education and research. In fact, they supported this claim by pointing to a long-standing affinity between engineering design and physical artifacts. "Those concerned with computer design," they explained, "pose no identification problem due to their direct relationship with the development of computers as physical devices" (p. 841). For these authors, computer designers were naturally machine-oriented engineers. And indeed, my preceding analysis of job advertisements and employment statistics revealed the strength of this relationship in the late 1950s and early 1960s.

Yet Miller and Seifert also promoted the development of a full range of computer know-how among engineering faculty that extended well beyond the design of "physical devices" to cover all four of the major constituent areas listed in their article. The authors even argued that engineering schools and faculty needed to develop and sustain their own expertise in the more mathematical areas of computing. "It should be made clear," Seifert and Miller stated, "that in identifying the mathematics group we are speaking of those professors within the engineering school who are active in developing and enlarging mathematical methods and not of the staff of the mathematics department" (p. 842). By arguing that all relevant phases of computing should remain within the province of engineering schools, the authors promoted their own vision of disciplinary independence and self-sufficiency. It is also likely that this position was developed in response to the ongoing extension of mathematics departments and faculties into many areas of computing, not to mention the tentative development of the "computer sciences." Yet regardless of their motivations, Miller and Seifert's recommendations appeared particularly synergistic with their home institution, where engineering had a long reputation for its prestige, mathematical and theoretical intensity, and wide-ranging scope.

In another special issue article titled "Setting up a Computing Faculty in a School of Engineering," Moore School director John Brainerd worked in directions that largely paralleled the comments of Miller and Seifert.[135] Noting the existence of ongoing debates regarding whether separate faculty and departments should be developed in areas such as the "Computer and Information-Processing Sciences," Brainerd downplayed the prospects for such departments by describing the proponents of such a split as "relatively few though at times outspoken"

[135] As noted in the preceding chapter, in as early as 1948 Brainerd had emphasized the major role of electrical engineers in the historical development of large-scale computing devices.

(Brainerd, 1960, p. 846). Summarizing his own position on the matter, Brainerd argued that existing departments were meeting most of the new field's needs, and he added that "the administrative need for a separate department or faculty is not evident despite the crossing of lines of various disciplines which can be envisaged" (p. 846). He even followed a number of earlier commentators by suggesting that computing could be viewed as a sub-field of engineering. As Brainerd explained, "the creation or extension of a 'computing faculty' group is much like that for any other new and important subdivision of engineering effort" (p. 851). Such comments revealed a rising sense of anxiety among engineers as the establishment of new computer-oriented disciplines and departments appeared an increasingly likely prospect at many schools, including Brainerd's own.

Brainerd's comments also reflected a common rhetorical strategy, where computing was framed as a partially or even wholly a province of engineering. In fact, preceding chapters show that this type of strategy can be traced back to the early 1950s. Yet Brainerd was also forced to admit that computer-related research and education involved other types of expertise. In fact, he hinted at rising concerns about the role of mathematicians, philosophers, and other "non-engineers" in the development of computer-related courses and programs, especially in engineering schools and departments. "A computing faculty in a school of engineering," Brainerd warned, "must be prepared to open their arms to the qualified non-engineer" (p. 850). As I discuss in more detail below, accommodating these "outsiders" was an important strategy – as well as a demanding challenge – for Brainerd and other engineers as they struggled to maintain control over new computer courses, programs, and faculty. Further, the possible participation of "qualified non-engineers" in engineering schools and departments seemed to call into question the identity of engineering as a distinct professional and disciplinary domain.

Brainerd's article raised yet another theme that has surfaced repeatedly in my analysis, namely the extent to which computer design was related to computer application or use. As the author explained, "Logic design should ideally be carried out with full knowledge of the ranges which the equipment designer can achieve as well as knowledge of the basic logic of the contemplated device; unfortunately logic design has fallen to a low state in some instances" (p. 849). The aforementioned article by Miller and Seifert also touched on this issue. After noting that the interests of computer design groups tended to concentrate on circuitry and logic, the authors added that "designers must have contact with those who are going to use the machines"

(Miller and Seifert, 1960, p. 841). Providing further insights regarding the budding divisions of labor among computer experts, these same authors described how a "computer languages group" could act as an essential link users and hardware of computing (p. 841). Of course, promoting cooperation between such groups was quite different than the type of proposal that authors such as Fein had bandied about, namely that the computer field writ large could somehow be unified via some sort of common theoretical framework or even disciplinary structure.

And finally, Norman R. Scott, a professor of electrical engineering at the University of Michigan, contributed the only article to the same *JEE* special issue that was primarily dedicated to discussing the development of courses and degree programs in the area of computer design and engineering (Scott, 1960). After emphasizing the general importance of computers in the engineering curriculum, the author reviewed a series of challenges that were facing electrical engineering faculties, with particular emphasis on the area that he called "computer engineering." Ultimately coming down in favor of devoting "many courses or even a degree program to the engineering of the digital computer" (p. 852), Scott asked a series of related questions about appropriate lab facilities, the relative coverage of analog versus digital topics, and the extent of training in computer use among electrical engineers (pp. 852-853). But even more importantly for the present analysis, the author queried: "What material is appropriate to computer engineering courses at the undergraduate level and the graduate level?" (p. 853). He also pondered the extent to which topics in the area of computer design and engineering should be taught by electrical engineering faculty versus other academic units, such as mathematics or philosophy departments (p. 853).

While leaving many of these questions open, the author described ongoing developments at the University of Michigan, where "courses in the engineering of computers have been presented since 1951" (p. 853). Scott also stressed that computer engineering courses often required a "broad background in electrical engineering and a high level of mathematical maturity," and he promoted the idea that computer engineering was a more appropriate area of specialization at the graduate level (p. 854). Building on these arguments, Scott listed some of the key topics that would likely be included in graduate courses in computer design and engineering. He further grouped these into a set of abstract topics that provided necessary mathematical and theoretical grounding – namely symbolic logic, Boolean algebra, state diagrams, switching, automata theory, and finite number systems – and Scott acknowledged that

philosophy or mathematics courses might help provide coverage in these areas. He also listed a set of subjects more closely associated with existing technologies and techniques, including arithmetic processes, command lists, the relation of "internal" and "external" languages, memory systems, and computer circuits and logic. While little more than an outline, Scott's article is noteworthy in that it was one of the earliest published descriptions of graduate-level coursework specifically dedicated to computer design and engineering. Further, Scott's remarks hinted at a growing sense among many electrical engineering educators that an emergent field of computer design or computer engineering was starting to coalesce around some common core bodies of knowledge and domains of technology, albeit largely as a specialization or branch of EE.

Many of the themes developed in the special issue of the *JEE* were echoed in a lengthy summary article on the topic of "Computer Education," which appeared in the 1963 edition of *Advances in Computers* (Tompkins, 1963). After first reviewing computer education and training programs in industry, author Howard E. Tompkins noted that universities lagged well behind industry "in their ability to teach new concepts and methods that have arisen from industrial research and development" (p. 139). Describing most university curricula in computing as "experimental" or "in transition," Tompkins outlined two major types of curriculum that were being developed at the time, one focused on the "engineering design of computers" and the other on the "utilization of computers" (p. 142).

Regarding the former, the author explained that most engineering curricula were not yet explicitly addressing the topic computer design, although he admitted that many programs and courses were grappling with a variety of closely associated topics. Tompkins also stated that a range of new technological developments that were closely related to digital computers – such as transistors and other types of solid-state devices – were not receiving adequate attention in most undergraduate engineering curricula. He added that more could be done to provide up-and-coming engineers with a synthetic view of computing, especially by cultivating expertise in areas ranging from electronics and circuits to logic and systems. Reacting to the challenges that came with developing such curricula, Tompkins emphasized that "[t]he two points of view, logical and electrical, should be coordinate into a meaningful whole" (p. 142). While offering little in the way of additional detail, he concluded his remarks on engineering design by referencing a series of textbooks in "computer engineering, logic, and systems design" (p. 142).

In his survey of "courses and curricula aimed at the education of computer users," on the other hand, Tompkins described a wide range of trends and possibilities, ranging from short, non-credit courses in programming to full degree programs with titles like "Computer and Information Sciences" or "Communications Science." Describing some of the developments that were on the horizon, Tompkins offered summary descriptions of computer-oriented degree programs at about a dozen universities, and he noted a variety of faculty training initiatives that were underway, including the aforementioned Ford Foundation Project. Concluding his rather wide-ranging discussion of computer education at the university level, Tompkins also pondered the process of discipline formation: "Will a recognized and established discipline, complete with traditions, established standards, and a few sacred cows, arise?" (p. 151). While a small but growing cadre of commentators were confidently projecting the establishment of such a discipline by this time, Tompkins took a somewhat more cautious – and perhaps more realistic – view. "Not without pain and effort, and not in any thoroughgoing way, Tompkins declared, "the dynamics of the situation are too dominant. The field is still developing and changing too rapidly to admit of much immediate standardization" (p, 152).

Conclusion

On the surface, it may appear somewhat ironic that electrical engineering departments – which had both reasonably strong and relatively early claims to many aspects of the design and development of computing machinery – were slow to bring computers into their courses and curricula. However, the direct influence of university-based electrical engineers in the computer field peaked in the 1940s and early 1950s at a handful of high-profile hubs of activity, such as MIT and the University of Pennsylvania. As the commercial computer industry took off in the 1950s, university computer design research declined accordingly, and by the mid-1950s it was increasingly rare for even the largest and best-funded schools to undertake major computer development projects or related research initiatives, albeit with a few notable exceptions. These trends – coupled with a strong, post-war tendency for electrical engineering departments and faculties to both embrace the engineering sciences and resist rapid curricular changes and technical specialization – created a climate in which computer-related topics and technologies filtered slowly into electrical engineering courses and curricula, even into the 1960s. In fact, many private-sector employers were largely content with this status quo, even if they frequently

assumed the primary burden of transforming recent graduates who were well-schooled in engineering fundamentals into effective engineering practitioners, such as via supplemental training programs and on-the-job experiences.

And while electrical engineers continued to fill large numbers of private-sector employment slots in areas ranging from electronics and circuit design to logical design and systems engineering, the demand for numerical analysts, programmers, and other applications experts rose rapidly through the 1950s and into the 1960s. With electrical engineering departments reluctant to move into these areas, other academic units and programs filled in the gaps. Mathematics departments, for example, started to produce large numbers of students who would go on to become analysts and programmers. In addition, a growing raft of commentators proposed the establishment of a new computer-oriented academic discipline that covered a wide spectrum of activity in the field. On the one hand, this movement seemed to be the sort of reform that commentators such as Carr and Gorn were looking for, as it suggested that it might be possible to reconcile some of computer field's major social and technical divides. On the other hand, there remained the very real possibility that the emergence and rise of the "computer sciences" might further fracture the field by driving a wedge between an emergent class of "computer scientists" and a counterpart pool of computer designers and engineers. In fact, this latter pattern appeared synergistic with the major sociotechnical boundaries that had come to dominate other aspects of the field, including the major professional societies, various private-sector worksites, and even the sphere of computer technology itself.

As Tompkins explained in 1963, many questions remained about whether a new computer-oriented discipline could emerge and thrive, especially against the backdrop of rapid technological change. Yet his focus on issues of "standardization" and "established traditions" partially missed the mark, given that the academic context more often serves as a site where disciplinary settlements are worked out and periodically renegotiated, with discipline-based departments and graduate degree programs helping to reproduce various fields of interest, while also preserving their internal and external stability. Tompkins also failed to ask an important follow-up question: if a new discipline of computing was successfully established, would it be sociotechnically integrated, thereby making one head out of the proverbial Humpty and Dumpty? Or would it reflect and reinforce the kinds of dichotomies that had grown up in other contexts of computing? Such questions help set the stage for the following chapter, which follows the

historical development of two major discipline-building movements in the computer field, the first centered on the independent development of “computer science,” and the latter involving the redoubled efforts of electrical engineering reformers and educators to promote the development of their own brand of computer-oriented courses and curricula within existing engineering schools and departments.

Chapter 5
Competing Images of Disciplinarity: Computer Science, COSINE, and Computer Engineering

In prior chapters I documented the historical trajectory of computer engineers and engineering by looking at various contexts of disciplinary and professional development, including professional societies and industry worksites, as well as the domain of computer technology itself. More specifically, my analysis highlighted how the computer field's early decades were significantly marked by the emergence of two distinct sociotechnical territories, one focused on software and programming, the other centered on engineering, design, and hardware. The preceding chapters also documented persistent uncertainty over the status of computer engineering as a distinct field or discipline. In fact, terms such as "computer engineer" and "computer engineering" were in wide circulation in the 1950s and 1960s, but primarily in reference to various jurisdictions of professional work in the context of industry. My account also revealed significant ambiguity in the use of "computer engineer" and related terms, especially in the mid-1950s. Yet even as these terms became more narrowly associated with the design of computer systems and associated devices, computer engineering was often couched as a subfield or branch of electrical engineering, rather than as a discipline or profession unto itself.

By delving into the educational arena, the prior chapter presented early evidence for the emergence of a discipline that over time came to be called "computer science." It also outlined the tentative efforts of electrical engineering faculties and departments to enter various areas of computing, including through a handful of graduate-level courses and programs dedicated to computer design and engineering. The present chapter carries this analysis of the educational sphere through the remainder of the 1960s and into the early 1970s, with particular emphasis on two related historical movements. The first of these involved ongoing efforts to define, position, and institutionalize the budding field of computer science, especially through the efforts of the ACM and its mathematically- and theoretically-oriented constituency. In addition to detailing a

series of early efforts to succinctly define this new field, I draw on the work of Abbott to suggest that the success of computer science largely hinged on both the negotiation of the field's disciplinary "settlement" and the establishment of new computer science departments and graduate degree programs. Hence, my analysis brings into further relief the importance of "bottom-up" processes of discipline-building.

In a segue section in the middle of this chapter, I turn to handful of "insiders" who critiqued the identity and direction of computer science education, and who questioned whether the emergent field was moving too far away from engineering and technology. This line of analysis provides an appropriate transition into the second major part of this chapter, which details how a new cadre of electrical engineers called for a thorough reorientation of electrical engineering education toward computers and computing. In fact, two Bell Labs researchers aptly captured the underlying ethos of this movement when they framed engineering and computing as a "holy alliance" in a 1960s-era memorandum. In order to realize this alliance, many of these reformers initially worked to bring computer science "into the fold" of electrical engineering, especially through the efforts of the COSINE Committee in the educational arena. Yet for a variety of reasons, the goals of this movement gradually shifted toward the development of courses, programs, and degree options in the area known as "computer engineering."

On the one hand, my analysis once again highlight the tendency of the computer field to cleave into hardware- and software-oriented spheres in a variety of contexts. But perhaps even more importantly, this chapter reveals the crucial importance of the academic sphere in the negotation and development of disciplines. In fact, ongoing efforts to attach the computer engineering moniker to various facets of electrical engineering quite crucially set the stage for the gradual and widespread emergence of computer science and computer engineering as distinct domains, complete with their own partially unique social and professional identities, bodies of knowledge, educational pathways, and spheres of technology. My analysis also reveals the extent to which these processes involved competing images of disciplinarity, as well as the ongoing negotiation of disciplinary settlements.

Part I – (The) Computer/Computing/Information Science(s): A Formative First Decade

Over roughly the span of a decade, the field that was growing up around terms such as "computer science" gained an impressive momentum. As historian Paul Ceruzzi explains, the

origins of the field can be traced back to at least the late 1950s, when "it was recognized that many topics that had much in common with each other (and all in common with the computer) were being taught in various departments in around most universities" (Ceruzzi, 1989, p. 266). This was also a time when rapid increases in the number of computer systems installed nationwide (and worldwide) was accompanied by rising demand for computer-oriented workers of all types. Recognizing and responding to these trends, a handful of forward looking individuals proposed the establishment of a new field or discipline, which the aforementioned Wienser dubbed the "communication sciences" and Fein called "computer science."

In subsequent years, an expanding cadre of actors and groups – many with ties to the mathematics community and the ACM – lent support to this discipline-building project, which increasingly went by the name of "computer science." In this section, I outline three different aspects of this historical trajectory, in roughly increasing order of importance. The first such aspect centers on early "top-down" efforts to define and name the proposed discipline. I then turn to ongoing moves to position the emergent field in a larger milieu of disciplines, especially in idealized terms. Third and finally, I discuss the establishment of computer science and related departments and curricula in the 1960s in order to shed light on the "institutionalization" of the field. This latter theme supports the argument that the rapid growth of computer science in the academic context was a pivotal development, both for the proponents of the new discipline and for other stakeholders, including electrical engineers.

Defining Computer Science as a Discipline[136]

Debates over the definition of "computer science" and closely related terms can be traced throughout the history of the field, even to the present. In fact, the long-standing lack of consensus on this matter suggests that the successful establishment and development of a discipline does not necessarily require widespread consensus about its precise definition or scope. Yet in order to gain a general sense for the evolving scope and orientation of computer science, it is worth reviewing some of the major definitions in play, especially in the field's formative first decade. These definitional efforts frequently involved processes of abstraction,

[136] The account presented in this section has some parallels with the prior efforts of Ceruzzi, who provides one of the better reviews of ongoing efforts to define "computer science" (Ceruzzi, 1989, pp. 265-270).

where attempts were made to capture the commonalities that united the full array of subjects, topics, and technologies that were at the heart of the proposed discipline.

As noted above, Wiesner made one of the earliest attempts at such a definition when he described the "communication sciences" as providing the appropriate mathematical and theoretical foundations for the study of complex computing, information processing, and communication systems, of both the natural and artificial variety (Wiesner, 1958). Fein, on the other hand, was likely the first author to proclaim the emergence of a new discipline called "computer science." And while his 1959 article hinted at the contours of the field via wide-ranging lists of potentially relevant topics and subjects, in 1961 Fein more succinctly defined the "computer sciences" as "the theory and practice of the design, programming, and application of computers" (1961a, p. 151). With a nod to Wiesner, he also suggested that the computer sciences were a branch of a proposed "supradiscipline" called the "computer-related sciences" or "synnoetics," which was more generally concerned with the properties and structure of "composite systems" (1961a, pp. 150-151).[137] Both Wiesner and Fein can be credited with recognizing some of the synergies that were forming between a wide range of fields, ranging from computing, cybernetics, and communication theory to parts of biology and even linguistics. And while the ambitious visions presented by these authors were perhaps ahead of their time, they helped set the stage for a larger movement that got underway in the 1960s.

Some of the first evidence for such a movement can be found in the *CACM*, which in 1960 featured a "Report on a Conference of University Computing Center Directors" (Morse, 1960). As the author of this report confidently explained, computer science was both a "new scientific field" and "discipline in its own right" (p. 520; 521).[138] Yet aside from listing a handful of courses and subjects being taught by university staff, the report was mostly silent on the definition and scope of the field. Mathematician George Forsythe, on the other hand, explained in a 1961 article that a new discipline of "Computer Sciences" was emerging. Describing the field in pluralistic terms, he indicated that the "theory of computer programming" was perhaps the "most important of the computer sciences," although he added that numerical analysis, data

[137] As Fein explained, composite systems could consist of various configurations of people, mechanisms, plants, animals, organisms, and/or "automata" (1961a, p. 150).

[138] It is worth noting that the terms "computing science" and "computer science" were used interchangeably in one section of the report. This flexibility of terminology closely parallels the use of terms such as "computing engineering" and "computer engineering" in the 1950s, as documented in the preceding chapter.

processing, and computer system design were also relevant to the "computer sciences" (Forsythe, 1961, pp. 177-178).

Definitional issues received further attention in 1963, when mathematician Saul Gorn described the development of "a new basic discipline" that he called the "computer and information sciences." The author also explained that one of the field's central concerns was the "'analysis' and 'synthesis' of 'mechanical languages' and their 'processors'" (Gorn, 1963, p. 150).[139] In addition to reflecting Gorn's research interests in the area of computer languages, framing the field in this manner placed implicit emphasis on the orientation of the field toward information and communication processes and systems. In a 1964 article, on the other hand, Thomas Keenan of the University of Rochester discussed the prior writings of Wiesner, Fein, and Gorn as he developed something of a "composite" definition for the new field. As Keenan explained, "the study of the organizational and structural properties of systems, arrays of symbols and mechanical languages which find their application in the processing and communication of information is at the heart of computer science" (Keenan, 1964, p. 206).[140]

As noted in previous chapters, through the 1960s "information," "information processing," and "information systems" were viewed as pivotal concerns for those working in many different phases computer field. This trend was also reflected in a preliminary ACM curriculum report published in 1965. In a section titled "Computer Science as a Discipline," the authors stated that "Computer science is concerned with *information* in much the same sense that physics is concerned with energy; it is devoted to the *representation*, *storage*, *manipulation* and *presentation* of information in an environment permitting automatic information systems" (ACM C^3S, 1965, p. 544). And in 1967, Forsythe similarly described computer science as "the art and science of representing and processing information and, in particular, processing information with the logical engines called automatic computers" (p. 3). Vladimir Slamecka of the George Institute of Technology, on the other hand, argued at a 1967 conference that "Information Science and Engineering" was a more accurate name for the field (Slamecka, 1968). Expressing dissatisfaction with prior definitions, Slamecka argued that the structure of the discipline was

[139] As Gorn explained, "The study of mechanical languages is concerned with the synthesis and analysis of systems of arrangements of symbols, and with the synthesis and analysis of processors which generate, recognize, translate, and generally interpret such systems in various ways" (Gorn, 1963, p. 151).
[140] At the end of his article, Keenan suggestively added that "[p]rogressive faculties now understand that computer science is more than FORTRAN programming" (Keenan, 1964, p. 209).

fundamentally based on theories of information, information processes, and information systems.[141]

But even as it was increasingly common to frame the emergent field as information-oriented, commentators such as Alan Perlis developed still other types of definitions. In a 1967 conference talk, Perlis echoed prior commentators when he explained that one principle goal of computer science centered on understanding "the organization and administration of information" (Perlis, 1968, p. 70). Yet he moved in somewhat different directions when he noted that "computer science is the study of the design, analysis, representation, and applications of *algorithms* on computers" (p. 70). And later in the same talk, Perlis echoed one of Forsythe's early comments when he suggested that "computer programming is at the root of computer science" (p. 76). While this array of characterizations perpetuated the image of a rather inchoate new field, they nonetheless informed ongoing efforts to map out and negotiate the disciplinary settlement of computer science.

Perlis addressed the issue from still another angle in a 1967 letter that he co-authored with Allen Newell and Herbert Simon, two of his colleagues at Carnegie Tech. Published in *Science* and addressing the question of "What is Computer Science," the letter started by with a seemingly straightforward argument: "There are computers. Ergo, computer science is the study of computers" (Newell, Perlis, and Simon, 1967, p. 1373). The authors went on to explain that this included all phenomena surrounding computers, including programs, algorithms, *and* hardware. On the one hand, Simon in part expanded on this line of argument in his suggestively titled 1969 tome, *The Sciences of the Artificial*. On the other hand – and as noted by Ceruzzi – the publication of texts such as Donald Knuth's *The Art of Computer Programming, Volume 1: Fundamental Algorithms* (1968) helped promote the notion that algorithms were a central topic of concern for computer scientists and programmers.[142]

---

[141] Still another definition along these lines was developed by the Department of Information Sciences at the University of Chicago: "The Information Sciences deal with the body of knowledge that relates to the structure, organization, transmission and transformation of information … This includes the investigation of information representation, as in the generic code or in codes for efficient message transmission, and the study of information processing devices, and techniques, such as computers and their programming systems" (Beckman, 1968, p. 40).

[142] The importance of the topic of algorithms admittedly has a longer history in the field. From about 1959 onward, for example, the *CACM* devoted an expanding number of pages to descriptions and discussions of algorithms. But as evidenced by the publication of Knuth's text and the remarks of commentators such

By the late 1960s, the definition of computer science and a series of closely related terms remained the subject of much discussion and debate. Yet there was growing consensus that a discipline had either formed or was well on its way to forming, and it was increasingly referred to as "computer science." Further, the dominant image of the field was largely based on a disciplinary settlement that encompassed algorithms, programming, information processing, and information systems, and that was grounded in theory and mathematics. In a sense, then, the term "computer science" was something of a misnomer, and a long string of commentators maintained that the emergent field was significantly independent from the actual machinery of computing.

Fein hinted at this theme in 1959 when he framed computing equipment as a "supplement" to university educational programs in the "computer sciences" (p. 11). And in 1961, he noted that more expansive terms such as "Computer-Related Sciences" were misleading because they overemphasized computer equipment (1961a, p. 161).[143] Forsythe similarly argued in 1961 that "the computer sciences are partly independent of actual automatic computers" (p. 178), while Atchison and Hamblen complained in 1964 that the term "computer science" was too "machine-oriented" (1964, p. 227). As the 1960s wore on, the apparent distance between the discipline and the machine only increased, leading Atchison to summarize in 1971 that "'[t]he pure computer scientists will probably move further and further from the machine itself and more and more into the theoretical aspects such as abstract structures of information and the theories of representation and transformation" (1971, p. 131). In many ways, this distancing was beneficial for computer science, as it insulated the field from rapid technological changes in computer technology. As I note below, courses and curricula became an important pathway for creating and maintaining space between the discipline of computer science and the rapidly changing state of the technological art. On the other hand, this trend also tended to expand the distance between the theoretical foundations of computer science and the more pragmatic application of computers to real world-problems, thereby opening up new opportunities for other actors and groups to assume leading positions in other spheres, including that of "hardware."

---

as Perlis, the view that algorithms were a central, defining concern of computer science did not gain significant traction until later in the 1960s.

[143] As noted above, Fein addressed this problem by coining and promoting an alternate term, namely "synnoetics."

Positioning and Settling Computer Science

While the development of top-down definitions for computer science is an important aspect of the present historical account, it is but one aspect of a larger discipline building process. In fact, if we take seriously Abbott's claim that "academic settlement involves a complex structure of relations with other disciplines" (2001, p. 141) we might expect to find extensive discussions about the position of computer science in a larger disciplinary milieu. As it turns out, many other commentators have uncovered such discussions and recognized their importance. As nicely summarized by engineer and historian Eric A. Weiss, for example:

> [I]n the late 1950s and early 1960s, the academic world was struggling with the question of where computing was to be fitted into its often hidebound structure. Was it a subdivision of mathematics or electrical engineering? Would it last or would it fade away? Did it have enough philosophical and intellectual content to be considered in any way a science in its own right? (1992, p. 76).

In this section I document ongoing efforts to address these types of issues, especially from the late 1950s to late 1960s. I place particular emphasis on how various actors and groups defined computer science in relation to a variety of other disciplines specifically, as well as to mathematics, the sciences, and engineering generally. In summary, my analysis brings into further relief the establishment and negotiation of a disciplinary settlement for computer science. I begin by looking at how these issues were addressed in rather abstract and idealized terms, and then turn to more pragmatic discussions about the "institutionalization" of the field.

To begin with, early commentators such as Wiesner and Fein framed the emergence of a new computer-related discipline as both drawing from and informing work in other fields. Wiesner noted a "close kinship" among many researchers working in diverse domains, ranging from engineering and the physical sciences to mathematics and even biology (1958). And while he admitted that engineers and inventors played an important role in the field of "communication sciences," he emphasized the need for further theoretical developments, which engineers alone could presumably not provide. Fein similarly noted the importance of establishing theoretical foundations for the proposed discipline, and he explicitly described "computer science" as "inter-

disciplinary." He added that fields such as computer science, mathematics, and library science "are both disciplines in themselves as well as service tools to other disciplines" (1959, p. 11).[144]

However, commentators such as Gorn argued that distinguishing the field from its neighbors required a more substantive rationale than simply suggesting that it was somehow "interdisciplinary." He therefore argued in 1963 that "essential differences in attitude and essential differences in background requirements" (p. 153) set the so-called "computer and information sciences" apart from other fields. Keenan worked in similar directions when he noted that computer science was distinguished from other fields by its distinct "intellectual orientation" (1964, p. 206). He more specifically explained that:

> The physicist is interested in the basic thermal or electromagnetic properties of the materials; the electrical engineer is interested in the behavior of these as components in an electrical circuit; the computer scientists is interested in [the] performance of these components in large arrays and their effect on the design of information processing equipment to yield greater utility or improved economics (p. 206).[145]

In summary, these authors were drawing very distinct boundaries between the engineers who were interested in the electronic components of computers and the "computer and information scientists" who grappled with higher-level matters such as logical design and machine organization. To put it another way, they argued that engineers and computer scientists saw very different things, even when they were looking at ostensibly the same technologies. Such passages hint at the extent to which the establishment of a given disciplinary settlement may extend beyond the drawing of epistemological boundaries to include claims about the desirable attitudes, methods, and even "culture" of those associated with a given field. In fact, these latter claims become particularly important when two or more disciplines maintain overlapping interests in a common domain of technology – in this case, computers and information systems.

Yet even if those working in the new discipline possessed a unique outlook, it was clear that much activity in this emergent domain was firmly situated at the intersection of

[144] In his 1961 article on the "supradiscipline" that he called the "computer-related sciences" or "synnoetics," Fein once again emphasized that the proposed discipline was "a tool for practitioners in other disciplines and … a discipline in itself" (1961a, p. 159). Such remarks hint at the author's intended analogy to other fields, such as mathematics.

[145] Keenan's example was likely influenced by Gorn's 1963 article, which featured a very similar passage.

mathematics, science, and engineering. This posed various challenges as commentators worked toward an appropriate name and disciplinary identity for the field. Forsythe, for example, stated in 1963 that "Computer Science seems to be about halfway in spirit between Humanities and Sciences, and Engineering" (p. 175). More specifically, he explained that concerns about design pointed to a location for computer science within engineering, while "the abstract nature of computing" suggested a close affiliation with the humanities and sciences in general, and mathematics, physics, and philosophy, in particular. And in 1967, Forsythe summarized that "computer science is in part a young deductive science, in part a young experimental science, and in part a new field of engineering design" (p. 4). Yet despite his acknowledgment of the field's hybrid identity, Forsythe ultimately expressed sympathies with situating computer science within university schools of arts and sciences, a point to which I will return.

Much discussion at the 1967 Stony Brook conference on University Education in Computing Science similarly centered on the multifaceted character of "computer science" or "computing science." For instance, Perlis explained that much of the "dilemma" of computer science centered on the field's constitution as "part mathematical science and part mathematical engineering" (1968, p. 71). Yet the suggestive title of his talk -- "Computer Science is Neither Mathematics nor Electrical Engineering" – made it clear that Perlis viewed the emergent discipline as sufficiently distinct and independent from other fields.

Slamecka offered additional insights on these issues in his presentation at the same conference on the topic of "Information Science and Engineering" (1968). As Slamecka explained, this field's engagement with "theories of information" and "information processes" aligned it with science, while its concern with information systems linked it to engineering. The author therefore concluded that "[t]he structure of the discipline … straddles and units [sic, unites] (rather than distinguishes) science and engineering" (p. 90). Slamecka went on to describe Information Science and Engineering as the first "metascience" to "concern itself actively with the synthesis of the various disciplines of science and engineering" (p. 90). While Slamecka's argument was substantially open to critique, it once more hinted at persistent questions about whether such a discipline could find a secure place in the "hidebound" structure of the university, where the boundaries between science and engineering were often deeply inscribed in the organization of colleges, departments, and degrees. To put it another way, to

what extent was it possible to establish and maintain the disciplinary settlement of a field that to some extent lived across the boundaries of science and engineering?

In their passionate defense of computer science published in *Science*, Newell, Perlis, and Simon responded to other points of objection regarding the identity and position of the proposed field. For instance, in addressing the critique that "computer science is a branch of electronics (or mathematics, psychology, etc.)," the authors admitted that many of the "phenomena of computers" were indeed relevant to other sciences. Yet they argued that "all of the phenomena of computers are not subsumed under any one existing science" (Newell, Perlis, and Simon, 1967, p. 1374). As suggested by such remarks, the authors viewed computer science as a unifying discipline for the study of diverse phenomena. They also responded to the claim that "computers belong to engineering, not science" by arguing that the computers "belong to both," and they added that "[t]ime will tell what professional specialization is desirable between analysis and synthesis, and between the pure study of computers and their application" (p. 1374). However, the authors' preference for framing the emergent field in scientific rather than engineering terms was clear, especially in light of both their choice of terminology and major points of argument.

As my analysis suggests, establishing convincing arguments for the identity and position of computer science was viewed as an important factor in building the field's legitimacy, especially in the academic context. Further, many of these arguments spoke to pressing concerns about what participants at the 1967 Stony Brook meeting referred to as the "intellectual respectability" of the proposed discipline. In 1968, Forsythe quipped that "[i]n a purely intellectual sense, such jurisdictional questions are sterile and a waste of time" (p. 455). He was forced to admit, however, that these issues were of "great importance" when linked to more pragmatic concerns, such as the organization of universities or the administration of research grants. And indeed, the following section reveals that discussions about the *ideal* or *preferred* position of computer science in the midst of science, mathematics, and engineering were increasingly overshadowed by its *actual* location within various institutions, especially from the mid-1960s onward. To put it another way, the problem of disciplinary settlement was ultimately not addressed via top-down decree, but rather by bottom-up and context-specific processes.

Instituting Computer Science – Departments and Programs

In Slamecka's view, realizing the new discipline that he called "information science and engineering" demanded "a patient and imaginative examination of the fabric of science for the purpose of structuring a utilitarian, logically consistent subset of knowledge which can be transmitted" (1968, p. 92). As the preceding analysis suggests, many of the proponents of computer science were exploring this idealized pathway as they worked to define and position the emergent discipline, often in rather abstract terms. It was a task made even more challenging by the persistent framing of computer science or information science as mathematical and scientific, yet also closely linked to engineering, design, and technology.[146]

Slamecka also explained that carefully defining and positioning the discipline in a top-down manner was crucial in setting the stage for the development of "goal-oriented" educational programs. However, he admitted that "the structure of educational programs is not necessarily identical with the structure of the discipline" (p. 91). As I document below, this potential schism between the ideals and realities of discipline building was abundantly clear by the time of Slamecka's remarks. In fact, I contend that the ongoing proliferation of computer science departments and programs through the 1960s lent crucial support to the momentum and legitimacy of the discipline, in spite of persistent debates over its definition, position, and "intellectual respectability."

Tentative calls to institutionalize computer science – or some variation thereof – first surfaced in the late 1950s. Wiesner (1958), for example, advocated the establishment of university research centers, with his own Communication Sciences Center at MIT as a working model. Fein (1959), on the other hand, soon thereafter called for the development of "graduate schools of computer sciences," which in ideal form consisted of multiple departments, "integrated" instructional programs, and dedicated faculty members. While clearly ambitious, his recommendation strategically placed computer science high in the university structure, thereby sidestepping thorny questions about where computer science should be located with respect to other, pre-existing schools or departments. As noted in the previous chapter, Fein also pitched the idea of establishing departments of computer science to various schools in the early 1950s,

[146] Using the language developed by Science and Technology Studies scholars, one might view the emergence and development of computer science as an important chapter in the history of twentieth-century "technoscience."

although his proposals were initially met with much skepticism.[147] Hence, other actors and groups – many with backgrounds in mathematics, as well as close ties to the academy and the ACM – largely picked up where Fein left off.

Saul Gorn stepped forward as another outspoken proponent of academic departments of computer science. In his 1963 article on the "Computer and Information Sciences," he framed the issue in historical terms by acknowledging that many mathematics departments and schools of engineering were important early sites for computer-oriented research and education. Yet he went on to ask: "[C]an such a rapidly growing discipline with clearly different interests and requirements continue indefinitely to be carried in an essentially different environment where accident has caused it to gestate?" (p. 155). While Gorn's use of the term "accident" was clearly a strategic overstatement, the author used an even more dramatic maternal metaphor to describe the discipline's future prospects:

> Would not the mother discipline of the particular environment eventually have to limit the nourishment it can afford to provide to such a growing child if it is not to limit its own growth and development? In such a case the new discipline would have to be able to fend for itself (p. 155).[148]

The author's message was clear: both mathematics and electrical engineering departments were significantly at cross-purposes when it came to promoting computer-related research and education. Gorn forecasted the continued expansion of the Computer and Information Sciences, especially as the demand for computer-oriented courses and expertise spread into other disciplines. He concluded by arguing that this process of growth and development "will, willy-nilly, have to stabilize, and when it does there will be a completely new department responsible for the new discipline" (p. 155).

Here we find an awareness of the close links between disciplines and departments, at least in the American academic structure. And while Gorn faced significant challenges in realizing such a department at the University of Pennsylvania – a point to which I will return –

[147] Some of the difficulty that Fein faced as he worked to realize his vision was likely linked to his status as an independent consultant whose background and interests crossed the boundaries of science and engineering, the academy and industry, and even the ACM and the IRE.

[148] In a 1963 article, Forsythe similarly described "the birth of a coherent body of technique, which I call computer science" (p. 169). While feminist theorists of technology have suggested that creating technologies can provide men with surrogate birthing or mothering experiences, the remarks of these commentators suggest that this line of argument might be extended to the study of disciplines and processes of disciplinary formation.

his colleague George Forsythe had better success at Stanford. In fact, Forsythe played a leading role in the establishment of a Division of Computer Science within the Mathematics Department in 1961, transformed in 1965 into an independent Department of Computer Science, one of the first of its kind in the nation (Lee, 1995, p. 314; Knuth, 1972, p. 722). For Forsythe, establishing such departments came with many advantages, such as allowing for faculty salary scales that were more competitive with industry (Forsythe, 1963, p. 174).[149] But even more importantly, he explained that such departments provided the necessary freedom to experiment with new curricula and degrees (p. 174).

And while Forsythe also expressed sympathies with situating computer science departments within schools of arts of sciences, the aforementioned Stony Brook conference revealed wide variations in how computer science was being instituted at different schools. According to one workshop report, computer science was variously being realized as a:

- major within mathematics or electrical engineering;
- separate department within a college of liberal arts and sciences or a college of engineering;
- separate department spanning colleges;
- separate department in the graduate school;
- separate school entirely; or,
- single unit combined with computer services and reporting to a vice-president or provost (Atchison, 1968, p. 171).

The report went on to explain that the workshop participants largely agreed that computer science "is now accepted as a separate academic discipline," and they explained that this pointed toward the establishment of a separate "organizational entity" of some kind (p. 175). And while this group failed to reach consensus on what this entity should look like or where it should be located, establishing computer science majors within other departments was clearly the least desirable of these approaches. The workshop report also indicated that appropriate solutions should be worked out on a school-by-school basis, with sensitivity to local conditions.

[149] Still others suggested that the independence of computer science would allow work in the field to be evaluated "on its own terms" (Morse, 1960, p. 521). Along similar lines – and as noted in the previous chapter – Forsythe similarly argued around this same time that courses in certain areas of computing might be developed more rapidly if the associated instructors were "not judged primarily by the standards of any existing department" (1961, p. 180).

On the surface, it might seem that these diverse manifestations of computer science at different institutions was not a particularly strong selling point for the proposed discipline, and 1960s-era statistics about the precise number of institutes, schools, or departments of computer science are difficult to come by. However, through the 1960s an expanding assortment of commentators and analysts were documenting impressive expansions in the number of computer science degree programs and options. Forsythe pointed in this direction when he noted in 1963 that "integrated" programs in computer science had been established in at least a dozen universities, although the structure and naming of these varied widely (p. 175).[150] And as mentioned in the previous chapter, around this same time Tompkins (1963) identified a roughly similar number and range of programs.

A more comprehensive survey of university computing center directors conducted in 1963 revealed that 28 of 93 responding North American institutions offered some type of computer-oriented degree programs, while another 18 were planning such programs (Atchison and Hamblen, 1964, p. 226). This report also documented wide variations in the naming and institutional location of these programs, and the authors even stated that the "profusion of degree programs and names for programs makes it abundantly clear that a precise discipline has not yet crystallized" (p. 227). Yet the authors also noted the widespread use of the phrase "computer science," and the survey data indicated that a large majority of respondents preferred this term as a moniker for academic programs. Alternatives such as "Information Processing" and "Information Science" were rated a distant second and third in popularity.[151] A 1965 report provided further evidence for these trends by indicating that more than 15 schools were offering doctorate degrees in computer-science or related areas, a total of 30 or more schools offered

[150] More specifically Forsythe pointed to a number of interdepartmental graduate degree programs, including "Systems and Communication Sciences" at Carnegie Institute of Technology, the aforementioned "Computer and Information Sciences" program at the University of Pennsylvania, and a "Communication Science" option at the University of Michigan. Forsythe also indicated that separate departments of Numerical Analysis were established at the Universities of Arizona and Wisconsin. At still other schools, computer science was linked to divisions of applied mathematics (1963, p. 175).

[151] Given that this particular survey polled university computer center directors – who tended to have both closer ties to mathematics departments and stronger sympathies with the emergent field of "computer science" – it is not entirely surprising that respondents preferred non-engineering program names. Yet in spite of its popularity, the authors complained that the term "computer science" was not ideal: "It is certainly true that this term has many shortcomings in appearing to be too machine-oriented but then too so does the term Association for Computing Machinery fall far short of describing our professional organization" (Atchison and Hamble, 1964, p. 227).

master's degrees, and at least 17 had similar options at the baccalaureate level (ACM C^3S, 1965, pp. 544-545).

A report published in 1967 provides additional evidence regarding both the growing number of computer-oriented degree programs and the increasing popularity of the "computer science" moniker (Hamblen, 1967). For starters, the report estimated that for the 1964-1965 academic year, about 143 distinct degree programs and options in computer science and related areas were offered nationwide at the baccalaureate and graduate levels.[152] More than 40% (58 of 143) of these were specifically dedicated to Computer Science or Information Science, and many of these were affiliated with departments of the same name (p. 66). These were impressive statistics, especially since the first programs of this type were established just five or so years prior. Computer Science options in Mathematics and Electrical Engineering claimed roughly another one-third of all programs, with a further respective breakdown of 17% (24 of 143) and 13% (19 of 143). The remaining programs and options were offered in other departments or as interdisciplinary programs, with titles ranging from "Business Data Processing" to "Systems and Communication Sciences" (pp. 66-67).

This same report also offered forecasts for 1968-1969. First, the survey pointed to a rapid potential for expansion in the total number of computer science and related programs, which were expected to grow from 143 to 226 in the span of just four years (p. 66). Further, dedicated Computer Science programs were expected to dominate this trend, with survey data suggesting that as many as 166 new computer science degree programs at the undergraduate and graduate levels were in active development. The number of computer science and related options in electrical engineering, on the other hand, was expected to increase from a total of 19 to just 23, while similar options in mathematics were expected to jump from 24 to 36 (p. 66). While these data were based only on the *planned* actions of responding institutions, the trends were unmistakable. In fact, these data clearly support the argument that much of the momentum of the computer science movement was being driven not by "top-down" definitional efforts, but rather by the "bottom-up" development of new academic departments and dedicated degree programs.

[152] Note that these data represent the total number of separate degree options, not separate schools. Hence, a school offering bachelor's, master's, and doctoral degrees in computer science would add three to this total, not one. Unfortunately, the original report does not provide enough original data for a more fine-grained analysis.

In addition to creating new patterns of disciplinary settlement for computer science, these departments and programs helped nurture the field's image as an independent discipline.

Instituting Computer Science – Courses and Curricula

While the growth of computer science departments and programs was a key trend in the 1960s, I have largely sidestepped the content of the associated curricula. Further, published discussions of computer science and related courses and curricula were sparse, especially in the late 1950s and early 1960s. This situation started to change rapidly, especially through the efforts of the Curriculum Committee on Computer Science, or C^3S. Originally established as a subcommittee of the ACM's Education Committee in 1962, the ACM recognized the C^3S as an independent committee in 1964 under the leadership of mathematician William Atchison (Atchison, 1985, p. 328; ACM C^3S, 1968, p. 152).[153]

While the early activities of the C^3S were primarily centered on organizing panel sessions at conferences and meetings, the growing prominence of this group was reflected in the April 1964 issue of the *CACM*, which featured a series of eight short articles on the topic of "Computer Science Curriculum." In addition to the aforementioned background piece by Keenan and survey article by Atchison and Hamblin, six of the papers in the special issue presented detailed descriptions of various courses that "could form a basis for a computer science curriculum" ("Computer Science Curriculum," 1964). And while a comprehensive review of this set of papers is beyond the scope of my analysis, a brief juxtaposition of two of these pieces provides important insights regarding the ways in which different actors approached the development of computer-oriented educational programs.

The two papers in question were focused on the development of a series of courses in the area of logic, which has long been viewed as a foundational topic for large swaths of work in the computer field. In fact, it was clear by the 1960s that most computer-oriented educational programs at the baccalaureate and graduate levels should provide students with some familiarity with this subject. In outlining a series of logic courses for what he called "the computer sciences," Purdue mathematician Robert R. Korfhage took a mathematical and theoretical

[153] Atchison held a Ph.D. in mathematics from the University of Illinois, and from 1955 to 1966 he was affiliated with the Georgia Institute of Technology. His roles at Georgia included head of the school's computer center, acting director of the School of Information Science, and professor of Information Science. In 1966 he became the director of the University of Maryland's Computer Science Center (Lee, 1995, pp. 46-48).

approach to the topic (1964). His proposed four-course sequence started with Introduction to Logic and Algorithms, followed by Logical Design, Mathematical Logic, and Computability and Algorithms. As suggested by these titles, the author's proposed content for three of the four courses was primarily oriented toward mathematics and algorithms, while computer design topics were bracketed off in the separate Logical Design course that the author largely failed to describe. In fact, he even argued that teaching the "exact physical form" of computer logic elements was best left to the departments that had the requisite expertise in electronics and physics (p. 216).

By contrast, David E. Muller of the University of Illinois-Urbana wrote on "The Place of Logical Design and Switching Theory in the Computer Curriculum" (1964). In his four-course sequence, Muller emphasized topics such as computer organization, switching theory, logical design, and system design. The author's proposal also avoided the term "computer science," and it largely framed mathematical logic as a prerequisite topic that was applied and extended in the proposed course series. As nicely summarized by reviewer Harvey L. Garner of the University of Michigan, "[t]he courses presented by Prof. Muller form an excellent and well-organized sequence which places logical design and switching theory in the context of *digital computer engineering*" (Garner, 1964, p. 224, my emphasis). However, Garner complained that some of the more important topics highlighted by Muller – such as switching theory and automata theory – were too deeply buried in a program ostensibly dedicated to "digital computer engineering."

While perhaps a fair critique, Garner's comments hinted at the conflicts of "intellectual orientation" that divided the more machine- and engineering-oriented proponents of computer education from those who took mathematics, algorithms, or even information as the field's common denominator. This schism was also evident in a special panel presentation on "Computer Science Curriculum." Held at the ACM's national conference in early 1964, the panelists discussed programs that were proposed or getting underway at the University of Maryland, Purdue, and Case Institute of Technology. The Computer Sciences program at Purdue stood at one end of the spectrum of possibilities given that it was described as "heavily mathematically oriented" – perhaps not surprising since that school's newly formed Department of Computer Science was located within a Division of Mathematical Sciences (Conte, 1964).[154]

[154] As explained in the program description, "Computer Sciences as a discipline is more closely related in methodology and philosophy to mathematics than to any other discipline. This belief is strengthened by

A group at Maryland, on the other hand, was developing a more middle-of-the-road "Computer Sciences" program that was largely based on mathematical foundations and focused on algorithms (Schweppe, 1964). However, this program did allow students to select distinct mathematics, physics, or electronics tracks, and the authors noted that electrical engineers had contributed to the development of the program (p. L1.1-2).

The program at Case, on the other hand, tilted strongly toward the engineering end of the spectrum. As indicated in a summary description authored by mathematics professor Richard Varga (1964), early efforts to develop computer-oriented courses and programs within the school's mathematics department were problematic because they demanded that students be trained as mathematicians. The author therefore argued that "the activities of the computer technologist at the undergraduate level are far more closely related to, and have more in common with, the training of engineers" (p. L1.3-1). Emphasizing the value of taking a systems-oriented approach to educating future computer experts, the report described a program that coupled an engineering core curriculum with two program options, one dedicated to "computer engineering" and the other to "numerical methods and programming" (p. L1.3-1). This was one of the first programs of its type to offer an option that was explicitly termed "computer engineering."[155]

As the development of computer science and related curricula gained momentum in subsequent years, the engineering-oriented curricula presented by Muller and Varga increasingly looked like an exception in an expanding sea of actual and proposed programs, many of which were oriented toward mathematics and theory. In another article in the same issue of the *CACM*, for example, Forsythe provided a partial sketch of an "An Undergraduate Curriculum in Numerical Analysis" (1964a) that was largely focused on mathematics and programming. And in 1964, the Committee on the Undergraduate Program in Mathematics (CUPM) of the Mathematical Association of America (MAA) published a short report titled "Recommendations on the Undergraduate Mathematics Program for Work in Computing" (Committee on the Undergraduate Program in Mathematics, 1964). Authored largely by mathematics professors – including notable figures such as Berkeley's A. H. Taub – the report insisted that "responsibility for research and training in Computer Science should be closely linked to mathematics" (p. 2).

---

the clear evidence that the leaders in research and development in computing today were largely trained as mathematicians" (Conte, 1964, p. L1.2-1).

[155] As outlined in the report description, students pursuing the computer engineering pathway took courses on electronic circuits, circuit analysis, and digital computer design (Varga, 1964, p. L1.3-1).

Perhaps not surprisingly, the courses and curriculum outlined in the report were mathematically oriented, and the authors explicitly argued that computer science researchers "must think like mathematicians" (p. 2).

The ongoing movement of computer science away from the domain of engineering and "hardware" was also evident in a 1965 report by the ACM C^3S titled "An Undergraduate Program in Computer Science – Preliminary Recommendations" (ACM C^3S, 1965). As noted above, the authors of this report were significantly concerned with defining the field, yet they also noted "the fact that computer science is now a distinct academic discipline is demonstrated by the rapidly increasing number of colleges and universities which have established departments of computer science" (p. 544). Much of the report was dedicated to describing sixteen courses leading to a proposed undergraduate computer science degree. In most general terms, the curriculum was oriented toward algorithms, programming, and mathematics, although topics such as computer organization, logic design, and switching theory were included to varying extents. The report also noted that an "electronics" option might be offered to students, yet the authors bracketed off a course on electronics as one of many optional electives for computer science majors. In an important sense, reports such as this one provided a partial description of – and prescription for – the disciplinary settlement of computer science.

Another notable development in the educational arena was the aforementioned 1967 Stony Brook conference, where numerous presentations and workshops were focused on discussing the development graduate-level degree programs in computer science and related areas. But even more important was the subsequent publication of the C^3S's influential *Curriculum 68* report (ACM C^3S, 1968). In contrast to both the committee's earlier recommendations and the dominant tone of the Stony Brook conference, the report explained that debates over the existence of "computer science" were being replaced by discussions about "what this discipline should be called and what it should include" (p. 153). And with regard to implementing computer science degree programs, the authors stressed the importance of establishing "independent academic units" and involving dedicated faculty "who consider themselves computer scientists" (pp. 166-167). Such statements once again pointed to the recognized strategic importance of establishing both a distinct identity for computer scientists and an independent institutional niche for the new field.

Curriculum 68 also provided detailed recommendations for computer science programs and curricula, and these were largely organized around three major subject areas, namely information structures and processes, information processing and systems, and methodologies. The "mathematical sciences" and "physical and information sciences" were also mentioned, but only as subjects of potential relevance. The report included detailed descriptions and outlines for 22 specific courses, as well as overall curricular recommendations at the undergraduate and graduate levels. And as Ceruzzi (1989) has noted, the report dispensed with "hardware" courses almost completely, preferring instead "an algorithmic approach and an emphasis on language and data structures" (p. 268).

And while perhaps the authors of the report probably did not fully realize it at the time, the publication of *Curriculum 68* was the outcome of a decade-long discipline-building project. Setting aside questions about the preferred definition and position of computer science, the report pointed the way toward the continued expansion of the field through ongoing, bottom-up efforts to develop courses and curricula. The importance of this report for the development of computer science has been widely recognized by commentators. Yet skeptics and critics continued to raise potent questions about the discipline-building project of computer science through the 1960s and beyond, even as the proliferation of computer science departments and programs served as a potent demonstration of the field's impressive institutional success. In fact, the following section turns to the efforts of a growing band of electrical engineers, whose alternative vision for computer science and its associated educational programs started to gain traction in the latter part of the 1960s. Yet on the whole, engineering educators reacted rather slowly to the emergence of computer science. And when they finally did respond, it was increasingly evident that their agenda was based on a partially distinct image of disciplinarity.

Segue – On the Boundaries of Computer Science and Engineering

In the preceding section, I examined how various actors and groups coalesced around the discipline building of project of "computer science." But as IBM's Frank Beckman noted at the 1967 Stony Brook conference, the "enormous range of intellectual activity in computing" tended to raise skepticism about such an effort, especially among those who were unfamiliar with the computer field (Beckman, 1968, p. 45). Yet even insiders expressed concerns about the identity and position of computer science. Stanley Gill of Imperial College (UK), for instance, noted at the same 1967 meeting that computer science remained somewhat short on theory (Gill, 1968). He therefore indicated that "information engineering" was probably a more appropriate descriptor for the field, and he explained that the appeal of the term "computer science" was likely linked to both the "glamour" of the word "science" and the less desirable popular association of engineering with "hardware" (p. 117). And while Gill implicitly lent support to the science-engineering divide by framing information engineers as "practitioners" and computer scientists as "theoreticians," he echoed commentators such as Perlis when he argued that computer science was neither a branch of electrical engineering nor mathematics.[156] In fact, he asserted that computer science should be viewed as a "new profession, based on a new discipline" (p. 121). As I discuss in more detail below, this conception of the field clearly bounded it off from other professions, most notably engineering.

Anthony Oettinger is another important actor in this story, and one might initially suspect that he favored the development of an independent discipline of computer science. After all, he followed Forsythe as the President of the ACM from 1966 to 1968, and he held the title of Professor of Linguistics and Applied Mathematics at Harvard University. Yet in addition to boasting a Ph.D. in Applied Mathematics from Harvard, Oettinger held an A.B. in Engineering Sciences and Applied Physics from the same school, and he was a senior member of the IEEE

[156] Gill even stated that the central importance of algorithms in computer science provided the field with "more internal coherence than electrical engineering" (Gill, 1968, p. 120). He also emphasized that computer science was no more of a "hotch-potch" than electrical engineering, and he snidely added: "Well, if electrical engineering is looking for a place in the sun, I wish it luck. But don't let it mess up computer science in the process" (p. 120). These comments were largely a retort to a talk by electrical engineering Lotfi Zadeh, who I discuss in substantial detail below.

("Anthony Oettinger's Home Page," 1998).[157] Like many before him, Oettinger's status as a "hybrid" actor provided him with a unique outlook that potentially put him at odds with some of his peers. In fact, Oettinger acknowledged in a 1967 talk that "the views I hold personally and some of the views that I feel that I ought to express, say in my official capacity as president of the ACM, are not always the same" (1968a, p. 27). Given such tensions, many of Oettinger's views on the emergence of computer science were deeply ambivalent, yet also very insightful.

Oettinger's concerns about the field of "computer science" were evident early in his Presidential tenure. In a 1966 letter to the ACM membership, for example, he noted that there were "weaknesses in the position of computer and information science as a new discipline" (1966b, p. 838). He more specifically pointed to a lack of "reliable descriptive data concerning the scope of computer science, education, and industry" (p. 838), and he identified a series of criticisms about the perceived value and necessity of establishing associated departments and educational programs. Paraphrasing what he described as a persistent "misconception," Oettinger added that "[c]omputer science is not a coherent intellectual discipline but rather a heterogeneous collection of bits and pieces from other disciplines" (p. 839). He went on to urge those sympathetic with the field to bolster its position by carefully identifying and describing its unifying themes and core subject matter.

Yet it was increasingly clear that Oettinger was not willing to participate in such a project. In fact, he delivered two talks in 1967 that revealed his ambivalence toward computer science. To begin with, he complained that the term "computer science" was something of a misnomer. "At the very least," he explained, "the title should be Computer Scientist *and Engineer*" (1968, p. 28, author's emphasis). In line with this remark, Oettinger spent considerable time discussing how "computing" was related to mathematics, science, and engineering. For example, he argued in a 1967 talk that the field was a science only to the extent that mathematics was a science (p. 604). And elsewhere, he noted that adopting a scientific identity for a large segment of the computer field was often linked to concerns about respectability and status, even though "much of what we do and a great deal more of what we should be doing and encouraging our students to do is, in fact, the practice of a brand of engineering" (1968, p. 28). As suggested by Oettinger's remarks, the proponents of computer

[157] In fact, a biographical sketch of Oettinger that was published in 1966 indicated that he was a member of the IEEE since 1947, but it is not clear whether he was a member of the AIEE and/or the IRE before the two groups merged (Oettinger, 1966a).

science generally preferred to adopt a scientific rather than engineering identity. This was not entirely surprising, given that many of these actors had backgrounds or academic appointments in mathematics or the physical sciences. Further, the cultural cachet of science was significantly higher than that of engineering, especially by the 1960s, while the formative image of computer science as an independent field made it look more like an aspiring scientific discipline rather than a branch of the engineering profession.

However, Oettinger nonetheless insisted on framing the computing field as a broad spectrum of activity, with mathematics and engineering largely defining its poles. "Whatever it [computer science] is," Oettinger explained, "on the one hand it has components of the purest of mathematics and on the other hand of the dirtiest of engineering" (1967, p. 605). Touching on similar themes in another talk, he worked to put a positive spin on this characterization:

> We, as a profession, as a group, happen to have been born at the time when, in many other areas, science and technology have just become fused. And so, rather than bemoaning the view that somehow we are neither fish nor fowl and thereby befoul the purities of science with the dirt of engineering and contaminate healthy robust engineering with pallid theoretical considerations, we should think of ourselves as the vanguard of the new outlook (1968, p. 34).

Such remarks paralleled the remarks of commentators such as Slamecka, who described "information science and engineering" as a "metascience" that united science and engineering. Yet as suggested by Oettinger's clever characterization, significant social, intellectual, and even cultural barriers stood in the way of realizing this ambitious vision for a combined field of computer science and engineering.

Oettinger further parted ways with many of his colleagues by once again questioning the value of establishing departments of computer science. In fact, he argued that the ongoing proliferation of such departments was largely the result of disciplinary politics and "rebellion."[158] As Oettinger explained:

> [I]n most universities computer people have had to rebel. They have had to rebel against archaic engineering schools that have just barely forgotten rotating

[158] Somewhat ironically, Finerman noted at the 1967 Stony Brook conference that many of the objections put forward by skeptics of computer science resulted from "political rather than intellectual factors" (1968, p. 196). As suggested by the remarks of Oettinger and Finerman, complaints about "political" motivations came from all sides of this story.

> machinery and maybe will tolerate a transistor or an integrated circuit but that certainly do not see that a machine … needs software to run. They have also had to get out of the clutches of mathematics departments that regard anything that is not completely pure, rigorous, and formal as some form of depravity (1967, p. 606).

As suggested by such remarks, the rebellion described by Oettinger came with many potentially negative consequences. More specifically, it was cutting off the field's abstract roots in mathematics, as well as its more pragmatic connections with physical machines, design methods, and pragmatic engineering criteria.

Further, Oettinger suggested that engineering departments were losing a rapidly growing sub-branch of engineering, while mathematics departments were losing the many students who were forced to seek computer-oriented degrees elsewhere. The larger message was clear: the secession of computer science was a negative development for all of the involved disciplines. Further, Oettinger suggested that working to redress this fragmentation might point toward better computer technologies, especially by both improving the "complementarity" of hardware and software development and linking design with intended applications (1967). However, the author placed particular emphasis on the value of an engineering outlook, and he enthusiastically praised the emergence of the concept of "software engineering."[159] Oettinger also argued that large, complex programming projects were best viewed as "major engineering problems" (1967, p. 606), and elsewhere he noted that engineering should be making moves into the domain known as computer science.

Oettinger's views were bold, especially given that he was delivering them to audiences that were primarily comprised of mathematicians and self-identified computer scientists. Yet despite his numerous insights and artful presentation, Oettinger's ambivalence limited his ability – and probably also his inclination – to effect change. In pragmatic terms, he concluded: "I am forced to split my mind and say that it is an intellectual mistake to have departments of computer science, while I believe there is no real tactical alternative to having them" (1968, p. 28). Yet in somewhat more idealistic terms, he argued that computer people should work to "infiltrate both

[159] This was one of the first times that the term "software engineering" appeared in publication. Just a few months prior, an employment ad from the Foxboro company sought "software engineers" (Foxboro, 1967). As I discuss in more detail below, software engineering emerged as an increasingly important site of negotiation for the overlapping interests of computer scientists and engineers, especially in the 1970s.

departments [of mathematics and electrical engineering] and take them over. From a dreamy missionary point of view, that's the goal. Meanwhile, we are lucky if we survive in their presence" (p. 35). In the end, however, it was clear that Oettinger – suspended as he was between the ideal and the actual – was largely content to "watch from the sidelines" (p. 34).

In subsequent years, other prominent actors such as Richard W. Hamming raised similar concerns. A mathematician, Bell Labs researchers, and former ACM President, Hamming was a well-known figure in the computer field by the late 1960s. In fact, he received the ACM's prestigious Turing Award in 1968, and he devoted much of his award lecture to arguing that "more than the usual engineering flavor be given to computer science" (1969, p. 3).[160] Building this argument, Hamming made an "arbitrary distinction" between science and engineering by indicating that the former was focused on "*what* is possible," while the latter was concerned with "*choosing*, from among the many possible ways, *one* that meets a number of often poorly stated economic and practical objectives" (p. 5, author's emphasis). Following this line of reasoning, he suggested that "computer engineering" was probably a more accurate label for the field than "computer science," although he cautioned that he was not advocating such a name change (p. 5). However, Hamming followed Oettinger by promoting the view that "training in software be given a more practical, engineering flavor" (p. 10). He also critiqued computer science and its educational program for being too strongly oriented toward mathematics.

In an important sense, commentators such as Oettinger and Hamming were developing critiques of computer science that were primarily aimed at the field's insiders. In fact, many of their views were outlined in ACM publications, which was increasingly a *de facto* locus of activity for the proponents of both computer science generally and computer science education specifically. But in calling for the injection of more engineering flavor into computer science, these authors were ultimately either ambivalent or undecided about the position of computer science in a larger disciplinary milieu. Further, they seemed to lack an awareness of the partially distinct images of disciplinarity that tended to prevail in science versus engineering.

A growing cadre of electrical engineers, on the other hand, entered this debate beginning in the mid 1960s. And in contrast to Oettinger's idealistic suggestion that computer people should "infiltrate" electrical engineering and take it over, these engineers were working to

[160] In the same year that he won the ACM's Turing Award, Hamming was honored as an IEEE Fellow. These honors hint at the extent to which Hamming's work crossed the boundaries between mathematics, science, and engineering.

transform their own field from the inside out. In fact, they initially asserted that they had both a historical right and a contemporary responsibility to develop computer-oriented programs and courses, including in the domain of “computer science.” And perhaps not surprisingly, the contentious nature of these types of claims led these actors to make key adjustments in their agendas and strategies.

Part II – Shifting Disciplinary Images: From Computer Science to Computer Engineering in Electrical Engineering

Bringing Computer Science Into the Fold: Lotfi Zadeh at Berkeley and Beyond

As discussed in Chapter 2, in a 1950 article Lotfi Zadeh urged his electrical engineering colleagues to develop the necessary mathematical and theoretical expertise so that they could take leadership roles in the design of electronic computing devices, or "thinking machines." Yet for more than a decade after this piece was published, Zadeh offered little in the way of follow-up commentary regarding the position of electrical engineers relative to computers and computing. Indeed, he was likely occupied with his intense research and teaching activities at Columbia University, which led to his rapid ascension to full professor in 1957. His career trajectory took another important turn in 1959, when he was lured to UC Berkeley's esteemed Department of Electrical Engineering (McNeill and Freiberger, 1993, p. 22). After taking over as chair of the department in 1963, Zadeh's work started to move in a number of important directions. In terms of research, his interests in decision analysis and system theory led him to establish important foundations for the field now known as "fuzzy logic" or "fuzzy theory."[161] In fact, this particular area of research remains one of Zadeh's main claims to historical fame.

But as a department head, Zadeh also found himself surrounded by debates about both the future of electrical engineering education and the rise of "the computer sciences." Some of the first evidence for Zadeh's engagement with these issues can be traced to changes in his own academic unit. In 1964, for example, a new electrical engineering curriculum was adopted at Berkeley that offered undergraduate students four distinct program options: Computer Science; Electronics, Fields, and Plasmas; Systems, Information, and Control; and General Electrical Engineering (Zadeh, 1967, p. 9). By 1965 the name of the department was changed from Electrical Engineering to Electrical Engineering and Computer Science (EECS), and in a 1967

[161] A 1965 article by Zadeh titled "Fuzzy Sets" helped establish him as a "founder" in the emerging field of fuzzy logic (Zadeh, 1965a). For a biography of Zadeh that is largely focused on this aspect of his career, see McNeill and Freiberger (1993). Seising (2005), on the other hand, provides a nice overview of the historical development of Zadeh's work in area of fuzzy logic, emphasizing "that the genesis of fuzzy sets is not a story of basic research in set theory or symbolic logic or philosophy of mathematics but it is a story of fundamental research of a mathematical oriented electrical engineer and system theorist" (p. 5). Seising's characterization of Zadeh as a mathematicially-oriented engineer also helps explain Zadeh's strong feelings about the close relation of computing and engineering.

presentation Zadeh boasted that total student enrollment in the department's computer science courses had risen from 150 in 1963-64 to more than 1900 in 1966-67 (Karp, 2004, para. 3; Zadeh, 1967, p. 10). While these developments suggested that Zadeh and his colleagues were working to somehow meld computer science and electrical engineering at Berkeley, more details regarding Zadeh's agenda and outlook can be gleaned from his writings during this time period.

The first article of relevance was published in both the *IEEE International Convention Record* and the *IEEE Transactions on Education* (Zadeh, 1965b; 1965c). Titled "Electrical Engineering at the Crossroads," it called on electrical engineers to stake out large swaths of the computer field as their own. Zadeh started by explaining that the "health and vitality" of electrical engineering was threatened by a number of pressing challenges. More specifically, he indicated that "[b]y far the most serious of these problems centers on the relationship between computer sciences and electrical engineering" (1965c, p. 30). After emphasizing the key role played by electrical engineers in the history of computing, the author indicated that electrical engineering departments were responding slowly to rapidly expanding computer use, rising demand for computer scientists and engineers, and the rise of the "computer sciences" (pp. 30-31).[162] Given these trends, the article indicated that many campuses were facing strong pressure to establish computer science programs and departments.[163]

In order to respond to such pressures, Zadeh proposed that the field of electrical engineering should bring "the development of computer sciences within its fold" (p. 31). Shoring up this argument, he added that electrical engineering both engaged with many topics relevant to the computer sciences and also had valuable resources to offer, such as manpower and facilities.

---

[162] In historical terms, Zadeh discussed how electrical engineers were pivotal in the early development of machine computation, including at the University of Pennsylvania and MIT. He also noted that a number of forward-looking electrical engineering departments at the University of Michigan, Carnegie Tech, and the University of Pennsylvania had set up special interdisciplinary curricula in communication and computer sciences. Putting forward a rather machine-centric view of the field, Zadeh added: "Historically, the technology of machine computation has been and still is largely within the province of electrical engineering, since large scale machine computers are primarily electronic devices" (1965cm p. 30). In this same article, Zadeh also noted that electrical engineers far outnumbered mathematicians in both the ACM and IEEE Computer Group (p. 31). The accuracy of this particular claim is not clear.

[163] Zadeh pointed to the aforementioned 1964 CUPM report as evidence for these pressures, although he steadfastly resisted the report's assertion that the field of computer science should retain close academic ties to mathematics. As Zadeh explained, many of the subjects typically associated with computer science were frequently offered by electrical engineering rather than mathematics departments (1965c, pp. 30-31). More specifically, he asserted that five of the eight elective areas identified in the CUPM report actually fell within the province of electrical engineering.

Zadeh also expressed considerable anxiety regarding the possible failure of such an agenda, and he explained that ongoing efforts to establish independent computer science departments "would be disastrous for electrical engineering in the long run, and would not necessarily be in the best interests of computer science" (p. 31). The author therefore concluded that "electrical engineering departments can provide a home for computer sciences within their domain and thus assume a leading role in the vital and rapidly growing field of engineering and scientific activity" (p. 33). Per Zadeh, bringing computer science into electrical engineering was an advantageous proposition for both fields.

Yet the article also acknowledged that this process would likely change the identity and orientation of electrical engineering. In fact, Zadeh tentatively argued that it was time to abandon the view of electrical engineering as a "single unified field of engineering," and he supported this claim by describing how Berkeley's EE curriculum had been split into four distinct "programs" (pp. 31-33).[164] Zadeh went on to boldy suggest that the term "electrical engineering" might be replaced by a new name that emphasized the orientation of the field toward electronics and information processing (1965b, p. 50; 1965c, p. 33). Many of his suggestions also replaced the term "engineering" with "science." [165] In addition to arguing that an alternate name would more accurately reflect the contemporary image and span of the field, he claimed that it would help "retain the vitality which it still has but is in danger of losing through inaction and lack of foresight" (1965c, p. 33). As suggested by this overview, Zadeh's agenda potentially put him at odds with many actors and groups, including both the proponents of separate computer science departments and other factions of the electrical engineering field.

[164] In the same journal issue, engineer Robert M. Saunders of the University of California Irvine followed a similar line of reasoning when he indicated that "electrical engineering as a separate and distinct discipline may not exist in 1975" (1965, p. 33). Yet in contrast to Zadeh – who emphasized the links between electrical engineering, electronics, and the computer and information sciences – Saunders noted that future electrical engineering faculties might be clustered in areas such as engineering science, materials engineering, guidance and control, and applied physics (pp. 33-34). While perhaps overstated, these authors' remarks revealed that the ongoing transformation of electrical engineering was highly probable, especially in the midst of ongoing changes in the technological and disciplinary landscape.

[165] The names proposed by the author included: Electronic and Information Sciences; Electronic and Information Engineering; Electronic Engineering and Information Sciences; Electronic, Systems, and Communication Sciences; and Electronic, Control and Information Sciences (1965b, p. 50). These suggestions reveal that Zadeh was unsure of whether to frame the field as linked to engineering, science, or some combination of the two. It is also worth noting that the author favored terms such as "information" over popular alternatives such as "computers" or "computing."

In fact, Zadeh's reputation clearly preceded him when he addressed these and other issues at the aforementioned Stony Brook conference in 1967 (Zadeh, 1968a). In a talk titled "The Dilemma of Computer Sciences," Zadeh started by suggesting that one of the conference chairs had introduced him to the audience as a "progressive conservative" (p. 61). He also went on to acknowledge that "[p]robably most of you anticipate that I will take a militant stand in favor of developing computer sciences within electrical engineering departments, rather than within separate computer science departments" (p. 61). As suggested by these remarks, Zadeh was viewed by many as progressive with respect to reforming electrical engineering education, yet conservative because it was assumed that he opposed the establishment of independent departments of computer science.

To be sure, Zadeh was in a difficult position. Yet the politically savvy Zadeh responded to the conference chair's characterization by indicating "there is no universal answer to the question of what is the best organizational structure for instruction and research in computer sciences in an academic environment" (p. 61). Yet Zadeh was willing to speak about the disciplinary position of computer science in more abstract and generalized terms, and he placed particular emphasis on discussing how the emergent field was related to both mathematics and electrical engineering. With regard to the former, he followed prior commentators such as Gorn and Keenan when he noted that the close relationship between mathematics and computer science stemmed largely from "the intrinsically abstract nature of the mental attitudes of the computer scientist and his lack of preoccupation with the physical aspects of signals and systems" (p. 63).

Turning to the relation of computer science and electrical engineering, Zadeh reiterated an important historical claim: "[E]lectrical engineering, by virtue of its long standing and deep involvement in information processing technology, has vital concern not only with the use but, more important, with the conception, design and construction of digital computers" (p. 64). He went on to identify a number of specific subjects and topics that were related to both electrical engineering and computer technology, and he noted that jurisdictional conflicts over computer science were significantly linked to overlapping concerns in the area of "information processing" (p. 64). And while he repeatedly emphasized that there was no "single formula" or "universal answer" to such conflicts, he emphasized that electrical engineering departments had a "responsibility" for providing their students with training in "digital information processing and

the computer sciences," especially in light of growing industry demand for expert workers in these areas (p. 66). This was an important line of argument, as it helped frame the efforts of electrical engineers as motivated by historical precedent and professional responsibilities, rather than by ill-defined or misguided "political" motivations.

Toward the end of his talk, Zadeh also delivered a concise summary of his agenda for electrical engineering departments:

> [E]ither by themselves or in cooperation with computer science departments, electrical engineering departments should be offering broad programs in computer sciences and information systems, covering such areas as hardware, logical design and machine organization, programming languages, automata theory, formal languages and artificial intelligence (p. 66).

In essence, Zadeh was tentatively mapping out a disciplinary settlement for electrical engineering departments in the domain of computing. He also went on to emphasize that affiliated educational programs should be oriented toward the needs of "information systems designers rather than users," and he added that gaining competence in computer science should be a ready possibility for electrical engineering students. Even more generally, Zadeh claimed that it was essential for electrical engineering and computer science departments to "learn to live with one another" and work as "partners" in the training of computer scientists and engineers (p. 66). He concluded his talk by noting that training for tens of thousands of computer scientists and engineers was needed in subsequent years, which suggested that there was plenty of demand to support a variety of computer-oriented educational programs, no matter their institutional or disciplinary location. Hence, Zadeh's claims suggested that computer science and electrical engineering might successfully coexist in the context of the academy, even in the midst of their overlapping and interpenetrating settlements in the domains of computing theory and computer technology. And indeed, disciplinary theorists such as Abbott have convincingly argued that this type of outcome is not only possible, but also quite common.

In summary, some of the earliest efforts to stake out large swaths of computing as a province of engineering can be traced back to the early and mid-1950s. Zadeh's remarks provide evidence for a revitalization of this movement that was both prompted by the rise of computer science and principally focused on the academic sphere. Below, I discuss how this movement expanded from the mid 1960s onward. Before doing so, however, it is worth reviewing a 1968

paper by Zadeh that reveals some of the key challenges and tensions that came with trying to bring computer science into electrical engineering. More specifically, my analysis suggests that engineers such as Zadeh were working with an image of computer science that substantially differed from how the discipline was viewed by mathematicians and computer scientists.

Table 5.1 – Containment Table for Computer Science (Zadeh, 1968b, p. 913)

Subject	Degree of Containment in CS
Programming languages	1
Computer design and organization	1
Data Structures	1
Models of computation	1
Operating systems	1
Programming systems	1
Formal languages and grammars	0.9
Computational linguistics	0.8
Automata theory	0.8
Finite-state systems	0.8
Theory of algorithms	0.9
Discrete mathematics	0.8
Mathematical logic	0.6
Combinatorics and graph theory	0.8
Dynamic programming	0.7
Mathematical programming	0.7
Numerical methods	0.8
Switching theory	0.8
Analog and hybrid computers	0.7
Computer graphics	0.7
Digital devices and circuits	0.7
Artificial intelligence and heuristic programming	0.9
Information retrieval	0.7
Information theory and coding	0.6
Pattern recognition and learning systems	0.6

Engineering Images of Computer Science: Discipline, Department, and/or Program?

Zadeh authored a follow-up article in 1968 that revisited and expanded on many of the issues raised in his prior writings (Zadeh, 1968b). Suggestively titled "Computer Science as a Discipline" and published in the *Journal of Engineering Education*, this particular piece also provides evidence for how Zadeh's participation in the discipline-building project of computer science was inflected by his own background and interests. Noting ongoing disagreements over the definition of computer science, Zadeh explicitly agreed with the efforts of the ACM C^3S to frame the proposed discipline of computer science as largely concerned with "information" (p. 913). Yet he indicated that these types of generalizations failed to adequately delineate the field's boundaries. He therefore used his own concept of "fuzzy sets" to develop a "containment table" for the major subjects of computer science, as shown in Table 5.1.[166]

The author's unique approach leads us to a number of important insights. To begin with, his list of major subject areas did not depart significantly from the courses and topics identified and described in the ACM's preliminary curricular recommendations, suggesting that Zadeh's understanding of the general scope of computer science was not all that radical or controversial. Yet unlike many other commentators, Zadeh developed a more "bottom-up" characterization of the field that was based on identifying a large number of more specific and well-established constituent subject areas.[167] Unlike many of the aforementioned attempts to define computer science in a top-down manner, this "fuzzy" approach nicely resonates with a more settlement-based model of discipline formation and development, where a given field may have associations of varying strength with a wide-range of epistemological and technological domains.

Yet in addition to providing summary descriptions for each of the subject areas listed in this containment table, Zadeh identified and discussed a related series of controversial questions:

Is computer science a discipline?

Is it a branch of science or engineering?

What is its relation to mathematics?

---

[166] As Zadeh explained, "[L]et us regard computer science as a name for a fuzzy set of subjects and attempt to concretize its meaning by associating with various subjects their respective degrees of containment (ranging from 0 to 1) in the fuzzy set of computer science" (1968b, p. 913). He also noted that the numerical values listed in the containment table were only "rough measures of inclusion, with no claims to universality or long-term validity" (p. 914).

[167] This approach also conveniently allowed Zadeh to introduce and promote his theoretical apparatus of "fuzzy sets" to an audience comprised largely of engineers and engineering educators.

> What is its relation to electrical engineering?
>
> Should the instruction and research in computer science be centered in an independent academic unit or should it be conducted within an established academic department? (p. 915).

The author pragmatically waved off the first issue by claiming that the growth and popularity of the field – especially in the educational arena – were ultimately more important long-term measures of success for computer science. Like many before him, Zadeh clearly recognized that achieving disciplinary legitimacy and recognition is often a grassroots process involving the establishment of courses, departments, and degree programs. Zadeh also noted two possible futures for computer science. On the one hand, he explained that the "core" subjects of the field provided it with a distinct "flavor and identity," and he noted that computer science might evolve into "a big and influential field in its own right" (p. 915). This outcome was probably a source of anxiety for engineers such as Zadeh, yet he also noted that the heterogeneity and rapid growth of computer science might instead cause the field to splinter and fragment.

As in his previous writings, Zadeh also discussed the position of computer science in a larger disciplinary milieu, and he noted ongoing efforts to link the field to mathematics generally and mathematics departments specifically. Yet Zadeh placed considerable emphasis on how electrical engineers were reacting to the expansion of computer science. In fact, he used an undated and unpublished Bell Labs memorandum titled "Engineering and Computing – A Holy Alliance" to speak for the many engineers who were highly skeptical of the emergent field. It is worth reproducing the lengthy passage from this memo – which was originally authored by engineers E. E. David, Jr. and Franklin F. Kuo – that Zadeh cited in his paper:

> There is, in fact, very little classical science behind computation today. On one hand there are the circuits, memories and systems which we call *hardware* which we associate with electrical engineering. On the other, there is computer *software* based upon linguistics, logic, and mathematics. There is "science" behind computation only in the same sense that information and detection theory behind communication can be called "science." Regardless of terminology, there is a real question of an appropriate philosophy for computing efforts in universities and research institutions. We believe that this philosophy should be rooted in *engineering* (quoted in Zadeh, pp. 915-916, author's emphasis).

To begin with, this passage once more revealed the extent to which the dualistic discourse of "hardware" and "software" had become a pervasive and convenient shorthand for the computer field's major sociotechnical boundaries. Further, this memo suggested that engineers were willing to acknowledge the mathematical dimensions of computing, especially in the realm of software. However, they steadfastly resisted the idea that work in the field should be primarily framed as scientific. In fact, they likely viewed their own work as no less scientific than the activities of the so-called computer scientists, especially given that these engineers hailed from an organization well known for its cutting-edge research and development activities.

Zadeh provided little additional commentary on the preceding passage, and his own views on "computer science as a discipline" suggest that he was somewhat more sympathetic with a scientific view of the emergent field. However, in a subsequent passage Zadeh did add that computer science had a "split personality" due to its relation to both mathematics and engineering. And while he noted that a similar characterization had led authors such as Perlis to insist that computer science was an independent field, Zadeh reiterated his view that there was a place for computer science in electrical engineering. He also stressed the importance of cooperation between competing departments, especially as electrical engineering education shifted toward digital techniques and participated in the training of large numbers of "digital system designers" and "computer scientists" (p. 916).

In even more general terms, participating in the discipline-building project of computer science may look like a particularly bold move for Zadeh, as it implicitly challenged the control that mathematicians and other non-engineers wielded over the emergent field. But Zadeh seemed to have the right kind of background to lead such a charge, especially given that much of his own work was highly mathematical, theoretical, and linked to the "engineering sciences." In a more recent interview, for instance, Zadeh's ruminations on his time as a faculty member at Columbia and Berkeley revealed the extent to which he was a long-time proponent of a mathematically intense flavor of engineering. As Zadeh explained, "I felt that my mission was that of teaching whatever subject I was teaching in a precise and rigorous fashion. In other words: to make engineering as close to mathematics as possible" (Zadeh, 2001). In light of such remarks, it is not entirely surprising that Zadeh saw electrical engineering as a natural home for computer science.

It is also worth highlighting Zadeh's ties with MIT's electrical engineering department, which had a long reputation for its orientation toward mathematics and the engineering sciences.

In fact, Zadeh used this department as something of an exemplar in his 1967 talk at Stony Brook. Noting the long and influential history of MIT electrical engineers in the "theory and practice of information processing in all its forms" (1968a, p. 65), Zadeh queried, "Would it make sense to set up a separate Department of Computer Sciences outside of Electrical Engineering in a case like that?" (p. 65). Answering in the negative, Zadeh turned to a suggestive metaphor of the physical body:

> Clearly, this could be done only by amputating a major part of the Electrical Engineering Department and combining it with parts of other departments. But where then would the cut in the body of electrical engineering be made? What professors in circuit theory, information theory, control systems, optimization techniques, pattern recognition and related areas be moved out of electrical engineering, or left behind? (p. 65)

While certainly dramatic, framing the issue in this matter revealed the stakes that were in play in debates over the position of computer science. For actors such as Zadeh, electrical engineering, computers, and computing were thoroughly intertwined, and driving a wedge between these domains amounted to an unnecessary act of violence against a unitary disciplinary body. Yet beginning in the mid 1960s and through the 1970s, even MIT was grappling with questions about the appropriate disciplinary position of computer science, as well as its relation to electrical engineering (Wildes and Lindgren, 1985, pp. 359-361; Aspray, 2000, pp. 49-51).

Further, it may seem that Zadeh's efforts to define computer science as a discipline to some extent undermined his argument that the field should remain a province of electrical engineering. And indeed, the preceding analysis reveals that many of the actors who similarly discussed the development of "computer science as a discipline" concluded that independent departments were crucial for the field's growth and success. However, I contend that engineers such as Zadeh were working with an image of disciplinarity that was at least partially divergent from the perspective held by many self-identified computer scientists. More specifically, disciplines in the context of engineering education have historically provided a way of organizing research and education, yet these divisions are rarely allowed to threaten the image of engineering as a single professional domain. Hence, it was not difficult for engineers such as Zadeh to conceptualize computer science as one engineering sub-discipline among many, but only as long as the field maintained the subservient identity of an "engineering science."

Yet there were many other actors who preferred to view computer science as a truly independent scientific discipline, unfettered from the potentially competing or even contradictory interests and commitments of other disciplines or professions. In fact, the influence of this latter group was evident at Berkeley, where Zadeh and his colleagues in electrical engineering were unable to block a partial secession of computer science faculty. As Zadeh was forced to admit in a 1967 talk, a new Department of Computer Science was being established in Berkeley's College of Letters and Science (1968a, p. 62). Per Zadeh, such a schism might be appropriate at a school such as Berkeley, which could "afford to have separate centers of activity in different colleges with different orientations" (p. 62). The new department was officially established in 1968, yet within a few years the situation was deemed unsustainable. Following much heated discussion and debate, the Department was moved back to the College of Engineering in 1973, transformed into a partially autonomous Computer Science Division within the EECS Department ("EECS History," n.d.). As many commentators have recognized, the events at Berkeley were watched closely, especially by faculty and administrators at other institutions that were similarly grappling with questions about both the identity of computer science as a discipline and its preferred position in the structure of the academy.[168]

Zadeh also played an influential role behind the scenes as a parallel scenario played out at the University of Pennsylvania. As the 1960s progressed, prominent faculty members such as the aforementioned Gorn and Carr were pushing for the establishment of a Department of Computer and Information Science at the school. According to Aspray, the realization of such a department looked increasingly likely by 1966, but was stymied at the last minute by John Brainerd, who at the time was serving as head of the Moore School of Electrical Engineering. As Aspray describes it, "Brainerd was generally supportive of computer science, for example, having been in favor of the hirings of Gorn and Carr – but only to the extent that computer science did not harm electrical engineering" (2000, p. 64). Further, Aspray explains that it was Zadeh who was instrumental in convincing Brainerd "of the benefits to electrical engineering of keeping computer science within his domain" (p. 64). A separate department of CIS was finally established at Penn in 1970, although it remained within the confines of the Moore School (p. 64). The compromises worked out at Berkeley and Penn were therefore quite similar in the end,

[168] As COSINE member Edward J. McCluskey explained in recent correspondence, "there were all these wars going on at universities for control of computer, quote computer science. … [T]he one that I think had the highest visibility was Berkeley" (McCluskey, 2005).

even if reached by different pathways. And as these schisms played out at these and other institutions, many were looking to the leaders of electrical engineering for further insight and inspiration. As I discuss in the following sections, the activities of the COSINE Committee were designed to provide some of this support and guidance.

An Introduction to the COSINE Committee: Historical Origins and Trajectory

While I have placed considerable emphasis on Zadeh's role in this historical account, he was but one important player in a larger movement that gained significant momentum in the mid and late 1960s. In fact, Zadeh's concerns about both the future of electrical engineering and its relation to computer science were shared with many of his colleagues, including Mac Van Valkenburg. Zadeh and Van Valkenburg's relationship can be traced back to the 1940s at MIT, with both men earning masters degrees in electrical engineering in 1946 (McNeill and Freiberger, 1993, p. 21; Zadeh, 1998; VanValkenburg, 1972, p. 246). Van Valkenburg also worked in the Radiation Laboratory and Research Laboratory of Electronics at MIT, and following graduation in 1946 he assumed instructor and then faculty positions at the University of Utah (Moone, 2002). He took leave to pursue a Ph.D. at Stanford, which he completed in 1952, and in 1955 he joined the Electrical Engineering faculty at the University of Illinois, Urbana-Champaign (Moone, 2002). Through the 1960s Van Valkenburg was an increasingly well-known engineering educator and textbook writer. And like Zadeh, he was also very active in the field of system theory.[169]

[169] In fact, Van Valkenburg helped organize the first Allerton Conference on Circuits and Systems in 1963. This event quickly grew to become one of the foremost conferences in the field of system theory (Moone, 2002). In the context of electrical engineering, system theory is generally concerned with the modeling and design of complex electrical or electronic systems, often with a strong theoretical and mathematical bent. It is also worth noting that many actors framed system theory as an emergent field or discipline. As Zadeh explained in 1963, "It is not sufficient, however, to put the label of 'system theory' on an aggregation of parts of several well-established disciplines. To acquire a distinct identity, system theory must develop its own body of concepts, problems, and techniques" (Cruz, 1963, p. 154). Yet despite such discipline-building rhetoric, there is little evidence that commentators such as Zadeh were inclined to promote the establishment of independent departments or degree programs in system theory. Hence, I claim that these authors conceptualized system theory as another sub-discipline of engineering, in a manner that was similar to how they viewed the field of computer science.

In early 1965, Zadeh and Van Valkenburg invited a number of electrical engineering department heads to a meeting at Berkeley.[170] According to Martha Sloan – an electrical engineering educator who both served on the COSINE Committee and evaluated the group's impact as a part of her dissertation research – "[t]he meeting was intended to reassess the relationships between computer science and electrical engineering and to study the role of electrical engineering departments in training computer scientists and engineers" (Sloan, 1973, p. 22). The participants, who included numerous department heads and various representatives of industry and government agencies, concluded that electrical engineering departments should lead this type of training. The principle outcome of this meeting was the formal establishment of the "Computer Sciences in Electrical Engineering" or "COSINE" Committee. As suggested by its name, the committee's identity and early agenda closely followed Zadeh's views on bringing the computer sciences "within the fold" of electrical engineering.

Members of the newly formed committee met at the annual meeting of the American Society for Engineering Education (ASEE) in June of 1965, and they received initial financial support from the Commission on Engineering Education of the National Academy of Engineering (NAE) (Sloan, 1973, p. 22). Later in 1965, the COSINE Committee – operating under the auspices of the Committee on Engineering Education – submitted a proposal to the National Science Foundation (NSF) for two additional years of funding for their activities (Huggins, 1969, p. 61). As suggested by these developments, the committee was operating in close coordination with the NAE, which itself was spun off from the National Academy of Sciences in 1964 as a private, independent, non-profit advisory group. The initial proposal was approved, and NSF support for the COSINE Committee officially commenced in July of 1966 (Committee on Computer Sciences in Electrical Engineering, 1968, p. 2). In 1968 and 1971, the group submitted successful proposals for continuations of NSF support (Sloan, 1972, pp. 23-24).

The COSINE Committee was active from 1965 to 1972, or a span of about eight years. The group spearheaded an array of activities during this time period, including on-site visits to universities, the organization of ten workshops and summer conferences and five EE chairmen's meetings, and the publication of 11 major reports. The leaders of COSINE also published bulletins, letters, and articles related to their efforts, and by 1970 the group was increasingly

[170] As evidenced by one account that was published a few years after the establishment of the COSINE Committee, Zadeh and Van Valkenburg were quickly recognized as the so-called "fathers" of the group (Huggins, 1968, p. 60).

focused on assessing both its own impact and other relevant trends in electrical engineering. According to Sloan, COSINE was composed of 14 core members, although only half of these were involved through the entire life of the committee (1972, p. 23). A total of 11 committee members held university appointments, and the remaining 3 were primarily affiliated with industry. The credentials of the group were also impressive. As noted by Sloan in her 1974 report on the impact of the group, "The committee included six current or former department chairmen, three members of the National Academy of Engineering, several IEEE Fellows, and authors of several texts" (p. 180). In addition, a total of more than 30 individuals – including, at one time or another, all of the core members – served on the various "task forces" that developed most of the COSINE reports.[171]

The group also maintained a working relationship with the ACM's the Curriculum Committee on Computer Science (C^3S), especially in the late 1960s. Edward J. McCluskey, for example, was a member of both COSINE and the ACM's C^3S, and he delivered a presentation on "The ACM- C^3S Curriculum" at the second COSINE-sponsored meeting of department heads, held in 1967 at Princeton University. McCluskey started his talk with the wry observation that "[i]t has been suggested that not everyone here is necessarily familiar with the C^3S abbreviation; perhaps even the ACM abbreviation may be strange" (1967, p. 6). In addition to describing both the ACM and its efforts in the area of curriculum development, McCluskey explained that the C^3S was interested in the perspectives of engineers, and he indicated that the ACM recommendations could provide useful inspiration for electrical engineering departments as they developed of computer-oriented courses. C^3S member William Viavant also acted a liaison between the two groups. His assistance was recognized in the COSINE Committee's inaugural 1967 report, and he also served as the chair of the 1968 Park City conference on Computers in Undergraduate Education, which was jointly sponsored by COSINE and the C^3S (COSINE Committee, 1967; Viavant, 1968).[172]

[171] The membership of each COSINE task force varied significantly, revealing the relatively loose structure of the committee. As I note below, this feature of COSINE contributed to an overall lack of cohesiveness and consistency in the numerous reports and recommendations that the group issued.

[172] While originally conceived as a follow-up to the 1967 Stony Brook conference, the agenda of this meeting was reoriented in response to the NSF's interest in sponsoring an event that was generally focused on computers in undergraduate education, rather than on computer science more specifically. As a result, only one of the five conference working groups was dedicated to "Curriculum and Programs in Computer Science." (Viavant, 1968).

Yet by 1968, a COSINE proposal for continued funding explained that there was "little overlap in the interests of this Committee [the C^3S] and COSINE, except at the beginning level" (Committee on Computer Sciences, 1968, p. 7). And as I discuss below, from the late 1960s onward the agenda of the COSINE Committee was reframed in ways that distanced the group from the ACM's educational efforts. This same 1968 proposal also indicated that the IEEE Computer Group was not significantly involved in curriculum development or educational programs, which further bolstered the argument that the COSINE activities filled an important and unmet need. More recently, McCluskey proposed two additional reasons to explain the distance between COSINE and the IEEE (McCluskey, 2005). First, the Computer Group and its members were still recovering from a lengthy and laborious merger process that distracted them from other activities. And second, there were likely concerns about the autonomy that the group might lose – as well as the level of bureaucracy it might face – if closely affiliated with the IEEE.[173] As I discuss below, doubts were also raised in the mid-1970s regarding the ability of the Computer Group's Education Committee to carry forward some part of the COSINE Committee's agenda and activities.

In summary, the COSINE Committee addressed a wide variety of topics and issues during its existence, as suggested by the report titles listed in Table 5.2. In the following section, I place considerable emphasis on two of these reports as a window into the group's historical trajectory. The first of these was the group's inaugural report, which was initially published in 1967 and appropriately titled *Computer Science(s) in Electrical Engineering* (COSINE Committee, 1967b). *An Undergraduate Computing Engineering Option for Electrical Engineering*, on the other hand, was first published in 1970 (COSINE Committee, 1970). These reports are important for at least four major reasons. First, they presented reasonably comprehensive and detailed curricular recommendations, while other COSINE publications tended to focus on more specific topics, such as suggestions for the development of courses and labs. Second, these were among the few COSINE reports that received wider distribution in major professional publications. Third, those surveyed at a 1970 meeting of department heads ranked these two reports as the most significant of the seven major COSINE publications that had been published to date. Fourth, finally, and perhaps most importantly, these reports provide

[173] Sloan has similarly pointed to the COSINE Committee's probable desire for "independence" (Sloan, 2005).

evidence for key shifts in the committee's larger agenda and orientation, including its movement away from "computer science" and toward "computer engineering."

Table 5.2 – COSINE Committee Reports

Publication Date	Report Title
September 1967	*Computer Science(s) in Electrical Engineering* (COSINE Committee, 1967b; 1968a)
September 1968	*Some Specifications for a Computer-Oriented First Course in Electrical Engineering* (COSINE Committee, 1968b)
October 1968	*An Undergraduate Electrical Engineering Course on Computer Organization* (COSINE Committee, 1968c)
1968	*Proceedings of the Meeting on Computer Science in Electrical Engineering of the Commission on Engineering Education, October 24-25, 1968* (COSINE Committee, 1968d)
November 1968	*Some Specifications for an Undergraduate Course on Digital Subsystems* (COSINE Committee, 1968e)
September 1969	*Impact of Computers in Electrical Engineering Education – A View from Industry* (COSINE Committee, 1969a)
December 1969	*Computer-Oriented Electrical Engineering Experiments 1969-1970* (COSINE Committee, 1969b)
January 1970	*An Undergraduate Computer Engineering Option for Electrical Engineering* (COSINE Committee, 1970; Coates, et al., 1971)
March 1971	*Digital Systems Laboratory Courses and Laboratory Development* (COSINE Committee, 1971a)
June 1971	*An Undergraduate Course on Operating Systems Principles* (COSINE Committee, 1971b)
April 1972	*Minicomputers in the Digital Laboratory Program* (COSINE Committee, 1972)

COSINE, The Early Years: Promoting "Computer Sciences in Electrical Engineering"

The COSINE Committee's first report, *Computer Sciences in Electrical Engineering*, was framed as a preliminary set of recommendations that grew out of the group's early meetings and workshops. In an introductory passage, the authors of this report followed prior commentators such as Zadeh by pointing to the "long standing and deep involvement" of electrical engineers in all phases of information processing technology, ranging from use and application to conception, design, and construction (COSINE Committee, 1967b, p. 5).[174] Yet the authors admitted that at least three major developments were impacting this relationship (pp. 5-6). First, they noted a shift in emphasis from "hardware" to "software" in the sphere of computer technology.[175] Second, they explained that this shift was in part stimulating the emergence of the field known as "computer sciences." And third, the report indicated that information processing systems were increasingly based on digital (or "discrete") rather than analog (or "continuous") electronics technology.

The authors argued that these developments were "creating an urgent need for a major reorganization of electrical engineering curricula" (p. 6), leading them to encourage greater flexibility in the standard electrical engineering curriculum. They also presented three more specific sets of recommendations. The first and most extensive of these was focused on the development of computer science programs within electrical engineering, while a second discussed how electrical engineering education could be reorganized to place greater emphasis on digital systems and related topics (pp. 9-19; pp. 21-23). The third major section of the report explored how computers might be incorporated into a wide range of existing electrical engineering courses, especially for analysis, design, and related tasks (pp. 25-31). This topic, in particular, had much in common with earlier efforts to incorporate computers into the engineering curriculum, as exemplified by the previously mentioned Ford Foundation project.

Yet as documented in the previous chapter, the other areas of reform that the committee addressed had received only scattered prior attention. And evidenced by both its title and content, much of the 1967 report was dedicated to developing recommendations for computer science

[174] In fact, one of the introductory passages in this report was identical to Zadeh's talk at the 1967 Stony Brook conference. Zadeh or one of his colleagues likely copied this passage into the COSINE report. This clearly reflected the influence of Zadeh and his agenda on the group's early activities.
[175] More specifically, the authors equated "hardware" with "circuit and component design" and "software" with "system organization and programming" (COSINE Committee, 1967b, p. 5).

programs within electrical engineering. The committee approached this matter in a rather cautious and strategic manner:

> Clearly, it would be unreasonable to equate computer sciences with electrical engineering, or to regard it as a subset of the latter. Nevertheless, the close relation between the two is presenting the electrical engineering departments with a special responsibility for the training of the large number of computer scientists who would be needed … in the years ahead (p. 5).

And later in the report, the authors indicated that those electrical engineering students who pursued a computer science major should "acquire substantive competence in computer sciences and related fields, comparable, but not necessarily similar in content, to that acquired by students in a typical computer science department" (p. 9). In light of such remarks, one might question the extent to which the committee was promoting the training of computer scientists, or perhaps instead electrical engineers with some baseline level of expertise in computer science. The latter appears quite likely, especially given my prior arguments about engineers viewing computer science as one sub-discipline among many "engineering sciences."

Nonetheless, these passages suggest that the committee remained somewhat at cross-purposes with their adoption of the term "computer science." They also seemed to maintain an awareness of the political baggage that came with their recommendations. Their report provided descriptions for four core and twelve recommended elective subjects for a computer science program situated within an undergraduate electrical engineering curriculum. And in many ways, the subjects outlined in the report overlapped significantly with the ACM's preliminary recommendations for the computer science curriculum, especially in areas such as programming, machine languages, algorithms, and discrete mathematics. However, the dominant structure of the electrical engineering curriculum placed significant limits on how much coursework could be dedicated to computer science programs, in spite of the authors' claim that such programs "may or may not include a core of required electrical engineering courses in areas outside of computer sciences" (p. 11).[176]

[176] Engineering education has long had a reputation for being conservative and slow changing, and electrical engineering is no exception. In fact, making changes to the "core curriculum" has been – and in many cases remains – a sure fire way to trigger passionate debates among engineering faculty members. I revisit this issue below.

In fact, the core subjects described in the report represented a relatively small amount of coursework, making it clear that the recommendations were primarily designed for the development of computer science majors or options *within* the confines of existing electrical engineering programs. This was a politically expedient move, as recommending more radical changes to the engineering curriculum would likely jeopardize the ability of schools to maintain accreditation under the guidelines developed by the Engineers Council for Professional Development (ECPD). To put it another way, moving too far away from the dominant model of engineering education could endanger a school's ability to produce "certified" graduates who could go on to become recognized as professional engineers.

As further inspiration for how such programs might be realized within existing departments, the report included three sample curricula in an attached appendix (COSINE Committee, 1967b, Appendix B). The examples included Computer Science bachelor degree programs situated within Colleges of Engineering at Berkeley and the University of Utah, as well as a proposed "Computer Science Program" within the existing structure of MIT's S.B. in Electrical Engineering.[177] Perhaps not surprisingly, all three of these programs retained a strong engineering orientation, albeit with a number of computer-oriented classes inserted in the curriculum. In many ways, this approach to revising the curriculum looked like a direct response to a question that was discussed at a 1967 COSINE meeting, namely: "What might constitute the *minimal* CS needs for all EE students; for a CS major within EE" (p. 34, my emphasis). This minimalist approach to computer science education stood in marked contrast with the ACM's ambitious curricular recommendations, which were designed to be more structured, cohesive, and comprehensive. In fact, the members of the ACM C^3S had a distinct edge over COSINE in this regard, because they could propose computer science degree programs that sidestepped the pre-existing educational requirements and restrictions that were characteristic of other fields.

The authors of the 1967 COSINE report also detailed how their curricular recommendations were related to computer technology. And unlike their ACM counterparts, they placed more explicit emphasis on the sphere of "hardware."[178] For example, the report

[177] According to the proceedings of a 1967 COSINE meeting, Syracuse University was also developing a separate B.S. degree in computer science that was to be administered by the school's electrical engineering department (COSINE Committee, 1967a, p. 33).

[178] As noted above, the curricula developed by the ACM C^3S placed considerable emphasis on programming, numerical analysis, algorithms, and related subject areas. However, this group clearly

indicated that student experiences in two of the four core subject areas should stress "computer hardware as the means of realizing programming functions" (p. 11). And elsewhere, the report indicated that the core subject of Computation Structures required "[c]onsiderable emphasis … on the interrelation and trade-offs between hardware and software techniques" (pp. 12-13). And for at least two major reasons, the authors added that the early stages of the curriculum should emphasize programming features before turning to more machine-oriented topics. First, they indicated that this approach could shed light on how "programming features" informed various aspects of machine organization. And second, the authors noted that this sequence of instruction could help students view conventional approaches to machine organization "in a less sacred light" (p. 11), thereby allowing them to consider alternatives ways to implement programming features in hardware. This looked like an important step toward recognizing – and perhaps even working to reconcile – some of the schisms that had grown up around the software and hardware phases of the field. However, this educational model continued historical precedent by framing engineers as the ultimate arbiters of computer design decisions.

With regard to realizing their proposed curriculum, the authors refused to take a position on jurisdictional issues, including questions about departmental responsibility for particular courses. However, they did stress that electrical engineering departments should cultivate faculty expertise in the computer sciences and related areas, and they encouraged close cooperation with other relevant departments. And at a 1967 COSINE meeting, a discussion group addressed closely related questions about why it was appropriate for electrical engineering departments to offer computer science courses and/or programs, including at the graduate level. As explained in one summary report, the focus of engineering education on "systems design" provided this justification (COSINE Committee, 1967a, p. 46). This report also emphasized that it was the

---

recognized that computer science students should have some familiarity with the hardware aspects of computers. Their 1965 report, for example, identified "Computer Organization and Programming" as a required course and "Logic Design and Switching Theory" as a highly recommended elective (ACM C^3S, 1965). "Curriculum 68" similarly recommended that students take a "Computer Organization" course (ACM C^3S, 1968). In addition, the 1965 report acknowledged that "[i]t has been suggested that the educational needs of those who will plan and design the computing and communication equipment should be given special consideration" (ACM C^3S, 1965, p. 545). Responding to this need, *Curriculum 68* identified a series of optional and elective courses that would allow students to specialize in the area of "Computer Organization and Design." However, there remain many open questions about how closely these recommendations were followed by schools, and commentators such as Ceruzzi have noted that the core of the 1968 curriculum almost entirely eschewed "hardware" courses and subjects, replacing them instead with an emphasis on algorithms, programming languages, and data structures (1988, p. 268).

responsibility of electrical engineering education to provide an "integrated engineering viewpoint" for those charged with designing digital systems (p. 46). Yet as I discuss below, it was increasingly questionable whether the committee's focus on the engineering and design aspects of digital systems was compatible with its use of the "computer science" moniker.

Transitional COSINE: From Computer Science to Computer Engineering

In March of 1968, the first major set of COSINE recommendations reached a wider audience through the publication of a condensed version of the group's 1967 report in the *IEEE Spectrum* magazine (COSINE Committee, 1968a). And in June of the same year, the COSINE Committee submitted a request for continued NSF funding via a proposal titled "A Program to Stimulate the Development of Electrical Engineering Courses and Curricula To Include the Computer Sciences" (Committee on Computer Sciences, 1968). As in the 1967 report, this document emphasized the goal of bringing the computer sciences into electrical engineering. At least on the surface, the titles and contents of these documents suggested that the committee was both united by a common purpose and headed in a consistent direction.

Yet a closer analysis of other early COSINE documents reveals important variations in the group's agenda. At the second COSINE sponsored meeting of department heads in 1967, for example, Van Valkenburg explained that the main objective of the COSINE Committee was to "assist Electrical Engineering Departments in reorienting their curricula to provide for a greater emphasis on digital technology and the associated symbol manipulation techniques" (1967, p. 3). A discussion group at the same conference, on the other hand, addressed the topic of "computer design in the undergraduate education," and participants identified a handful of courses that might make up a "computer design option" for electrical engineering students (COSINE Committee, 1967a, pp. 36-37).[179] Even the group's aforementioned 1968 proposal hinted at the group's wide-ranging objectives, which also included encouraging the use of computers for design and analysis, as well as more generally reorienting electrical engineering courses and curricula toward digital techniques. While these objectives were not necessarily at odds with developing computer science in electrical engineering, realizing these diverse goals clearly demanded a range of different strategies and approaches. It was therefore possible that the

[179] The five courses included: Introductory Computer Concepts and Programming; Switching Theory and Logic Design; Computer Organization and Digital Systems Design; Laboratory, Digital Devices and Circuits; and Advanced Programming (COSINE Committee, 1967a, p. 37).

Committee's agenda was too wide and its resources were stretched too thin, especially given the many potential barriers and challenges they were facing.

By late 1968, the orientation of the committee was beginning to shift more markedly. Evidence for this trend can be found in the published proceedings of a third COSINE-sponsored meeting of electrical engineering department heads, held at Stanford in October of 1968. This event included a paper by COSINE Committee member Clarence L. Coates, who joined the group in 1967. Like many actors in this history, Coates' career trajectory straddled the boundaries of engineering, science, and computing. After receiving a Ph.D. in Electrical Engineering from the University of Illinois in 1953, Coates worked as an assistant professor at the same school, and then as a research scientist at General Electric ("1993 OECE Recipiants," 1993). In 1963 he joined the faculty at the University of Texas at Austin, where he variously served as Professor of Computer Sciences, supervisor of the graduate Information Sciences program, and head of the Department of Electrical Engineering. He returned to the Electrical Engineering department at the University of Illinois at Urbana in 1971, and in 1973 he took over as the head of Purdue's School of Electrical Engineering.[180]

As suggested by the title of his 1968 talk, Coates issued a passionate plea for the development of "University Education in Computer Engineering" (Coates, 1968). More specifically, he started by noting rapid growth in both graduate and undergraduate computer science programs, and he explicitly emphasized that these were often "science oriented" (p. 5). Coates also explained that "education in computer engineering is being neglected at most institutions," and he stressed that this type of education demanded an "engineering educational environment," which presumably only colleges and departments of engineering could provide (p. 5). He also presented data – which he drew from the aforementioned Stony Brook proceedings – to highlight both the rapid growth of computer science education and the relative lack of computer-oriented degree programs in electrical engineering.[181] As noted above, such statistics generated significant anxiety for many electrical engineering educators, Coates included.

[180] According to this same source, Coates later spearheaded the development of a computer engineering degree program at Purdue ("1993 OECE Recipients," 1993).

[181] More specifically, Coates noted that the total number of computer science degree programs was forecasted to increase from 58 to 240 from 1964 to 1968, while the number of computer options in electrical engineering was expected to rise from 19 to just 23 during this same period (1968, p. 5). While Coates' figures for computer science are based on a different interpretation of the data than what I presented above, they highlight the same overall trends.

The author also admitted these trends were not necessarily a cause for alarm, especially if computer science programs provided an adequate an appropriate type of education. However, he complained that most computer science programs were largely focused on software and theory, with particular emphasis on topics such as programming, numerical analysis, formal languages, automata theory, and applications. He therefore argued that computer science education was deficient in "the hardware aspects of computers, in the hardware-software interface area, and in systems for which the computer is a component part" (p. 5). The author explained that a major reason for such deficiencies centered on the links between computer science and the arts and sciences, especially in terms of the background and interests of computer science faculty, as well as the dominant institutional location of computer science departments and programs. He also complained that the courses recommended by the ACM lacked an engineering orientation. According to Coates, those responsible for developing *Curriculum 68* simply had "no interest, experience, or appreciation for engineering" (p. 7), and he concluded that the ACM's recommendations were ultimately "a computer science curriculum and not a curriculum for computer engineering" (p. 7). Such remarks suggest that Coates was appealing to engineers who maintained deep-seated feelings about how their work was distinguished from that of scientists.

Coates went on to argue against the idea that computer engineering was somehow "a part of" computer science, and he instead framed computer science and computer engineering as distinct domains that needed separate educational programs. "Where we have failed," Coates opined, "is to recognize that computer science education and computer engineering education are *not* the same and that there is a need for both" (p. 10). And while the author failed to provide a direct definition for computer engineering, he hinted at the meaning of the term when he noted rising demand for "engineers who are trained in the analysis, organization, and design of systems that perform one or more of the functions of control, communications, recognition, processing and retrieval" (p. 7). As Coates explained, the most practical way to provide this type of training involved the establishment of computer engineering options within electrical engineering departments, and he indicated that such programs would place extensive emphasis on subject areas such as control systems, information and communication theory, logic design and switching theory, machine organization, and programming (p. 10). In addition to providing an updated definition for the field, Coates' remarks tentatively and partially outlined a disciplinary settlement for "computer engineering."

Responding to a potential point of criticism, Coates also argued that it was necessary for computer engineering education to move away from some of the "fundamental" subjects that had long been at the core of the standard electrical engineering curriculum. In fact, Coates indicated that a computer engineering program would likely eschew any engagement with the subject area of "power systems," thereby completing a historical trend that had started many decades earlier with the rise of electronics. He also noted that such a program would place relatively less emphasis on electromagnetic theory, network theory, electron materials and devices, and electronic circuits (p. 10). This was a major call for change, especially given that most 1960s-era electrical engineering programs – oriented as they were toward both the engineering sciences and electronics – tended to cover these subject areas rather extensively.

Comparing his proposal to the earlier shift in the field from power to electronics, Coates also argued that electrical engineering education was entering a new period of transition. More specifically, he described the growth of computer engineering as representing the emergence of "a new epoch" in electrical engineering, and he even went so far as to state: "I am not now suggesting that the electronics epoch is ending, although this may be true" (pp. 7, 10). He concluded his talk by boldly declaring, "I would chide you as the leaders of electrical engineering education, as well as we of COSINE, for failing to recognize long ago the need for education in computer engineering" (p. 10). Of course, the previous chapter revealed that scattered commentators were calling for the establishment of such programs by the late 1950s and early 1960s, and commentators such as Vincent Rideout and Norman Scott even used the phrase "computer engineering" to describe graduate electrical engineering programs that were oriented toward computer system design and associated subjects. Yet these types of programs had failed to gain significant momentum outside of a handful of institutions, such as Scott's own University of Michigan. In the meantime, computer science had emerged and grown prodigiously, often beyond the purview of electrical engineering.

Coates' concerns received additional attention at the same 1968 COSINE meeting through a workshop that was aptly titled "Computer Engineering Rather than Computer Science" (1968d, pp. 16-18). As indicated in one post-workshop summary report, "[t]here seems to be a well defined separation of interest developing between the curricula of Computer Science Departments and Computer Science programs offered within Electrical Engineering Departments" (p. 16). The report then identified a series of topic areas in mathematics, electrical

engineering, and computer theory that were of particular interest "to an engineer working in the area of computers and information systems" (p. 17). In fact, it was noted that four of the topics in the electrical engineering category fell outside of the ACM's curricular recommendations.[182] This piece of evidence provided additional support for the claim that computer-oriented programs in electrical engineering were at least partially divergent from computer science. The report even asserted that "[a] consensus was reached that the existence of a Computer Science Department must not interfere with the development of a strong computer oriented program in Electrical Engineering" (p. 17).

Three additional COSINE reports were published in 1968 and 1969, and each lent a measure of support to the agenda outlined by Coates. In fact, Coates was the only member of the COSINE Committee who participated in the development of all three of these documents. The first such report provided recommendations for an undergraduate course dedicated to the subject of "Computer Organization" (COSINE Committee, 1968c). Contextualizing their efforts, the authors of this report acknowledged the ongoing establishment of Computer Science departments, yet they argued that "there is an ever increasing need for electrical engineers whose undergraduate program provided a familiarity with digital system design" (p. 2). They explained that such programs required engagement with both the "hardware and software aspects of digital systems," such as via the course outlined in the report. In fact, they made it clear that "such a course should be offered by the Electrical Engineering Department and should correlate the design and organizational aspects of the subject" (p. 2). In light of the different philosophies of course and program design to which these passages elude, electrical engineering and computer science departments at many schools were developing and offering their own, separate versions of courses in overlapping areas of interest, including computer organization.[183]

Another COSINE Task Force presented specifications in 1968 for a one-year undergraduate elective course in the area of "Digital Subsystems" (COSINE Committee, 1968e). The content for this course took a bottom-up and hardware-oriented approach, beginning with

[182] These four topic areas included: Circuits and Systems; Electronics; Control, Communication, and Information Theory; and Solid State Electronics (p. 17).

[183] In a more recent conversation, McCluskey discussed how control over individual courses became an important site of negotiation and competition for rival departments: "[O]ne of the ways in which this battle [for control of computer science] was fought out was by control of which courses the students in the department could take. And I'm sure there are many instances at universities where there were two computer activities, in two different departments, where one of the departments wouldn't recognize the courses in the other department" (McCluskey, 2005).

basic circuits and simple functional units, and proceeding to the design of complete "digital subsystems." And while this course partially overlapped with computer organization, the two courses clearly complemented one another and covered many of the subject areas that were historically associated with the domain of "computer engineering."

A 1969 COSINE report, on the other hand, was more generally concerned with the *Impact of Computers on Electrical Engineering Education – A View From Industry* (COSINE Committee, 1969a).[184] As the authors of the report indicated, one of the main goals of the COSINE Committee was to "keep abreast of trends and developments in the area of computer engineering and computer science and bring this information to the attention of electrical engineering educators" (p. 1). In addition to summarizing how computers and digital systems technology were impacting the actual practice of engineering in industry, the report discussed how electrical engineering educators might respond to these trends. The task force lobbied for more flexible curricula, and they emphasized the importance of student experiences in design-oriented projects. And in terms of topical coverage, they complained that "electrical engineering departments are not updating their curricula in this area as fast as the present and future practice of engineering would warrant," especially given the extent to which computers and "digital systems concepts" had permeated engineering practice (p. 1). The report stressed that students should have opportunities to use computers for engineering problem solving, while also gaining experience with the design and simulation of digital circuits and systems.

It is also worth noting that the authors of this report did not specifically discuss computer science or computer engineering programs, preferring instead to frame their discussion as more generally relevant to the education of electrical engineering students. However, the COSINE Committee's next major report made it clear that the group's agenda and activities were increasingly being advanced under the banner of "computer engineering." As a result, their work pointed to the potential development of new alignments and synergies between the field's professional jurisdictions and its disciplinary settlements.

[184] This report also indicated that one of the earliest formal activities of the committee was a late 1966 meeting with representatives of industry that was intended to "determine the impact that computer technology was having upon industry" (p. 1). The close ties between COSINE and industry stood in marked contrast with the ACM's curricular efforts in computer science. In fact, evidence suggests that the emergent field of "computer science" had very low visibility in industry, at least through much of the 1960s. As Eric Weiss of Sun Oil Company explained at the 1967 Stony Brook conference, "I made inquiry of my colleagues in industry to get their views of computing science and its relevance to their world. … Too often their reply was a question, 'What is computing science?'" (1968, p. 105).

COSINE and Computer Engineering: Expanding EE From the Inside Out

The COSINE Committee's *An Undergraduate Computer Engineering Option for Electrical Engineering* (COSINE Committee, 1970) was developed by a seven-member task force, which included only three of the authors listed in the group's 1967 report. In fact, notable individuals such as Van Valkenburg and Zadeh were not directly involved with this task force, while Coates acted as chair. The document therefore represented another unique set of interests, agendas, and stakeholders, and in most general terms it cemented the COSINE Committee's movement away from "computer science" and toward "computer engineering." And perhaps more than any other document, this report can be credited with stimulating the widespread development of computer engineering education. Yet by framing computer engineering as a branch or sub-discipline of electrical engineering, this set of recommendations also hinted at potentially disruptive shifts in the identity and disciplinary settlement of the parent field.

The authors started the report by indicating that their efforts were prompted by a "growing demand for education in computer engineering and the limited opportunities for study in this area" (p. 1). As additional background, they briefly outlined the history of electrical engineering from the 1930s onward, with particular emphasis on the shifting orientation of the field from power to electronics. The authors also followed prior commentators by noting that digital technologies and systems were increasingly central topics in the field. Given this trend, they explicitly described "computer engineering" as that part of electrical engineering concerned with "the organization, design, and utilization of digital processing systems as general purpose computers or as components of systems concerned with communication, control, measurement, or signal processing" (p. 1). This definition was generally consistent with how commentators such as Coates had used the term in the past, especially with regard to the field's focus on the design and organization of computer systems and the components thereof.

Yet this passage also hinted at another emergent area of technical expertise, where engineers were being called upon to incorporate computers into even larger technological systems. In fact, demand for this type of expertise was likely on the rise, especially given both the increasing availability and falling costs of computers around this time. Hence, even if there remained relatively few employment slots for those who were directly involved in the design of computer systems and components, expanding the field's settlement to include the design of

these larger types of computer-based systems looked like an appropriate move for those who advocated the ongoing expansion of computer engineering education.

Adopting the computer engineering moniker was also a sound strategy for these authors, especially given both its deeper historical roots and the difficulties that came with co-opting atlernate terms such as “computer science.” This move also allowed the authors to frame computer engineering as primarily or even wholly a branch of electrical engineering. In fact, the 1970 report even avoided the use of terms such as “field” or “discipline,” and it instead described computer engineering as a “new dimension” of electrical engineering. The committee’s activities were therefore reframed as a more natural expansion or extension of their own field, *from the inside out*. This approach stood in marked contrast with prior efforts to bring the *outside* discipline of computer science *into* electrical engineering, thereby leading to potential conflicts between the dominant image of computer science as an independent discipline and electrical engineering as a part of the engineering profession.

Yet the authors acknowledged that the subject of their report at least partially overlapped with prior efforts to develop computer-oriented educational programs, and they responded directly to questions about the necessity of their efforts. More specifically, the authors cleverly argued that previous studies were inadequate because they were not directly concerned with *computer engineering*. They also explained that the ACM’s *Curriculum 68* was a “science-oriented software program and not an engineering programming for education in digital processing system design” (p. 2), and they added that existing computer science departments and programs were turning out “software specialists.” The authors noted that the 1967 COSINE report, on the other hand, was designed to “indicate a minimal set of courses that could be included in the undergraduate electrical engineering curriculum … [to] introduce the student to the basic techniques and theoretical concepts of computing” (p. 2).

This same report went on to argue that computer engineers required a different type of education that covered “the design of software, hardware and systems” (p. 2). The authors also stressed that programs in this area should provide students with an understanding of “the important relationships and ‘trade-off’s’ between the hardware and software components of the system and an understanding of how these functions should be partitioned in the system organization in view of the intended applications” (pp. 2-3). With these objectives in mind, the task force identified a total of seventeen subject areas for a computer engineering option, and

these were further split into background, basic, and elective categories.[185] And in contrast to the 1967 COSINE report, this new set of recommendations provided more detailed information about the subject matter, semester hours, and overall structure of such a program. The report also included "possible" computer engineering curricula that were custom-tailored for implementation at four different universities, namely Carnegie-Mellon, Hawaii, Princeton, and University of Texas-Austin (pp. 7-10).[186] Unlike the curricular samples presented in 1967, these degree outlines were clearly identified as computer engineering programs *within* electrical engineering. This was an important shift, as it suggested that computer engineering students would receive degrees in electrical engineering rather than computer science. Electrical engineering educators could therefore move into the domain of computing in ways preserved their identity as electrical engineers who were training future engineering professionals. A slightly revised version of the same report reached a larger audience in 1971 through its publication in the *Proceedings of the IEEE* (Coates, et al., 1971). And in a new forward, COSINE chairman Van Valkenburg stressed the importance of the group's recommendations, especially given the ongoing and rapid growth of computer engineering education within electrical engineering departments (Coates, et al., 1971, p. 854).

The COSINE Committee also released three new reports in 1971 and 1972, and two of these were focused on the development of laboratory work that was compatible with the group's larger set of curricular recommendations (COSINE Committee, 1971a; 1971b; 1972). In one of these reports, the authors explained: "As digital system and computer engineering concepts have been integrated into the undergraduate electrical engineering curriculum, many departments have begun revising their laboratory programs to include more work with digital networks and mini-computers" (1971a, p. 1). Such passages once again reveal the extent to which larger currents of

[185] The six "background" subjects included physics, calculus and differential equations, linear and abstract algebra, probability theory, electric and electronic circuits, and introductory computer programming. With the exception of probability theory, most electrical engineering programs required that students take a similar set of core coursework. The four basic subjects identified in the report were switching theory and logical design, machine structure and machine language programming, computer organization, and systems programming and operating systems. And finally, the seven elective areas recommended by the committee were programming languages and translation, numerical analysis, logic and automata theory, communication systems, operations research, simulation and modeling, and field analysis. As this overview reveals, many of the topics that the ACM C^3S identified and described as core requirements for computer science were electives in the COSINE recommendations for computer engineering.

[186] All four of the proposed curricula were developed for schools where members of the task force served as faculty. These sample curricula also featured an entirely different group of schools as compared to the 1967 COSINE report.

change were finally beginning to move through electrical engineering education. The third of these reports, on the other hand, proposed an undergraduate course on "Operating Systems Principles." As explained by the authors, such a course would likely be realized as an elective "for students whose major interest is in the engineering of computer systems and software" (1972, p. 1). This particular report hinted at the extent to which the domain of software was becoming a more pivotal site for negotiating the boundaries between computer engineering and computer science, a point to which I will return.

As the preceding review makes clear, from the late 1960s and into the 1970s the COSINE Committee eagerly promoted its agenda under the banner of computer engineering. In fact, even Zadeh came to temper his use of the term "computer science." In a 1971 article on "Impact of Computers on the Orientation of Electrical Engineering Curricula" that was published in the *IEEE Transactions on Education*, Zadeh emphasized that "electrical engineering has a special responsibility to train its students in both the basic and applied aspects of computer science and engineering" (Zadeh, 1971, p. 153). Yet he admitted that the efforts of the COSINE Committee represented "a rather belated response on the part of electrical engineering educators to the challenge of the computer revolution" (p. 154). He went on to once more lobby for greater flexibility and multi-option systems in the electrical engineering curriculum. He also argued that required upper division core courses should be replaced by "a system of recommended programs" (p. 154). As Zadeh explained, one of principal advantage of such a "free curriculum" was that it "comes to grips with a basic fact of life, namely, that electrical engineering is no longer a unified field of study with a clearly definable single core; rather, it is an aggregation of subject areas" (p. 154). Per Zadeh, these included clusters of subjects in areas such as: systems, information, and control; computers and digital systems; circuits and electron devices; electromagnetics; bioelectrics; and urban and public systems (1971, p. 154).[187]

[187] It is again worth noting that these types of reform discussions could trigger passionate debates about whether various core courses or subjects should remain in the electrical engineering curriculum. In fact, McCluskey more recently recounted one such debate: "I have to tell you, the biggest fight that I remember, and you said tensions, and I'm talking about, there were not only tensions, but this was a fight, was a COSINE meeting, and it was out here at Stanford. And there was one guy on the committee. What the committee was discussing was whether there should be an E&M [electromagnetism] course in the EE undergraduate curriculum, or in the computer engineering undergraduate curriculum. … [A]nd this one guy was [whistles], he was very emotional about this. And he didn't think there ought to be one there. He thought we were compromising, selling our souls. I can remember him pounding on the table and walking out. But that's the only one I remember like that. And it turns out he was wrong" (McCluskey, 2005).

Comments such as these suggest that the concept of disciplinary settlement provides an appropriate lens for understanding the historical development of electrical engineering through this time period. More specifically, Zadeh's remarks revealed that electrical engineering – like computer science – was neither easily nor succinctly definable, as its domain ultimately comprised a range of loosely connected subjects, and many of these were shared with other fields. Hence, both Zadeh's article and the COSINE Committee reports from the early 1970s pointed to the persistent challenges that electrical engineering educators faced as they grappled with how to reform and revise their curricula in ways that accommodated the field's increasing diversity and scope while simultaneously preserving its cohesion and unique identity. These challenges were only compounded given that many of the proposed reforms could potentially threaten the dominant image of electrical engineering and its various sub-disciplines – including computer engineering – as unambiguous parts of the engineering profession.

Evaluating the "Impact" of COSINE and the Growth of Computer Engineering Education

From 1970 to 1972, the COSINE Committee conducted a series of surveys that were designed to both evaluate the impact of the group and document other relevant trends in electrical engineering education (Sloan, 1973, p. 27). Members of the group also used the survey data to support their agenda. In 1971, for example, Van Valkenburg noted that 87 of 203 (or 43% of) electrical engineering departments that responded to one survey offered an undergraduate option or program in computer engineering, while another 35 of the surveyed schools planned to offer such an option in the coming year (Coates, et al., 1971, p. 854).[188] As Van Valkenburg explained, these data revealed the importance of the COSINE recommendations with regard to the computer engineering curriculum. Yet it was the final COSINE survey – which was completed in 1972, just before COSINE disbanded – that provided the most detailed data regarding relevant educational trends in electrical engineering through the life of the Committee.

[188] Van Valkenburg repeatedly deployed similar statistics to discuss the movement of computers and computing into electrical engineering education. In another 1971 commentary, for example, he explained: "A recent survey of department chairmen I conducted had responses from 201 universities. Of this number, 86 indicated that they now offer an option or program in *computer engineering* at the undergraduate level, and 34 more showed some indication that there might be such a program or option within a year" (Dertouzos, et al., 1971). And while it is not clear why Van Valkenburg presented slightly different statistics in this piece, the larger trends were clear.

These results were summarized in a 1973 article that was authored by Sloan, Coates, and McCluskey (1973) and published in the IEEE's widely-read *Computer* magazine.

To begin with, it is worth noting how the history of the committee was framed in this document. In an introductory passage, the authors summarized that the COSINE Committee "was organized in September, 1965 to help electrical engineering departments develop educational programs in computer engineering and to design other courses to use digital computers" (p. 30). This was a very strategic depiction, as it framed computer engineering as one of the group's primary, original concerns, even though the early efforts of the committee were couched in terms of "computer science." In fact, in the early 1970s the COSINE acronym was often used without any reference to its original meaning, and the group's 1971 request for additional funding from the NSF was cleverly titled "Proposal for a Project in Computers in Electrical Engineering (COSINE)" (Sloan, 1973, p. 97). This evidence suggests that the group adjusted both its name and historical narrative to distance itself from the term computer science.

With regard to the survey data, the authors of the report summarized the results with a mix of enthusiasm and anxiety. On the one hand, they indicated that just under half (49%) of the responding schools had computer engineering options (p. 33). By comparison, 54% of these schools had computer science departments (p. 32). They also indicated an increase in the number of "digital faculty" in electrical engineering departments from one in 1965-66 to three or four in 1972-73, and with the total number of digital faculty at all schools more than doubling during this same time period (p. 32).[189] With regard to curricula, the authors noted that six of the basic courses recommended by COSINE for a computer engineering option were taught at 80% or more of all responding schools, and the majority of these schools offered six of the seven recommended elective courses (p. 37). And while the survey indicated that a growing number of these courses were taught by electrical engineering departments, a relatively large number of the core and elective subjects – especially in the areas of programming and software – were often taught outside of EE (p. 37). These conclusions once more pointed to a central tension that came with the growth of computer engineering education, namely that these types of programs were being established in ways that did not require electrical engineering departments to make large

[189] While not explicitly defined, "digital faculty" likely referred to those faculty whose primary research and teaching interests were in digital rather than analog technologies and techniques. Later in the report, the authors also noted that the presence of a computer engineering option was strongly correlated with the number of digital faculty at any given institution (p. 32).

teaching or research commitments in computer engineering, digital technology and techniques, and related areas.

The authors concluded the report by boasting that the "[t]he dominant theme of this report is the rapid growth of computer engineering in electrical engineering departments in the past seven years" (p. 38).[190] However, they cautioned that this rapid expansion was probably coming to an end, and they issued a number of caveats. To begin with, they used the collected survey data to conclude that the presence of computer science departments in colleges of engineering tended to inhibit the development of computing engineering options in electrical engineering departments.[191] The authors also referenced a number of other surveys and studies to suggest that computer science degree *programs* were still being established more rapidly than computer engineering options, although they noted that the most rapid growth of computer science *departments* was probably over.[192]

Through her dissertation research and a variety of derivative publications, Sloan attempted an even more thorough evaluation of "The Impact of the COSINE Committee on the Undergraduate Electrical Engineering Curriculum" (1973; 1974). Many of her conclusions are worth summarizing here. To begin with, she echoed the last of the COSINE surveys when she reported on the general expansion of computer engineering within electrical engineering education. More specifically, her research revealed that computer engineering options had been established in 16 of 46 (of 35% of) surveyed departments (1974, p. 185).[193] Along similar lines, she documented impressive growth in the number of computer engineering courses and faculty during the COSINE years. As yet another indicator of these trends, she noted that the number of

[190] In comparative terms, the authors added that "Computer engineering options did not start real growth until after 1965 but then developed so rapidly that the mean and median years for establishment of computer science departments and computer engineering options are the same, 1968" (p. 33). These were certainly impressive statistics, especially given the relative lag between these two educational movements.
[191] They also explained that the presence of computer science departments outside of schools or colleges of engineering did not have a similar impact. Further, it is difficult to determine whether the survey data showed causation or merely correlation with respect to these trends.
[192] As explained by the authors, "82% of schools without computer science departments do not plan to establish one within the next two years" (p. 32). The report also indicated that "the problem of the proper place for computer science departments has not been uniformly solved" (p. 32). More specifically, their data showed that 33% of computer science departments were in liberal arts colleges, 25% were in engineering colleges, and 24% were in other colleges (p. 32).
[193] The results of the larger 1972 survey – which indicated that just under 50% of schools had computer engineering options – was probably more accurate than Sloan's figure of 35%, which was based on a much smaller sample.

electrical engineering departments teaching computer organization as a subject or course had increased from 13% in 1965-66 to 69% in 1971-72 (1973, p. 64). Given these pieces of evidence, Sloan summarized that "[t]here has been considerable growth of computer engineering in electrical engineering departments along the lines recommended by the COSINE Committee" (1974, p. 189).

But what role did the COSINE Committee play in triggering or shaping these developments? The data presented by Sloan suggested that the COSINE reports played a particularly important role in ongoing efforts to establish computer engineering courses and options, especially as compared to the group's other activities (Sloan, 1974, pp. 182-183). On the other hand, she identified at least three major reasons why the ultimate impact of these documents was limited (p. 188). First, she explained that the distribution of the reports was relatively haphazard, and that the committee lacked sufficient resources to develop follow-up publications, such as textbooks. Second, Sloan added that the reports appeared relatively late, especially when compared to the ACM's recommendations. Third and finally, she noted that the COSINE reports featured "different and occasionally conflicting recommendations" (p. 188), including guidelines for a total of 23 different courses (Sloan, 1973, p. 62). As Sloan concluded, "When compared to the orderly curriculum of ACM 65 and ACM 68, the COSINE curriculum is hard to identify" (1974, p. 188). However, she failed to comment on how this ambiguity was linked to the group's reorientation from computer science to computer engineering in the late 1960s. In fact, like many other "insiders" she retrospectively framed "computer engineering" as the committee's primary concern throughout its history.[194]

Sloan's report also acknowledged some of the areas where the COSINE Committee had largely failed to initiate significant change. For instance, she concluded that "COSINE did not succeed in increasing computer-oriented material in traditional courses, except for circuits courses" (1974, p. 188). As noted above, this was one of the three main goals originally articulated by the committee, and the failure to make substantial headway in this area hints at the formidable barriers faced by those who wished to make widespread reforms in electrical

[194] For example, the abstract for Sloan's 1974 article summarized: "The COSINE Committee, formed in 1965, recommended that electrical engineering departments develop computer engineering courses" (p. 179). Elsewhere, she spelled out the group's original name, yet framed its activities as primarily focused on computer engineering.

engineering education.[195] The committee was also criticized for failing to involve smaller schools in its agenda. In fact, the group originally planned to address this problem through site visits to smaller and less prestigious schools, but few of these visits ever happened, and those that did tended to involve larger institutions (Sloan, 1974, pp. 181-182). As one of Sloan's survey respondents explained, "The actual base of COSINE seems to be a very small fraternity – as a result, it may be making recommendations that are inappropriate to many schools" (p. 182). The failure of the group to reach out to a larger constituency also helps explain why the expansion of computer engineering courses, options, and faculty seemed to be leveling off by the early 1970s. On a closely related note, Sloan added that context-dependent considerations such as department size and departmental policies against multi-option degree systems probably placed an upper limits on the total number of computer engineering options that could be established in electrical engineering departments nationwide, at least in the short term (1974, p. 185).

Sloan ultimately concluded that there were no clear causal relationships between the activities of the COSINE Committee and a variety of larger trends in electrical engineering education. Further, she indicated that she was unable to locate any department that had directly patterned its curriculum on COSINE recommendations. Sloan was therefore left with the rather simple conclusion that the group was visible and that it likely influenced or helped inspire the expansion of computer engineering. In fact, her own survey data revealed that the top sources of outside influence on the development of computer engineering courses in 46 electrical engineering departments included industry (6 responses), graduate schools (6 responses), the environment or "local interests" (5 responses), and COSINE (5 responses) (1974, p. 184). Along similar lines, Sloan only found a handful of textbooks that closely followed COSINE recommendations for course content. On the other hand, she noted that a number of COSINE members felt that the group had a favorable impact, although such insider responses are naturally highly subjective.[196]

---

[195] In fact, Sloan's evaluation indicated that the COSINE meetings and workshops had failed to attract many faculty members who were not already department heads or self-identified "computer engineers" (1974, p. 181).

[196] As Sloan explained, one of the COSINE Committee members felt that the group had significantly altered the course of many electrical engineering departments, especially with regard to their involvement in computer science and computer engineering. Other members suggested that "the main effect of the

In the end, Sloan got to the crux of the problem of causality when she noted that "an increase in courses recommended by a committee is creditable to the committee if only because the committee were good educational prophets" (1974, p. 188). One is therefore left with a rather simple conclusion, namely that the efforts of the COSINE Committee were significantly correlated with growth in many areas of computer engineering education. However, addressing questions of causality is not the central goal of my project. Instead, for the present analysis it is worth emphasizing that the activities of the COSINE Committee and the growth of computer engineering education were significantly co-produced, in that they reflected and reinforced one another. As a result of this process, computer engineering became an increasingly important marker for an expanding domain of electrical engineering education and a growing pool of computer-oriented electrical engineers. And while computer engineering was by no means a new term or even new domain of activity, from the early 1970s onward it was taking on important new meanings and trajectories in the academic sphere.

Conclusion

As noted in preceding chapters, by the mid-1960s a "sociotechnical parity" had been established in many areas of the computer field. For instance, the ACM and the IEEE Computer Group boasted roughly equal numbers of members, and each organization had carved out a partially distinct scope and constituency. In addition, hardware and software had emerged as markers for two distinct – yet intimately related – domains of technology, knowledge, and practice. The present chapter analyzed the historical development of two major spheres of computer-oriented education, namely "computer science" and "computer engineering." Here too we find a notable parity. By the early 1970s approximately 54% of more than 200 major universities had computer science departments, while only a slightly smaller percentage (49%) offered computer engineering options in departments of electrical engineering. And while the recommended courses and curricula for educational programs in both of these areas included some overlapping subjects, computer science courses and programs clearly tended to tilt toward software, applications, and programming, while the dominant mode of computer engineering education placed greater emphasis on hardware, design, and digital systems.

---

committee was to hasten development of computer engineering and of computer-oriented traditional courses by one or two years" (1973, p. 62).

In summary, the major social and technical schisms that had earlier emerged in the sphere of industry and in the system of professional societies were largely reproduced in the educational sphere, despite the efforts of many actors to challenge these boundaries. To put it more succinctly, computer engineering emerged as a foil to computer science – just as software had earlier emerged as a foil to hardware – and each of these two domains was linked to a partially distinct disciplinary settlement. Yet as noted above, this process happened neither overnight nor without significant struggle. The first phase of this historical trajectory centered on the objective of bringing the computer sciences into the fold of electrical engineering education, such as via new courses and degree programs. The early agenda and activities of Lotfi Zadeh and the COSINE Committee exemplify this approach.

However, using this strategy to reclaim large swaths of computing as a territory of electrical engineering was beset by difficulties. For example, Zadeh's own efforts to outline the contours of computer science "as a discipline" partially undermined his argument that the emergent field – or at least large parts of it – should be brought into electrical engineering. Further, this strategy suggested that such an amalgamation might challenge the respective dominant image of computer science as an independent discipline and electrical engineering as first and foremost a part of the engineering profession. In fact, Zadeh even went so far as to suggest that the electrical engineering field be renamed, while the 1967 COSINE report added that electrical engineering departments might grant dedicated degrees in computer science. Given the potential for such changes to generate disciplinary and professional instabilities, these proposals were surely a source of much anxiety, both for the proponents of computer science and the conservative old guard of electrical engineering education. And indeed, the events that unfolded at schools such as Berkeley and the University of Pennsylvania in the late 1960s and early 1970s revealed what kinds of disruptions were possible when the competing agendas of engineers and computer scientists came into direct contact and conflict.

In light of these challenges, we find that from the late 1960s onward the agenda and activities of the COSINE Committee were reoriented. In fact, COSINE publications and other sources reveal that the group's identity and history were rather swiftly reframed under the guise of "computer engineering." In a sense this change was largely rhetorical, especially given that relatively few electrical engineering departments were making concerted moves at the time to bring large swaths of computer science within their purview. However, the expanding roster of

COSINE reports did provide useful guidance and inspiration for engineering educators. The group's 1970 curriculum recommendations, for example, emphasized the extent to which a computer engineering "option" could be integrated into existing electrical engineering programs, including through new courses and offerings from other departments. These adjustments made moving into computer engineering a more tractable and appealing prospect for many departments. Further, the emergence of computer engineering courses and options was a pivotal development, as it seemed to bring the academic sphere into closer alignment with the scope and orientation of the IEEE Computer Society, the various divisions of labor that were increasingly prevalent in the private sector, and the sociotechnical boundaries of hardware and software.

Of course, one might question why the major goals outlined by the COSINE Committee did not emerge and gain traction earlier. After all, the preceding chapter revealed that commentators such as Rideout and Scott were describing computer-oriented programs in electrical engineering by the late 1950s and early 1960s, and a handful of universities were blazing important new trails in this area. Yet widespread change in engineering education is often notoriously slow and difficult, and the field of electrical engineering is no exception. It was increasingly evident from the late 1950s onward, for instance, that electrical engineering educators were moving slowly with regard to both incorporating computer use into a broad range of courses and providing students with a more balanced exposure to analog and digital technology. Further, it did not appear highly problematic that computer-oriented research and education was uneven from school to school, especially given that most electrical engineering departments only faced modest competition for students and resources from other academic units or colleges. In addition, resource restrictions meant that most departments could only move into some limited number of sub-disciplines and specialties, and so it was quite natural that some embraced and others avoided computer design and engineering. And finally, it is worth noting that the aforementioned Carr alleged in the late 1960s that industry had actively blocked university research and educational activities in some areas of "computer equipment and engineering.[197]

[197] At the 1967 Stony Brook conference, the aforementioned John W. Carr III argued that computer science never willingly left the area of "computer equipment and engineering." As Carr explained, "I think that everyone knows that the large computer manufacturers have done everything possible to abolish computer engineering within the universities. Anyone who has tried to get a grant for study of computer hardware – as a colleague of mine did for $10,000 and then had a large computer corporation

To be sure, a variety of factors contributed to the overall lack of university education and research in many areas of computing. However, the growth of computer science departments and programs reconfigured the disciplinary and institutional landscape in ways that many electrical engineering educators could no longer ignore. And despite the partial success of the COSINE Committee in responding to the rise of computer science and challenging the status quo of electrical engineering education, the future of computer engineering was by no means assured. It was rarely referred to or defined as a distinct discipline or field, for example, which revealed that it remained at least partially circumscribed and subservient to the electrical engineering profession writ large.[198] Further, there remained important points of contestation and overlap between computer engineering and computer science, especially in the academic arena. And whether these disciplinary settlements were sustainable in the long run remained an open question, especially in light of ongoing and dramatic changes in the technology and knowledge of computing.

After the COSINE Committee wrapped up its activities in 1972, a handful of commentators commented on the future of computer engineering. Coates, McCluskey, and Sloan, for example, insisted that the "expansion of computer engineering programs should still occur" (1973, p. 38). However, these authors recognized that numerous barriers stood in the way of realizing their vision. They explained, for instance, that "the reluctance of many electrical engineering departments to move into software, even into software engineering, may handicap their growth relative to computer science departments as the balance between hardware and software activities continues to shift" (p. 38). Sloan similarly noted that "most electrical engineering departments … had not been swayed by attempts of the COSINE to promote software, or what they termed 'software engineering'" (1974, p. 184). As I discuss in the

---

go to Washington and insist that this was now a function of the manufacturer – should make known this fact. I disagree heartily with the statement that people have gone outside the area of computer engineering voluntarily. We were forced, dragged kicking and screaming, away from computer equipment. If anyone wants to support my organization with funds to create some really imaginative computer equipment, I would certainly appreciate knowing of such sources" (quoted Oettinger, 1968, p. 38). Additional evidence for Carr's claim is difficult to find. However, the close collaboration between the COSINE Committee and industry suggests that this type of resistance was likely fading by the late 1960s and early 1970s.

[198] In fact, others had noticed the peculiar historical arc of "communication engineering." As engineer S. Seshu explained in a 1963 panel discussion, "The division of electrical engineering into branches thirty years ago was effected to permit the communication engineers to develop. Later we abolished the division when communication engineering matured, in order to pull up the others" (Cruz, 1963, p. 158). It was entirely possible that computer engineering would meet a similar fate.

following chapter, emergent areas such as “software engineering” were increasingly important domains of disciplinary negotiation at the intersection of computer science and computer engineering. Yet this was not the one among many points of overlap between the two fields. In the following chapter, I discuss how both the ACM and the IEEE made parallel moves into other emergent domains, including “computer architecture” and microprogramming.

There also remained questions about who would carry forward the torch of the COSINE Committee. Sloan, for example, noted in her 1974 evaluation of COSINE that the group had disbanded without successfully transferring its functions to another organization, such as the Education Committee of the IEEE’s Computer Society. In fact, she suggested that this alternate group was largely ill-equipped for such a task, especially given that it lacked resources, met infrequently, and was not working with a liaison from the COSINE Committee (Sloan, 1974, p. 182). Yet by 1974 the Computer Society’s Education Committee had launched its own efforts to develop an undergraduate curriculum in “Computer Science and Engineering.” As one member of the group explained, a major goal of this effort was to “bridge the tar pit” that existed at the intersection of computer science and computer engineering education (Mulder, 1975).

One might find the committee’s use of the compound phrase “computer science and engineering” somewhat surprising, especially given that many members of the COSINE Committee had strenuously emphasized that there were important distinctions to be made between computer science and computer engineering, especially in the educational context. In subsequent chapters I follow this issue into 1970s and 1980s, with particular emphasis on the competing forces of disciplinary integration and fragmentation that persistently swirled around the sociotechnical milieu of “computer science and engineering.”

Figure 6.1 – Hardware vs. Software: The Two Faces of Computers (Jensen, 1973, p. 14) © 1973 IEEE

Chapter 6

Janus-Faced Technology, Janus-Faced Field: (Re)Negotiating the Sociotechnical Settlements

The November, 1973 issue of *Computer* was topically dedicated to "hardware vs. software: the two faces of computers." Building on rich metaphorical foundations, the issue featured prominent graphical representations of Janus, the well-known Roman god of gates, doorways, beginnings, and endings who is often depicted with two faces gazing in opposite directions. In the suitable, high-technology revision of this mythology shown in Figure 6.1, an image of Janus – holding his traditional key and staff – was placed against a backdrop of circuit-boards, logic gates, and flow-chart symbols (Jensen, 1973, p. 14). The "two face" title and associated graphics revealed the extent to which computer systems could be viewed as coherent entities, albeit with distinct software and hardware "faces." This metaphor had much in common with the Humpty Dumpty analogy discussed in Chapter Four, which the editors of *Datamation* had trotted out in the mid-1960s. Yet the Janus image was arguably even more apt, given that it so effectively captured how the computer field was simultaneously – and perhaps paradoxically – both united yet divided, integrated yet fragmented.[199]

An introductory article authored by guest editor E. Douglas Jensen further described how the major themes of the special issue were related to the accompanying imagery. He started by explaining that the "interface" between operating systems and "computer architecture" was historically very "one-directional" and "unbalanced" (Jensen, 1973, p. 15).[200] Claiming that rapid

[199] Other historians of computing have found similar value in the Janus metaphor. Edwards, for example, states that "[c]omputers display, Janus-like, a double aspect. They consist simultaneously of hardware, whose heritage lies within the history of technology, and software, whose ancestry lies in mathematics and formal logic" (1996, p. xii). According to Edwards, this insight also helps explain the existence of a long divide in the history of computing between those accounts that focus on hardware and those that look at software. And like Edwards', I see my own analysis as challenging this historiographic divide.

[200] As I discuss in more detail below, the term "computer architecture" generally refers to the fundamental operational structure of a computer system.

technological developments – such as in the area of semiconductors – were tending to further accentuate the divide between hardware and software, Jensen called for the establishment of a more "symbiotic relationship between the operating system and the architecture, which requires bidirectional interaction between the two disciplines from the outset of computer design" (p. 15). The author's conflation of technology and discipline in this passage is telling, as it once more revealed the persistent intertwining of the social and technical, especially in discussions about the evolving relation of software and hardware.

However, actually achieving this type of "bidirectional interaction" was a challenging proposition. In fact, for almost two decades a variety of outspoken commentators had complained about the barriers that stood between hardware and software, computer designers and users. And despite both rapid technological change and ongoing discussions about how to bridge or even overcome these boundaries, Jensen's remarks suggest that the historical status quo had largely prevailed, even into the 1970s. And indeed, preceding chapters provide substantial support for this claim by documenting the evolving Janus-faced character of various worksites, educational settings, discourses, and technologies, especially from the 1950s into the 1970s.

Yet in previous chapters I largely sidestepped the parallel historical evolution of the major professional societies that maintained settlements in the computer field. The present chapter fills in this gap by focusing on the activities of the IEEE Computer Group – renamed the Computer Society in 1970 – from the mid-1960s into the 1980s. More specifically, I emphasize how various structures and processes of sociotechnical mediation both emerged during this period and helped maintain a modicum of stability, both within an expanding Computer Society and between the Computer Society and other groups, such as the ACM. By selectively enabling various flows of information, technical knowledge, people, and power, these structures and processes helped enable the emergence of an increasingly Janus-faced system of professional societies that consisted primarily of the Computer Society and ACM. Hence, the respective sociotechnical settlements claimed by these two groups moved toward unprecedented levels of overlap and interpenetration. In fact, I argue that the evolving relation of these two organizations bore an increasingly striking resemblance to the evolving relation of hardware and software. To put it another way, this chapter brings into further relief the coproduction of technologies and professional societies.

From Computer Group in Crisis to a More Autonomous Computer Society

As discussed in Chapter 3, by the mid-1960s a kind of parity had been established between the ACM and the IEEE Computer Group. Each society boasted more than 10,000 members, and they cooperated as peer organizations in AFIPS and the Joint Computer Conferences. These groups also claimed sociotechnical settlements that were partially distinct, yet also partially overlapping. As Willis Ware nicely summarized in 1963, "The IEEE is largely the hardware population of the computing field, and the ACM, largely the software population which has grown into information processing through scientific computing" (p. 42). Yet despite the apparent balance and stability that had been achieved in this "system of professional societies," countervailing forces were omnipresent. As noted previously, the joint conferences were increasingly dominated by ACM members and interests in the early and mid-1960s, and the leaders of the ACM were working hard to expand the group's membership, scope, and influence. Much of the "hardware population" of the field, on the other hand, was distracted by formation of the IEEE Computer Group out of the AIEE CDC and IRE-PGEC. And as the dust settled in the wake of this merger, many commentators felt that the state of the emergent organization was underwhelming, especially when compared to the ACM.

Anxiety about the position and future of the Computer Group was particularly evident in a series of letters and position pieces that were published in early issues of the *Computer Group News*. Established in 1966 and bearing a general resemblance in form and purpose to the *Communications of the ACM*, this new periodical was intended as an outlet for news, tutorial papers, summaries, and other material not suitable for publication in the more technical *Transactions on Electronic Computers*. Computer Group chair Samuel Levine opened the first issue with an appropriately anxious tone when he stated that "[t]he Computer Group faces a basic challenge in maintaining its role as the leading professional computer engineering society. Its membership has been relatively static in the past few years in spite of the continued rapid growth of the industry" (Levine, 1966). Levine went on to explain that this challenge was being met in a number of ways, including by publication of the *News*, broadening the technical coverage of *Transactions*, and initiating an aggressive membership drive. He also stressed the importance of the Computer Group's technical committees, especially as the organization moved into new areas of interest. As Levine explained in a subsequent letter, the "[a]ctivities of the

Technical Committees should reflect the changing scope of the theory and practices of computer sciences" (Levine, 1967a). I return to this theme in more detail below.

The Group also organized and executed its First Annual IEEE Computer Conference in 1967. Officially described as "a forum to meet the specialized requirements of the Computer Group Membership," the conference featured 38 papers and attracted more than 450 attendees ("First Annual," 1967; "Report on the Chicago Gathering," 1967). And according to a follow-up report published in *Datamation*, the event "was planned to fill what was felt to be a void in conferences for hardware specialists" ("Panels Feature," 1967, pp. 109-110). As suggested by this overview, the first Computer Group conference was a symbolically important development. On the one hand, the leaders of the Computer Group likely viewed existing events – such as the joint computer conferences – as neither fully in tune with their needs nor amenable to reform. On the other hand, initiating a new annual conference created yet another point of parity between the Computer Group and the ACM, the latter of which had been planning and holding its own annual meetings since its formation in 1947.

The conference therefore stood as a renewed symbol of independence, both for the Computer Group and its primary constituency of computer-oriented engineers. But the event also revealed persistent undercurrents of anxiety, especially as the participants took stock of their position in the computer field. Even before the event started, for example, a pre-conference digest set an appropriately introspective tone: "Due to the rapid growth of computer technology during the past decade, many people in the computer industry as well as academic institutions are confused about the role a modern electrical engineer should play in the computer field" ("First Annual," 1967, p. 1). In light of such concerns, the schedule included a panel session on "Computer Science in Electrical Engineering Curricula," which drew an impressive 300 attendees ("Report on the Chicago Gathering," 1967). According to one post-conference report, the resulting discussion closely paralleled other 1960s-era debates about computer science education, as documented in the preceding chapter. Well-known computer scientists like George Forsythe and Alan Perlis, for example, lobbied for the independent development of computer science as a discipline and in separate academic departments. Electrical engineering educator and COSINE member Mac Van Valkenburg, on the other hand, used the panel to emphasize both the

historical and prospective role of electrical engineering departments in computer science and associated areas.[201]

Related themes surfaced at a second, smaller panel discussion at the conference that was topically dedicated to "The Role of Electrical Engineers in Computer Science" ("First Annual," 1967, pp. 3-4). Reflecting the dominant image of the field's major sociotechnical factions, one report summarized that the panel was focused on "the touchy subject of the relationship between hardware and software (and their human representatives)" ("Panels Feature," 1967, p. 109). Another post-conference session report made an even more suggestive claim, namely that "the electrical engineer in computer science may be … a vanishing breed" (Fife, 1968, p. 20). This same report indicated that the panelists discussed a number of possible future roles for engineers in the computer field, while placing particular emphasis on increasing levels of specialization, systems-oriented work, and cross-disciplinary collaboration. They also discussed the engineer's role in the "messy" area of software, and one panelist noted that developing "more hardware" and enhancing the ability of engineers to deal with hardware/software trade-offs might provide some relief in this area. A final topic of discussion centered on the future role of the Computer Group, as well as its relationship with the ACM. And while the association of these two groups was described as one of possible "rivalry" or "competition," the panelists ultimately concluded that "it was the responsibility of Group Members, by active participation, to make the Group into whatever would serve them best" (Fife, 1968, p. 20).

Such remarks implied a bottom-up, democratic model for a professional society, where the group's larger scope and agenda ultimately reflected the will of its members. While this view was perhaps valid to a point, the leaders of the Computer Group were also stepping forward around this time with their own ambitious visions for the organization. One snapshot of this movement can be found in a planning document that was authored by Levine and published in 1968 in the *Computer Group News* (Levine, 1968). The former chairman started by once more noting that "the Computer Group has not advanced in pace with the rapid growth of the computer industry," and he added that "independent professional Societies in the computer field have been

[201] One conference report credited Van Valkenburg with delivering an especially witty attack on the proponents of computer science: "Van Valkenburg, who said he had been told in Russia that *the* drink was 'Vodka and'; Vodka alone, he was told, is colorless, odorless and tasteless. Taken with something else – caviar, for instance – it becomes exciting. So it is with computer science, which, alone, is colorless, odorless, tasteless. With EE it becomes palatable" ("Panels Feature," 1967, p. 109).

progressive and effective, both in terms of growth and providing service to their membership" (p. 16). Levine went on to argue that the group needed to reach beyond the "computer designers and systems engineers" who traditionally dominated the organization's membership roster. And indeed, a personnel survey conducted in 1968 and published in 1969 revealed that approximately 79% (or 7309 of 9310) of responding Computer Group members held engineering degrees (Davis, 1969, p. 7), while another report indicated that around 46% of Computer Society members were working in the hardware engineering area by the late 1960s ("Computer Society Members," 1972).[202]

In order to stimulate additional growth, Levine claimed that the Group should be concerned with "all facets of computer technology" (p. 17), and he noted that the organization was beginning to make concerted moves into software and applications. He also explained that the group's prospective members should include "professionals in electrical engineering, related engineering disciplines, computer science, programming and systems engineering" (p. 17). Such statements revealed the extent to which the Computer Group's ambitions increasingly extended beyond the province of "computer design and engineering," even to the point of including various pools of computer professionals who were much less likely to hold engineering degrees.

Levine also argued that "the Group should strive to become a semi-autonomous Society within the IEEE," thereby increasing its prestige, enhancing its appeal to those from a variety of disciplines, and providing the group with greater operational flexibility and better financial resources (p. 18). He added that such a move "would provide a basis for attracting other organizations to merge with the Group; for example one has indicated receptiveness to merge with the Computer Group if it is established as a semi-autonomous organization within the IEEE" (p. 18). While Levine refrained from identifying the organization in question, elsewhere he noted that it was desirable for the Computer Group to explore various merger possibilities, both within and beyond the IEEE.[203]

---

[202] When asked about occupational specialties on this same survey, the "Circuit, Component, and Logical Design" and "Systems Engineering Design" categories received the largest number of member responses. However, reasonably large numbers of members expressed involvement in other relevant areas of activity, such as "Systems Programming" and "Scientific and Engineering Applications."

[203] Evidence suggests that the organization in question was the Simulations Councils, Inc. In a short interview with incoming IEEE President Albert Hoagland published in 1972, one question noted that "[t]here has been some talk that the Simulations Councils, Inc. may merge with us" (Hoagland, 1972a).

Succeeding chairmen largely advanced Levine's agenda. Electrical engineer and consultant L. Charles (Charlie) Hobbs, for example, started his tenure as chairman with a letter noting that Computer Group's Administrative Committee (or "AdCom") was continuing to grapple with questions about the organization's "proper purpose, identity, image, and course of action" (Hobbs, 1968a). Further, he emphasized that the Group occupied a rather "unique position," sandwiched as it was between the IEEE and the rest of the computer field. And later the same year, he noted that the Group was ramping up a number of new programs, including expanded conference and publication activities, the establishment of new regional and technical committees, and moves toward greater operational autonomy (Hobbs, 1968b). When Edward McCluskey took over as chair in 1970, he placed particular emphasis on expanding Technical Committee activities and attracting new members to the group. In fact, McCluskey declared in his first letter in *Computer Group News* that 1970 was "The Year of the Opening," and he clarified by calling for "an opening of Group membership to a new type of colleague, an opening of technical committees to new types of members and activities" (McCluskey, 1970a, p. 3).

Yet a number of stubborn barriers stood in the way of realizing these objectives. As McCluskey was forced to acknowledge, there were persistent concerns that "the Computer Group would fail to realize its full potential as THE professional society for computer engineers as long as membership was restricted to those individuals willing and able to qualify as IEEE members" (p. 2). In order to address this issue, he noted that plans were being formulated to allow individuals to become members of the Computer Group without also having to join the IEEE. One important step toward realizing this goal came later in the year, when McCluskey announced that the IEEE had tentatively approved the Computer Group's petition for Society status (McCluskey, 1970b).

McCluskey also identified how this change was beneficial on at least two major levels. On the one hand, he noted that the Society's enduring affiliation with the IEEE provided uninterrupted service to the group's traditional constituency, namely "electrical engineers specializing in computers." On the other hand, he explained that associated changes would allow the group to better serve "that newer type of professional who regards himself [sic] as a computer engineer or scientist rather than an electrical engineer." McCluskey concluded his announcement by extending "an enthusiastic invitation to all who consider themselves computer scientists or engineers to join the IEEE Computer Society" (1970b). Such remarks reveal the

ongoing emergence of a distinct disciplinary identity for computer engineers and computer scientists, while also suggesting an ongoing differentiation of computer engineering and electrical engineering. However, many questions remained about whether these boundaries would deepen and expand, shift, or even fade away.

The transition from Group to Society status also led to changes in the organization's Constitution, and a number of these are worth highlighting here. To begin with, the 1965 version of the constitution included a statement of objective which indicated that "[t]he Group shall strive for the advancement of the theory and practice of the computer sciences" ("IEEE Computer Group," 1965, p. 2). The 1970 revision, on the other hand, stated that "[t]he Society shall strive to advance the theory and practice of computer and information processing technology" ("Provisional IEEE Computer Society," 1970, p. 33). As suggested by these passages, the leaders of the group were toning down their use of terms such as "computer science," which their predecessors had eagerly embraced in the mid-1960s. And by emphasizing words such as "technology," they returned to what had long been recognized as the organization's core focus.

The revisions also involved a significant adjustment to the Society's statement of scope. More specifically, the detailed five-part statement that appeared in the 1965 Constitution was dramatically simplified: "The scope of the Society shall encompass … [a]ll aspects of design, theory, and practice relating to digital and analog devices, computation and information processing" ("Provisional IEEE Computer Society," p. 33).[204] This broad-brush declaration strongly reflected ongoing efforts to expand the group's settlement into diverse domains. One final Constitutional change worth noting involved the rhetorical elevation of the group's "Chairman" and "Vice-Chairman" posts to "President" and "Vice-President." While this may appear a minor change in terminology, it was symbolically important because it created yet another point of parity between the Computer Society and the ACM.[205]

[204] This revision had much in common with the first section of the old statement, which declared that the group's scope covered "[a]ll aspects of design, theory, and practice relating to systems for digital and analog computation and information processing" ("IEEE Computer Group," 1965, p. 2).

[205] I realized the importance of this point in a recent conversation with Ed McCluskey. When I referred to him as a former chairman of the Computer Society, he quickly corrected me by noting that he had been a President of the organization. As McCluskey explained, "This was a big deal! Because the ACM had a President" (McCluskey, 2005).

In summary, the transition from Computer Group to Computer Society was an important turning point for this organization. In fact, some of the earliest calls to elevate the status and improve the autonomy of the group can be traced back to at least 1963 and 1964, when the AIEE CDC and IRE PGEC were merged with one another. And a variety of subsequent leaders – including Levine, Hobbs, and McCluskey – helped keep this reform movement alive through the 1960s and into the 1970s.[206] In more forward-looking terms, realizing Society status was highly synergistic with many of the other goals that had been set for the organization, such as attracting new members, building up the technical committees, nurturing new conferences, and expanding publications. Yet as one might suspect, the Computer Society's enhanced autonomy and rising aspirations raised new questions, both about its role in the computer field generally and its relations with the IEEE and ACM more specifically. In the sections that follow, I document how new processes and structures of sociotechnical mediation helped maintain stability in this system of professional societies. In fact, this stability is all the more impressive given the many destabilizing forces – ranging from frenetic growth to rapid technical change – that increasingly pervaded all phases of the field.

Expansion and Identity, Merger Talks and Mediation (Part I)

By the late-1960s, ongoing efforts to expand the Computer Group's membership started to bear fruit. To begin with, the group claimed approximately 10,000 members at the beginning of 1966, and more than 11,000 by 1967 (Levine, 1967b; Levine, 1968). In early 1968, chairman Hobbs boasted that the group's membership had passed the 12,000 mark, making it "the largest group in the IEEE" (Hobbs, 1968a). And by March of 1969, membership chair Tom Lindsay indicated that the membership had risen above 15,000, largely through ongoing efforts to recruit new group members from the ranks of the IEEE writ large (Linday, 1969). Growing the organization by 5,000 members in less than five years was an impressive feat. Yet as commentators such as Levine and McCluskey argued, a much larger pool of prospective

[206] In February of 1964, for example, chairman Walter Anderson suggested that the group be renamed the "Society for Electronic Computers." As Anderson explained, "We continue firm in our belief that a change in the group designation should include the replacement of the word 'Group' with one which signifies a larger organization" (Anderson, 1964a). And in April of the same year, Anderson noted that the group's proposed constitution and bylaws included the name "Society for Computer Sciences" (Anderson, 1964b). At the end of the year, however, incoming chairman Keith Uncapher announced that the "Computer Group" name was official, at least until the IEEE overhauled its nomenclature (Uncapher, 1964b).

members lay outside of the IEEE, and reaching out to them was an important long-term growth strategy. An appropriate affiliate or joint membership plan was sorely needed, so that individuals without engineering degrees or interests could join the Computer Group, but not the IEEE.

In the early 1970s, the Computer Society's increased autonomy and improved relations with the ACM helped set the stage for a new affiliate membership plan, while also triggering discussions about the more radical possibility of a merger. As incoming Society President Albert Hoagland explained in early 1972, "[W]e need to further explore the possible gains from closer working relations with the ACM" (Hoagland, 1972a). And just a few months later, a brief interview with Hoagland that appeared in *Computer Group News* more explicitly raised the question: "Should we merge with the ACM and leave the IEEE?" (Hoagland, 1972b). As Hoagland explained, survey data suggested that roughly one third of Computer Society members were also affiliated with the ACM. However, he added that approximately 75% of Computer Society members also belonged to other societies or groups within the IEEE. Once again, these data hinted at the extent to which the Computer Society was situated in a rather unique and perhaps even uncomfortable situation, sandwiched as it was between the IEEE and ACM, as well as a handful of other groups and societies.

Acknowledging the overlapping settlements claimed by these two groups, Hoagland also explained that the ACM and the Computer Society maintained "mutual areas of interest," yet he cautioned that "we don't compete directly with the ACM, nor should we." Hoagland added: "I don't believe a 'conglomerate' computer society would better serve the diverse interests of professionals in the field, although the bureaucratic potential may appeal to some." Perhaps not surprisingly, these remarks helped generate substantial follow-up debate and discussion. In the next issue of *Computer Group News*, for example, one letter to the editor argued that the Computer Society should leave the IEEE, yet the author added that "going to the ACM seems to be a cop-out" (Hettinger, 1972). Yet another letter similarly spoke out against a possible merger of the Computer Group and the ACM (Macnaughton, 1972). However, the author called for more interaction between the two groups, and he outlined some of the specific ways in which this might be accomplished.

By early 1973 the ACM and the Computer Society were moving even closer, as reflected in a swapping of Presidential messages. In his editorial that was published in the January 1973 issue of *Computer* (formerly the *Computer Group News*), ACM President Tony Ralston noted

the "current close relations" between the two groups, and he explained that around 5000 individuals were probably members of both societies (Ralston, 1973a). He suggestively added:

> At one time it may have made sense to characterize the Computer Society as the 'hardware' society and ACM as the 'software' society but it is doubtful that this makes sense any longer. Not only is the distinction between hardware and software becoming increasingly blurred by things such as microprogramming but, more significantly, it is becoming impossible for most 'software' people to do their jobs properly in ignorance of hardware or vice versa (p. 1).

Such passages once more revealed the extent to which the boundaries around the Computer Society and ACM remained deeply intertwined with both the contemporary state of computer technology and the actual practice and identity of professionals in the field. In fact, the ultimate justification for keeping the Computer Society and the ACM separate was in part called into question by ongoing technological changes, which made it increasingly apparent that the boundaries around hardware and software were neither obvious nor fixed.

In this same letter Ralston also explained that a special ACM committee had been formed to explore the Association's relationship with the Computer Society. Further, he discussed how cooperation between the two groups might be improved on four different levels. The first of these centered on a developing a range of specific activities, such as distributing publications to the members of both groups, providing cross-over discounts for conference registrations and publications, and expanding joint sponsorship of conferences and workshops. A second and somewhat higher level of cooperation involved coordinating and possibly even merging various Computer Society Technical Committees with counterpart Special Interest Groups in the ACM. Third, Ralston spoke to the possibility of a joint or affiliate membership plan. And fourth, the ACM President pointed to the prospects of a complete merger of the two organizations, although he identified a series of prerequisite conditions that would need to precede such a development.

The following month, a message from the Computer Society's President was published in the *Communications of the ACM*. And like Ralston, Hoagland hinted at the extent to which reevaluating the relationship between the two societies was significantly linked to larger currents of sociotechnical change:

> The growing interrelationship between hardware and software activities in the computer field has made this relationship a subject of active interest recently –

> since the Computer society is identified as primarily serving the engineering community while ACM does the same for programming (1973, p. 67).

Hoagland went on to describe the relation of these two group as complementary rather than competitive, and he once again spoke to the value of "bridge building" activities. In fact, he announced that the Computer Society had approved its own affiliate membership plan, with the ACM recognized as the first qualified affiliate society. For a discounted rate, ACM members could joint the Computer Society while foregoing membership in the IEEE. Hoagland also noted the possibility of a merger, and he explained that the required negotiations would necessarily involve the ACM, the Computer Society, and the IEEE. He also cautioned that none of the groups "at this time sees an imperative for merger in terms of its own self-understood roles and goals" (1973, p. 68). Yet regardless of questions about the feasibility or desirability of such a merger, Hoagland's comments revealed the extent to which the more autonomous and independent Computer Society was increasingly positioned between the IEEE and the ACM.

It is also worth underscoring the extent to which the identity of the Computer Society and its members and activities remained significantly linked to computer system design and engineering, especially through the mid-1970s. For example, Stephen Yau – who served as the group's President in 1974 and 1975 – frequently couched the identity and scope of the Computer Society in the terms of "computer engineering." And in one of his Presidential messages to the membership, Yau noted that the Society should "aim at serving all computer engineering professionals rather than only those with qualified backgrounds in electrical and electronic engineering" (Yau, 1974a). While this comment may initially appear confusing, it suggests that Yau was using the term "computer engineering" to cover a rather broad array of computer professionals, including many programmers and computer scientists who did not happen to hold engineering degrees. On the other hand, such remarks hinted at thorny questions about whether it was appropriate to frame this array of professionals as falling within the province of engineering.

The shifting identity of the Society and its members was also evident in its publication offerings and conferences. In late 1972, for example, the group's *Computer* magazine was given a new byline: "The Voice of the Computer Design Professional." But as one reader complained in a follow-up letter, the phrase was misleading given that "the majority of subscribers to *Computer* are involved in software, not in design" (Viehman, 1973). In light of such concerns the editors quickly adopted a new byline that read "The Voice of the Computer *System* Design

Professional," and they explained that "The motto ... had been amended to more accurately reflect the magazine's concerns in both the software and hardware areas" (Viehman, 1973).

The topical orientation of the Society's conference series provides another window into these themes. In 1971, for example, the fifth annual meeting was dedicated to the topic of "Hardware/Software/Firmware Trade-offs" – a clear reflection of the group's presence on the major boundaries of the computer field. In 1972 this same conference was given the catchy new "COMPCON" name, and was also topically dedicated to "Innovations in Computer Systems Design." As declared in a pre-conference report, this event was "THE conference on computer system design and engineering, both in hardware and software development" ("COMPCON," 1972). While other groups surely maintained overlapping interests with the Computer Society in many areas of hardware and software, emphasizing terms such as design, engineering, and development helped mark the group's sociotechnical settlement as partially proprietary.

Around this same time, a committee was formed to evaluate the conference series and establish a more consistent identity for the events. As committee member Rex Rice explained in a 1973 report, the event should primarily serve "system designers," and should not duplicate ACM or AFIPS events. Further fleshing out the conference's preferred "character," he added:

> On the one hand we can consider applications studies insofar as they affect architecture, and on the other hand we can delve into components and their use so long as we do not stray into the physics of device design. Between these extremes is ample room for software subjects, hardware subjects, circuit considerations, and many component subjects (Rice, 1973, p. 15).

Here we find yet another expression of the group's settlement, in this case positioning it squarely between end-user applications, on the one hand, and end-user applications, on the other. This statement also continued a longer historical trend of excluding from the group's purview those topics that were neither directly nor obviously related to the design of computer systems or components. In fact, and as noted above, such efforts can be traced back to the mid 1950s.

Yet despite these questions about the identity and orientation of the COMPCON series, these conferences appeared effective in bringing together the Computer Society's core constituency, namely computer system designers and engineers. Further, the continued success and vitality of the Computer Society's conference series was reflected in the shift to a bi-annual schedule beginning in 1974. Around this same time, the celebrated Joint Computer Conferences

were experiencing a sharp decline in attendance and revenue.[207] In fact, the long tradition of organizing two JCCs per year was ultimately deemed unsustainable, and beginning in 1974 the event was rechristened the National Computer Conference (NCC) and held just once per year. Indeed, the parallel rise of the COMPCON series and decline of the joint computer conference series is no historical accident. These events were surely competing with one another, and groups such as the Computer Society and ACM were ultimately committed to their own events.

It is further worth underscoring how the shifting landscape of computer conferences both reflected and reinforced a more general reconfiguration and renegotiation of the Computer Society's relation with the ACM from the late-1960s through mid-1970s. As discussed in Chapter Three, the Joint Computer Conferences, National Joint Computer Committee, and AFIPS had served as important common points of contact and negotiation for a system of professional societies that principally consisted of the ACM, IRE PGEC, and AIEE CDC. Yet as the scope, membership, and aspirations of both the ACM and the IEEE Computer Group/Society expanded from the mid 1960s onward, the joint conferences and AFIPS became less important in this system. To put it another way, the importance of these groups and events as "sociotechnical mediators" went into a period of decline.

The leaders and members of the Computer Society and ACM therefore increasingly sidestepped these organizations and conferences, and instead started to rely on more direct structures and processes of mediation. In fact, Ralston's remarks in particular hinted at how this mediation was being worked out on a number of different levels, ranging from joint publication initiatives and conference registration deals to affiliate membership plans and the coordination of committee activities. There also remained the possibility that the Computer Society might one day split off from the IEEE or even merge with the ACM. In the sections that follow I trace forward the Computer Society's expanding sociotechnical settlement, as well as its evolving relationship with other organizations.

Sociotechnical Expansion and Mediation: New Committees for Emergent Fields

By the late 1960s, the Computer Society was evolving through two distinct mechanisms. The first of these involved the group's expanding settlement, especially in terms of its

[207] In 1972, for example, Computer Society President Albert Hoagland noted "[t]he present downturn in the fortunes of the JCCs" (Hoagland, 1972a).

membership, scope, and activities. A second type of mechanism is more aptly captured by terms such as *mediation* and *negotiation*. On the one hand, various processes and structures of mediation maintained an overall balance within the Computer Society, even against the backdrop of rapid sociotechnical change. On the other hand, many of these processes and structures also helped create a modicum of stability between the Computer Society and other overlapping organizations, such as the ACM. In the sections that follow I provide a more detailed analysis of these processes by reviewing the Computer Society's historical trajectory through the 1970s and into the 1980s. I begin by taking a detailed look at the group's movement into emergent subdisciplines such as computer architecture, software engineering, and microprogramming, with particular emphasis on the establishment of new technical committees in these areas.

As noted above, by the late 1960s the leaders of the Computer Group were placing significant emphasis on the role and activities of technical committees. And indeed, the establishment of new committees stood as a potent institutional expression of the group's settlement in various phases of the computer field. Many of these committees also assumed important roles as "sociotechnical mediators." Early evidence for these themes can be found in a 1967 news item, where Levine summarized that the Computer Group's Technical Committee on Programming was charged with

> the important task of producing an *interface* with the more hardware oriented activities of the Computer Group. In addition, it should be the focus for the exposition of the development of software systems as an integral part of the development of computer systems ("New Programming Committee Chairman," 1967, my emphasis).

The use of the term "interface" in this passage is telling, as it revealed the extent to which the Programming committee was internally mediating the relation of the Computer Group's more hardware- and software-oriented factions, just as microprogramming and operating systems were viewed as key interfaces between physical computing machinery and its associated software and ultimate application(s). Further, Levine's remarks suggested that the programming committee's activities might actually improve the integration of software in the overall design of computer systems, reflecting a key point of cross-over between the committee's social and technical functions. By the late 1960s and early 1970s, a variety of new sub-disciplines were gaining

recognition in the computer field, and they became increasingly important sites for similar types of sociotechnical mediation.

Computer Architecture

The field of "computer architecture" is a particularly relevant example for the present analysis. As background, terms such as "architecture" and "architectural" started to creep into various computer publications in the mid and late 1960s, largely in the context of discussions about the organization and structure of computer systems. In 1969, for example, an article titled "Evolving Digital Computer System Architectures" appeared in *Computer Group News* (Joseph, 1969). As author Earl Joseph explained, the word architecture referred rather broadly to "a style of design or construction of computer systems" (p. 4). The term received another important boost in 1970 with the publication of Caxton Foster's *Computer Architecture* (Foster, 1970a). Describing computer architecture as both a "field" and an "art," Foster explained that the primary job of the computer architect was to "assemble the units turned out by the logical designer into a useful, flexible tool that is called a computer" (p. xi).[208] As suggested by this description, the title of "computer architect" was in many ways synonymous with "system designer," especially given that the latter type of worker had long been responsible for turning a variety of functional "building blocks" into working computer systems. In fact, the content of Joseph's 1969 article explicitly framed "computer system designers" as the main arbiters of computer architecture design decisions.

Yet Foster's broad use of the term placed somewhat greater emphasis on the "art" of computer design, and he explicitly argued that computer architecture work demanded a thorough familiarity with software. As Foster explained, the computer architect should be a "competent machine language programmer, preferably with experience in software systems" (xi). Similar themes surfaced in 1972, when Foster served as guest editor for a special issue of *Computer* that was dedicated to the topic of computer architecture (Foster, 1972). Foster's introductory remarks described computer architecture as a "recently recognized discipline" and a "profession," and he emphasized that practitioners in the new field were working on the boundaries of software and

[208] Foster also defined computer architecture as "the art of designing a machine that will be a pleasure to work with" (Foster, 1970a, p. xi). While perhaps tongue-in-cheek, this characterization hints at the notion that a computer architect is a special breed of computer designer who takes seriously the importance of usability and applications.

hardware. “A computer architect,” Foster explained, “should be aware of the problems of software development and the potentialities of hardware developments.”

The boundary-spanning character of computer architecture was further evident by the rapid movement of both the ACM and Computer Society into this emergent sub-field. The Computer Society’s Technical Committee on Computer Architecture (TCCA) was established in 1970, while the ACM’s Special Interest Committee on Computer Architecture (SICARCH) was founded in 1971 and elevated to SIG status by early 1972 (Foster, 1972).[209] The two groups also quickly cultivated a cooperative relationship, as reflected in a letter that was published in *Computer* in 1972. As reported by SICARCH chair Michael J. Flynn, participants at a joint meeting of the SICARCH and TCCA had raised a number of pointed concerns about the overall lack of cooperation between the ACM and the Computer Society (Flynn, 1972). In response, the two groups passed a unanimous resolution:

> Resolved ACM-SICARCH and IEEE CS-TCCA request that their parent organizations develop policies which will support our operational mergers within technical components, within chapters, and within students chapters the two societies appropriate to their mutual benefits.

Flynn went on to report that the Computer Society had 17,000 members and the ACM about 25,000, and he added that “a large number (five to ten thousand; no one seems sure) of these are joint members.”[210] He also asked how the interests of the field might best be served. Relating these issues to the educational sphere, Flynn noted that “the bifurcated professional attitude is reflected too often at universities with overlapping Computer Science and Electrical Engineering Department structures.”

---

[209] The founding of SICARCH also reflects the ACM’s movement into emergent boundary areas. In fact, Caxton Foster complained in a 1970 letter about the lack of ACM involvement in the area of computer organization (Foster, 1970b). "In glancing over recent publications of the ACM,” Foster explained, “I was struck by the dearth of papers on the subject of machine organization or architecture. ... [A] total of 6 out of some 330 publications of the Association for Computing Machinery bear on the subject of the design or organization of machinery for computing.” Foster concluded his letter by tentatively proposing the establishment of a Computer Architecture SIC. As noted above, it was founded the following year.

[210] According to an AFIPS survey that was conducted and published in 1971, approximately 20% (or 633 of 3,110) of responding ACM members also belonged to the IEEE, while 35.4% (or 633 of 1,790) of responding IEEE members also belonged to the ACM (Dickmann, 1971, p. 2). Extrapolating these figures to the full member populations suggests that at least 5,000 to 6,000 individuals were joint members of the two groups in the early 1970s.

It was fitting that these two particular groups stepped forward with such a call, especially given the extent to which the emergent field of computer architecture was positioned on the boundaries of computer science and computer engineering, software and hardware. And while the members of SICARCH and TCCA could do only so much to encourage a merger of their parent organizations, their cooperative activities helped model how the boundaries around the ACM and the Computer Society could actively be blurred. In 1973, for example, the two groups organized and co-sponsored the first in a long series of joint symposia on computer architecture. The second such symposium was held in 1975, and thereafter the event was held annually. Summarizing the state of the field, a report on the second symposium explained that the program for event was based on "the relatively unsophisticated but exacting view that architecture is the study of those aspects in the analysis and design of computers which specifically relate their structure and their function" (King and Garcia, 1975, p. 79). A 1976 description of the TCCA, on the other hand, noted that the committee itself was primarily concerned with "research and development in the integrated hardware and software design of both general-purpose and special purpose digital computers" ("IEEE Computer Society, 1976, p. 26). In fact, some of the first tools and techniques for developing such integrated systems were emerging around this time, and they helped stimulate the eventual emergence of a growing body of research in the area appropriately known as "hardware/software codesign." I return to this theme below.

As this overview reveals, the TCCA was acting as an important "interface" between the Computer Society and the ACM, just as computer architects were playing an increasingly pivotal role in mediating computer "structure" and "function," such as by improving the integration of software and hardware. Yet this same committee was also serving a similar role *within* the Computer Society. As noted above, the TCCA collaborated with the Technical Committee on Operating Systems to organize a "Workshop on the Interaction of Operating Systems and Computer Architecture" in early 1973. As nicely summarized in one report, the event was intended to encourage "further dialogue and interaction between operating systems and computer architecture" (Jensen, 1973, p. 15). Materials from the workshop were also published in a special issue of *Computer*, and guest editor E. Douglas Jensen explained that the publication might help "stimulate further dialogue and interaction between the two disciplines from the outset of the computer design."

Such comments reveal that improving the interface between software and hardware was viewed by many as a social and disciplinary problem as much as a technical challenge. In fact, one might surmise that one way to overcome the stubborn hardware-software divide was to encourage improved cooperation and communication between computer architects, systems programmers, and other factions of the computer design process. And indeed, this is precisely what commentators such as Carr had championed in as early as the 1950s. Hence, these types of workshops and publications helped ameliorate some of the sociotechnical barriers that were characteristic of the computer field generally, as well as the vertical organization of the Computer Society's technical committees more specifically. These themes are brought into further relief in the following sections, where I focus on the Computer Society's movement into the areas of software engineering and microprogramming.

Software Engineering

While the Computer Society's involvement in the emergent field of computer architecture in the 1970s is largely a story of intra- and inter-society mediation, the group's early movement into the domain of software engineering helps bring themes of both expansion and mediation into further relief. As background, the term "software engineering" first started to surface around 1967, when the first calls were made for an international conference dedicated to this newly-named domain. And as noted in the previous chapter, by 1967 ACM President Anthony Oettinger praised the emergence of the notion of software engineering (Oettinger, 1967), while his successor Richard Hamming argued in a 1968 lecture that "more than the usual engineering flavor be given to computer science" (Hamming, 1969, p. 3). While these commentators surely had manifold reasons for promoting an engineering-oriented view of software and computer science, they were clearly responding to a number of major challenges that were being grappled with around this time. Many critics, for example, were starting to complain that theoretically-oriented computer science programs were turning out graduates who were ill-prepared to undertake large programming projects. And just as importantly, discussions about a so-called "software crisis" were rising to a fever pitch, especially as software development budgets skyrocketed and hardware costs continued to fall.

The concept of software engineering gained further momentum in 1968 and 1969 through back-to-back conferences that were sponsored by the Science Committee of the North Atlantic

Treaty Organisation (NATO). As the organizers of the first conference explained, the phrase "software engineering" was deliberately selected as a "provocative" unifying theme for the event, although Mahoney has argued that the phrase was only provocative to the extent that it relied on terminology and metaphors that were vague, ill-defined, or uncommon (Mahoney, 2004, p. 9). Yet like other terms such as "computer science," the lack of a clear definition for the term was quickly overshadowed by its appeal, and by the late 1960s and early 1970s the phrase "software engineering" was appearing with increasing frequency in a variety professional and trade journals, including those published by the ACM and the Computer Society.

The Computer Society made its first formal claims in this emergent domain in the mid 1970s. In 1974, for example, the Computer Society Board approved both the formation of a Technical Committee on Software Engineering (TCSE) and the publication of a new periodical titled *IEEE Transactions on Software Engineering* (Yau, 1974a; 1974b). As President Sephen Yau summarized, this new journal covered "all aspects of specifications, design, development, management, test, maintenance, and documentation of computer software" (1974b, p. 2).[211] The Computer Society's movement into software engineering was also evident in May of 1975, when a special issue of *Computer* was dedicated to the topic. In addition, the Computer Society and the National Bureau of Standards (NBS) co-sponsored the First National Conference on Software Engineering, which was held in September of 1975. With 700 total attendees, the event was deemed a success. And in late 1975, a Prospectus for the Software Engineering TC was published in *Computer* ("Software Engineering Prospectus," 1975). By providing both a definition for the phrase "software engineering" and a description of the typical professional responsibilities of "software engineers," the prospectus further bolstered the group's settlement in this area.

The ACM made its own formal moves into software engineering in 1976 with the establishment of a Special Interest Committee on Software Engineering (SICSOFT) (Sammet, 1976). From the very beginning, the group established a cooperative relationship with its counterpart committee in the Computer Society. In the group's first newsletter, for example,

[211] In a subsequent announcement, the scope of the journal was outlined in more detail: "Devoted to the engineering aspects of computer software, the new Transactions will present state-of-the-art research papers in such specific areas as programming methodology, software reliability, system performance evaluation, software development management, and software development tools. Other areas covered will include hardware-software interface, man-machine interaction, software development for minicomputers, and the use of automatic programming" ("Announcing a Major new Publication," 1975).

editor Peter Neumann noted the "obvious overlap in scope between SICSOFT and the IEEE Computer Society Technical Committee on Software Engineering" (Neumann, 1976, pp. 2-3).[212] Further, he added that differences in the membership of each organization and the difficulties of a possible merger of the ACM and Computer Society helped justify this "duplication," and he explained that Tony Wasserman was acting as a liaison between the two groups through his role as both SICSOFT Vice Chairman and TCSE executive committee member (p. 3). The close relation of the two groups was also reflected in the Second International Conference on Software Engineering, which was jointly sponsored in 1976 by the ACM, IEEE, and National Bureau of Standards (NBS) ("Conference Report," 1976). When Wasserman took over as chair in 1977, SICSOFT claimed an impressive 2000 members, and by mid-1977 it was upgraded to SIG status (Wasserman, 1977a; 1977b). By this time it was clear that both SIGSOFT and the TCSE were well established.

Given that the leaders of the Computer Society appeared eager to expand the group's settlement into software and applications, the concept of "software engineering" emerged at an opportune time. By moving into this domain the organization could make strong claims in the sphere of software, while simultaneously retaining the engineering-oriented image of the group and its members. Publications such as the *Transactions on Software Engineering* helped legitimate and secure the Computer Society's settlement in this area, while the ACM's parallel movement into software engineering was mediated through the cooperative efforts of the TCSE and SIGSOFT groups. And just as the emergence of software engineering itself tended to call into question some of the deeply entrenched schisms that separated engineers and engineering from computer scientists and computer programming, the reasonably close and cordial relation of the TCSE and SIGSOFT revealed another important point of overlap between the ACM and the Computer Society.

Microprogramming

The area of "microprogramming" provides further evidence for some of the processes of sociotechnical mediation that were at work in the 1970s. Yet unlike relatively young fields such

[212] Neumann also noted overlaps between SICSOFT and other ACM SIGs, including SIGPLAN (Programming Languages) and SIGOPS (Operating Systems) (Neumann, 1976, p. 2). Yet he countered that SICSOFT might provide "a more global viewpoint," especially by emphasizing the many common subjects and interests that spanned these various groups. Hence, the SICSOFT group was assuming a mediating role both within the ACM and between the ACM and the Computer Society.

as software engineering and computer architecture, microprogramming had a longer history, and it was somewhat more closely bound to technology and the technological state of the art. Credit often goes to British computer pioneer Maurice Wilkes for coining the term in 1951, used to describe a new type of computer design with a modifiable rather than fixed instruction set (Wilkes, 1989).[213] With such a machine, a computer designer or programmer could more easily modify existing – or even create entirely new – operation codes. While this was certainly a novel and promising concept, limitations in memory technology, conservative cultures of computer design, and deeply entrenched schisms between computer designers and users all helped stall the commercial realization of the idea. However, the EDSAC 2 finally went into operation at Wilkes' own University of Cambridge in 1958 as the first microprogrammed computer, thereby helping to establish the feasibility of the concept (Wilkes, 1992).

An even more important development came in the early-1960s when computer giant IBM decided to implement microprogramming in a number of different models in their new 360 line of computers (Smotherman, 1999). And while successfully implementing the idea required significant research in memory technology, the company came to realize that microprogramming technology could greatly improve compatibility across a range of different computer models. By the mid-1960s, the proliferation of microprogrammed machines from IBM and other vendors also stimulated new approaches to simulation and emulation, where one type of computer could be configured to operate like another model. In light of these and other virtues – such as dramatically increasing the flexibility of a given system – microprogramming techniques were increasingly common from the late 1960s onward, as evidenced by an expanding assortment of publications on the topic, including full-length texts such as *Microprogramming: Principles and Practices* (Husson, 1970). As nicely summarized in one historical account: "Because of the success of the IBM System/360 product line, by the late 1960's microprogramming became the implementation technique of choice for most computers except the very fastest and the very simplest. This situation lasted two decades" (Smotherman, 1999).

Given that the technology of microprogramming was squarely situated on the boundaries of software and hardware, it is perhaps not surprising that the Computer Society and ACM maintained overlapping interests in this area. Yet collaboration once again prevailed over

[213] It is worth noting that Wilkes' background was not in engineering, but rather mathematics. Once again, we find an important development in the area of computer design coming from a non-engineer.

competition. By the late 1960s, for example, the Computer Society and ACM were co-sponsoring a series of annual workshops on microprogramming, which were continued in the 1970s as the two groups made more formal moves into this domain. By at least 1970, an ACM Special Interest Group on Microprogramming (or SIGMICRO) was ramping up its activities, including through the publication of a newsletter (Carlson, 1970). On the other hand, the Computer Society's Technical Committee on Microprogramming was finally established in 1974 (Yau, 1974b, p. 3). Cross-talk between the TC and the SIG was very evident, as suggested by the continued joint sponsorship and organization of a long series of annual microprogramming workshops. And by 1975, a single individual – IBM microprogramming expert J. Michael Galey – was simultaneously serving as the chair of *both* ACM SIGMICRO and the Microprogramming TC (Galey, 1975). To a significant extent, the two groups were functioning like a single entity.

It is worth noting that the increasingly close alignment of these and other committees and groups was recognized at the time. As nicely summarized in a 1975 Computer Society subcommittee report, "Many Technical Committees of the Computer Society are paralleled by similar Special Interest Groups/Committees within the ACM. The overlap in interest has been significant enough in some cases that the TC and SIG have operated virtually as a single organization with key individuals holding dual offices" (Salisbury, Snyder, and Smith, 1975). However, the authors added that ongoing efforts to operate some combined TCs and SIGs as a single organization "have generally been opposed by the parent organizations, placing emphasis on differences between the two societies rather than the similarities." Such remarks once again revealed the many challenges faced by those who favored a partial or even full merger of the two parent organizations.

In August of 1975, the aforementioned Galey served as guest editor for a special issue of *Computer* that was topically dedicated to "Microprogramming: The Bridge between Hardware and Software." In a simple illustration that accompanied Galey's introduction to the feature articles, a computer chip appeared suspended in a chasm, acting as a sort of bridge for a curving road that flowed from one side of the valley to the other. It was an apt visual metaphor, given that microprogramming technology had grown up on the boundaries of hardware and software. In fact, new terms such as "firmware" emerged and started to gain traction in the late 1960s to

describe this expanding, intermediate level of computer technology. A 1967 *Datamation* article by Ascher Opler is often credited as the original source of this term (Opler, 1967). [214]

Yet in light of the preceding historical review, this line of analysis can be taken one step further. That is, the bridge metaphor worked equally well in describing how the Technical Committee on Microprogramming and its SIGMICRO counterpart had emerged as crucial bridges between the ACM and the Computer Society. To put it another way, these two groups – as well as a number of other TCs and SIGs – acted as the metaphorical "firmware" between the more hardware-oriented Computer Society and the software- and applications-oriented ACM. Further, it was no coincidence that these sorts of bridging activities grew up in areas where these organizations and their members maintained overlapping interests. Below, I discuss in more detail this striking similarity between the actual organization of computer technology and the structure and relation of these two professional societies. Yet in order to provide additional background details for this argument, it is first necessary to more generally document the Computer Society's technical committee structure through the 1970s and into the 1980s, as well as the group's more general evolution and expansion.

Expansion and Identity, Merger Talks and Mediation (Part II)

As detailed in the preceding section, technical committees were an increasingly important part of the Computer Society's growth in the 1970s. In fact, by 1976 the Society consisted of 17 technical committees ("IEEE Computer Society Technical Committees," 1976), and in 1978 the ongoing expansion of the committee structure prompted the formation of two umbrella Technical Interest Councils (TICs) ("Technical Interest Councils," 1979). With one TIC encompassing the area of "Systems Technology" and another dedicated to "Software and Applications," the TICs were established to review, coordinate, and manage an increasingly large and unwieldy assortment of committees. Further, one cannot help but notice that the divide between the TICs roughly reflected the persistent hardware-software schism. The establishment of the TICs can

[214] As Opler explained, "I believe it worthwhile to introduce a new word into our vocabulary: *firmware*. I use this term to designate the microprograms resident in the computer's control memory, which specializes the logical design for a special purpose, e.g., the emulation of another computer" (p. 22, author's emphasis). Referencing Opler's article, microprogramming pioneer Maurice V. Wilkes summarized in 1969 that "firmware may take its place along with software and hardware as the main commodities of the computer field" (Wilkes, 1969, p. 143).

therefore be viewed as a structural fix that brought the Computer Society's two faces into further relief, especially as the group's settlement covered an ever-larger sociotechnical territory.

The growth of the Society continued in subsequent years, and by 1979 the group claimed 20 technical committees. President Oscar Garcia noted that the number of TCs had increased to 26 by early 1982, and he added that committees had recently been formed to cover "such new areas as distributed processing, multiple-valued logic, optical processing, computer graphics, computers and the handicapped, computational medicine, VLSI [Very Large Scale Integration], and office automation" (Garcia, 1982). The number of committees jumped to 30 by late 1982, and the roster of newly established TCs covered areas as diverse as computer languages, personal computing, and robotics ("Chairmen Named," 1982). And in a 1983 report, Roy Russo – the Computer Society's Vice President for Technical Activities – nicely summarized the pivotal role that the technical committees had assumed within the organization (Russo, 1983). To begin with, he noted that the TCs were extensively involved in running conferences, developing standards, publishing newsletters, and sponsoring special issues of Computer Society publications (p. 3). Referring to the TCs as "the backbone of the Computer Society," his remarks also hinted at the role that the committees were playing as sociotechnical mediators. "[O]ur technical activities," Russo explained, "provide the principal mechanism for cooperation with other professional organizations both within and outside the IEEE, for example, the Communications Society [of the IEEE] and the ACM" (Russo, 1983, p. 6).

Membership contests and other campaigns helped the Computer Society maintain a steady pattern of growth into the mid 1970s, and total membership finally passed the 20,000 mark by the end of 1974 (Yau, 1975, p. 3). By 1975 the group was entering a period of more rapid development, fueled in part by both the expansion of the Society's activities and the addition of large numbers of new affiliate and student members. As outgoing President Yau boasted in 1976 – the year of the group's 25th anniversary – "the Society is stronger and healthier than ever," and he added that membership had risen above 23,000 (Yau, 1976, p. 4). Subsequent announcements revealed that the society was comprised of 33,000 members by late July of 1978 – an increase of more than 10,000 individuals in less than two years (Smith, 1978). As explained by Computer Society President Merlin Smith, these statistics were attributable to both the growth of the computer field and the responsiveness of the Society to member interests. But regardless of the ultimate reasons, the expansion largely continued unabated.

References to the Computer Society as mainly a computer engineering society were also increasingly rare in the mid and late 1970s, and a 1977 update to the Society's constitution helped made more explicit the group's increasingly expansive settlement. For example, the terms "application" and "science" were inserted into the group's stated objectives: "The society shall strive to advance the theory, practice, and *application* of computer and information processing *science* and technology" ("IEEE Computer Society Constitution," 1977, p. 108, my emphasis). A similar change was also made to the Society's official statement scope, with the updated version declaring: "The scope of the society shall encompass all aspects of theory, design, practice, and application relating to computer and information processing science and technology" (p. 108). And if there remained any doubts about the group's wide-ranging agenda around this time, President Merlin G. Smith explained that "society leadership is now committed to provide technical programs and publications across much of the total hardware, software, and applications spectrum" (Smith, 1977a).

From the late 1970s onward this broad commitment was realized in a variety of ways, including through the formation of technical committees, as noted above. New publications and conferences provide additional evidence for this trend. As noted above, *Transactions on Software Engineering* was added as the group's third major publication in 1975. *IEEE Transactions on Pattern Analysis and Machine Intelligence* was launched in 1979, while *IEEE Computer Graphics and Applications* and *IEEE Micro* (which was dedicated to microcomputers and related topics) were added in 1981. The addition of *IEEE Design and Test of Computers* and *IEEE Software* in early 1984 raised the group's total number of regular publications to eight. New conferences and workshops also reflected the group's growth, and by the early 1980s the Society was annually sponsoring or co-sponsoring as many as 45 to 50 special interest conferences, workshops, and symposia (Yau, 1981).

One new conference of particular note was the first IEEE Computer Society International Computer Software and Applications Conference (COMPSAC), first held in 1977. As one pre-conference report explained, "COMPSAC 77 will bring together computer practitioners, users, and researchers to share their ideas, experiences, and requirements for applications software, management techniques, and software development support, including automated techniques" ("COMPSAC 77," 1977). As suggested by this description, the Computer Society was serious about expanding its settlement to cover significant swaths of software and applications.

Membership growth was also rather dramatic from the late 1970s into the 1980s. In fact, by early 1980 the size of the Society had reached parity with the ACM, with both groups claiming about 44,000 members (McCracken, 1980, p. 66; "Record Growth," 1980). By the end of 1981, total membership rose above 62,000. On the one hand, this number made the Computer Society the largest society within the IEEE, *by a factor of three* (Bonn, 1982, p. 4). In fact, Division Director Dick Simmons noted in 1982 that the Computer Society was growing at twice the rate of any of the IEEE's 31 technical societies (Simmons, 1982, p. 6). On the other hand, the leaders of the Computer Society could finally boast that they were at the helm of the "the largest professional technical society devoted to computers" ("IEEE-CS Membership," 1982). This was no small feat, as the expansion of the ACM had long outpaced the Computer Society. And in October of 1982, President Oscar Garcia noted that the membership stood at 66,605, which represented a doubling of the size of the organization in less than four years (Garcia, 1983, p. 5). For reference, ACM President David H. Brandin reported around this same time this his organization was comprised of about 57,000 members (Brandin, 1982a, p. 769).

In a replay of the early 1970s, this level of expansion helped trigger renegotiations of the Computer Society's relations with both the IEEE and ACM. On the IEEE side, for example, the Computer Society made bold moves in 1980 and 1981 to change the name of the parent institute. As Dick Simmons – who at the time was serving as a division director on the IEEE Board of Directors – explained:

> Now might be the time to consider changing the name of the institute from the Institute of Electrical and Electronics Engineers to the Institute of Electrical and Computer Engineers. In my opinion this would encourage increased participation of the computer professional in both the Computer Society and the IEEE (Simmons, 1980).

This was a striking development, as it suggested the field of computer engineering was taking on a distinct professional and disciplinary identity. And while this particular measure had broad support among the leaders of the Computer Society, in 1981 it failed to gain sufficient traction among the leaders of the parent organization (Feng, 1980; Bonn, 1982, p. 4). The Computer Society was more successful in 1982 when it secured a second division director slot on the IEEE Board (Bonn, 1982; Garcia, 1983). This new position gave the Computer Society control of two

of eight total division director slots, effectively doubling the group's representation in the top leadership structure of the IEEE (Bonn, 1982, p. 4).

Yet as many surely recognized at the time, the Computer Society's prominent position – both with regard to the IEEE specifically and the computer field generally – was at least partially due to an expanding roster of affiliate members. Membership and Information President Dennis Fife reported that the Society was comprised of almost 72,000 members by the end of 1982, and he noted that roughly a quarter of the 10,000 members that had been added in 1982 were affiliate members. "Many of these members are not engineers by training," Fife explained, "but many of them either participate heavily in or, in fact, lead activities of the society that benefit all members" (Fife, 1983, p. 6). And in 1984, when the Computer Society boasted somewhere in the neighborhood of 80,000 members, an editorial on the group's relationship with the ACM revealed that somewhere between 15,000 and 20,000 individuals were paying dues to both organizations (Carlson and Simmons, 1984).

As suggested by such statistics, the expansion of the Computer Society had deepened the group's overlap with the ACM, while also eroding the dominant image of the Society as first and foremost a computer engineering organization. In fact, questions about whether it was sustainable or desirable to maintain this bifurcation between the computer field's two major professional societies received renewed attention in the early and mid 1980s, especially under the leadership of Computer Society President Oscar Garcia. Some of the earliest evidence for this trend can be found in a special message from ACM President David H. Brandin that was published in the November 1982 issue of *Computer*. While ostensibly concerned with the topic of technology transfer – at a time of growing concern about the global competitiveness of the U.S. computer industry – Brandin added that "Oscar Garcia has said that there are no two technical societies on the face of this earth that have more in common than ACM and the IEEE-CS. I agree with him" (Brandin, 1982).

Another important touchstone for this movement surfaced in mid-1983, when Garcia and Bradin authored a special joint message that was published in both *Computer* and the *CACM*. Suggestively titled "Where do parallel lines meet? or The Common Goals of ACM and the IEEE-CS" (Brandin and Garcia, 1983), the two Presidents noted that "[t]he distinction between the two societies is no longer clear," and they added that increasing levels of intersociety competition reflected ongoing efforts "to break out of the binary mold of thinking – hardware vs.

software, engineering vs. computer science" (p. 6). The two presidents indicated that both the Computer Society Board and the ACM Council had recently voted in favor of resolutions that supported accelerated cooperation and possibly even a merger between the two groups, and they added that an *ad hoc* Committee for Intersociety Cooperation had been established (pp. 6-7).

In a summary report published in early 1984, the leaders of this committee identified a series of common objectives for the two groups, as well as some areas were cooperative progress was being made (Carlson and Simmons, 1984). In fact, they discussed five key areas of interest or concern that were shared by the two groups, namely membership overlap, educational activities, conferences, technical committees and special interest groups, and publications. As a more specific example of the cooperative efforts that were once more gaining momentum, Carlson and Simmons pointed to a 1983 analysis that revealed that 19 of the Computer Society's Technical Committees had "a significant correspondence" to an equal number of ACM Special Interest Groups (p. 89). Conversely, only 12 TCs and 14 SIGs had minimal or no overlap.

Such statistics revealed the increasingly overlapping and interpenetrating character of the settlements claimed by these two professional societies, while also revealing the potential for further crosstalk and coordination. As the authors explained, an aggressive program for promoting cooperation between TCs and SIGs had been proposed, and a joint meeting of all TC and SIG chairs was planned for later in the year. The report also noted that plans were underway to develop a new joint publication for the members of both societies, although the authors acknowledged that preliminary discussions had revealed that such an endeavor was beset by problems that were "editorial, technical, administrative, legal, financial, and even political in nature" (p. 89). In light of such challenges, Carlson and Simmons admitted that "a full merger will not occur rapidly," but they nonetheless spoke to the value of ongoing efforts to identify common areas of interest and improve service to the members of both groups.

Martha Sloan – who served as Computer Society President from 1984 to 1985 – ended her term by repeating the message that the organization should "[b]uild stronger bridges to IEEE and ACM," and she called on her colleagues to "meet the challenge of unifying the computing profession by improving coordination with ACM while exploring possibilities of a merger" (Sloan, 1985, p. 7). Yet in the wake of Sloan's Presidency, merger discussions were gradually

superceded by more pressing matters.[215] In fact, both groups were making important moves in the area of conferences in the latter half of the 1980s. For example, the cordial mid-1970s relation of the ACM and the Computer Society helped lead to the establishment of a new Fall Joint Computer Conference (FJCC) series. Intended as a replacement to the ACM's National Conference and the Computer Society's Fall COMPCON, the first meeting was held in 1986 and attracted 3000 people ("ACM and IEEE-CS Launch," 1987). Yet in a somewhat surprising turn of events, low attendance at the second of these events in the led the Computer Society and ACM to cancel FJCC 88 and its successors ("The Last FJCC," 1987).

In a sense, this turn of events symbolized the extent to which the two parent organizations had once again backed away from a possible merger. It is also worth noting that the failed revival of the FJCC was organized outside the purview of AFIPS. In fact, the event looked like something of a throwback to the mid-1950s, when the joint computer conferences emerged as an important common point of contact and negotiation for three main groups, namely the ACM, AIEE CDC, and IRE-PGEC. And despite the failure of this new joint conference, other developments made it clear that the leaders of the ACM and the Computer Society preferred to work out their relationship directly, rather than through intermediaries such as AFIPS.

This theme was brought into further relief in 1987, when both organizations gave notice that they were formally withdrawing from the National Computer Conference series (Abrahams, 1987; "Computer Society Votes," 1987). After attendance peaked at 100,000 in 1983, the NCCs entered a downward spiral, and by 1987 attendance was estimated at a paltry 14,000. Both groups expressed concerns that the events were no longer meeting the needs of the professional community, and were also a potential financial liability for the sponsoring organizations. In fact, the Computer Society Board even went so far as to pass a movement calling for the cancellation of all future NCCs ("Computer Society Votes," 1987). The leaders of AFIPS voted to continue the event, but by this time the writing was on the wall. The last NCC conference was held in 1987, and AFIPS itself was dissolved in 1990.

[215] In a 1987 commentary on the ACM's development over roughly the preceding fifteen years, Eric Weiss noted that the ACM had long maintained "cordial working relationships" with "natural rivals" such as the Computer Society, yet he added that "attempts to merge the two always fail" ().

Conclusion

On the one hand, the demise of the Joint Computer Conference series in 1987 marked the end of a historical era that can be traced back to the first such event in 1951. On the other hand, by the late 1980s it was clear that other and more direct forms of sociotechnical mediation had emerged to maintain stability both within an expanding Computer Society and between the Computer Society and the ACM. These mediators ranged from affiliate membership plans and jointly-sponsored conferences to coordinated publication activities. In this chapter I also placed particular emphasis on the mediating role of the Computer Society's Technical Committees and the ACM's Special Interest Groups. As noted above, from the 1970s onward these sub-groups became key points of contact and negotiation – or "interfaces" – between the two parent organizations. More specifically, these groups and their activities helped smoothed the flow of information, people, and power, both within and between the Computer Society and the ACM.

On a closely related note, my analysis also describes the striking structural similarities between the dominant model of computer system design and the structure and relation of these two professional societies. Just as the SIGs and TCs acted as "interfaces" between the hardware-tilted Computer Society and the software- and applications-oriented ACM, so too did intermediate levels of technology such as firmware and operating systems act as bridges between the physical hardware of computing and the associated software and applications. Further, many of the SIGs and TCs that were cooperating most closely – sometimes even to the point of near-merger – were situated in sociotechnical boundary-areas, such as microprogramming, computer architecture, and software engineering. The Janus-faced character of computer technology was therefore both reflected in and reinforced by the computer field's Janus-faced professional societies.

Yet my claims about the historical coproduction of the social and the technical are not entirely without precedent. In a 1968 *Datamation* article, for example, researcher Melvin E. Conway tentatively worked in similar directions when he argued that "organizations which design systems … are constrained to produce designs which are copies of the communication structures of these organizations" (1968, p. 31). He used the term "homomorphism" to describe these "structure-preserving relationships," and he cleverly noted that a given organization will tend to "stamp out an image of itself in every design it produces" (p. 30). Applying this idea to the example of working computer systems, the author discussed how the hardware, system

software, and applications associated with such systems were frequently linked to three distinct sets of "designers," namely the computer manufacturer's engineers and system programmers, and the end-user's application programmers. Emphasizing the extent to which the design of a technology often reflected pre-existing channels of organizational communication, Conway also noted "[t]hose rare instances where the system hardware and software tend to cooperate rather than merely tolerate each other are associated with manufacturers whose programmers and engineers bear a similar relationship" (p. 30-31). Following this line of reasoning, the author argued that effective design required a flexible and lean organization with good communication between design groups.

The ideas presented by Conway in this article remain novel and thought provoking, and his main thesis eventually gained fame in computer circles as "Conway's Law."[216] Yet the thrust of his article remains limited by its unidirectional character. That is, it posits that technological design can be inflected by organizational structures, but it fails to discuss the reciprocal shaping of organizations by the structure or design of technology. A more recent text helps fill out the other half of this equation. In *Design Rules, Volume 1: The Power of Modularity*, business and economics experts Carliss Baldwin and Kim Clark analyze the historical evolution of computer technologies, firms, and markets (2000). In so doing, they emphasize the crucial importance of modularity in computer design and development from the 1960s onward, especially against the backdrop of dramatic increases in the complexity of computer technology. The more specific cases they discuss include the IBM 360 and Digital PDP computer lines, both of which featured modular designs that helped enable the establishment of new "modular clusters" of manufacturing firms in a variety of niche vertical markets, producing add-on products ranging from disk drives and display terminals to circuit boards and pre-packaged software. In summary, Baldwin and Clark essentially reverse Conway's Law by documenting "the structure of the design [of computer systems] influencing the structure of firms and markets in the surrounding industry" (p. 15).

[216] As described in one well-known hacker's dictionary, Conway's Law is "[t] he rule that the organization of the software and the organization of the software team will be congruent; commonly stated as 'If you have four groups working on a compiler, you'll get a 4-pass compiler'. The original statement was more general, 'Organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations'" ("Conway's Law," 2003).

In line with the work of Carliss and Baldwin, the structure and relation of the Computer Society and the ACM significantly reflected the dominant structure of computer technology. Yet one must be wary of the technological determinism that can accompany such a description, where computer technology is implicitly or explicitly framed as a primary or even singular shaper of social structure. Carliss and Baldwin also place significant emphasis on the market, which at times shades into economic determinism. Conway's analysis therefore acts as a corrective to this view by suggesting that the structure of organizations and institutions can powerfully inflect the processes and products of technological design. And indeed, this type of thinking resonates with a long line of reformers who argued that the improved integration of physical computer machines and their ultimate application could be achieved by encouraging enhanced communication between – and perhaps even the integration of – hardware and software design groups.

For the present analysis, however, I set aside questions of directionality – even if one may find specific cases and examples that provide additional support for either Conway's Law or the claims of Carliss and Baldwin. Instead, my primary focus is on larger patterns of sociotechnical coproduction that span diverse contexts. My account emphasizes the persistent intertwining of the social and technical structure, reflecting and reinforcing one another, sometimes even to the point of being indistinguishable. In fact, my approach has some continuity with Peter Galison's analysis of 1940s-era plans for the organizational structure of the Radiation Laboratory at MIT, which tended to mirror the various technological systems that were in development there. As summarized by Galison, "Material objects – those building blocks of microwave devices – were inseparable from … questions of administrative and conceptual control" (1997, p. 247). As suggested by Galison's use of the term "inseparable" in this passage, the theoretical concept of "co-production" nicely captures this deep intertwining of social and technical structure.

And while the present chapter traced these processes of coproduction in the sphere of professional societies, in prior chapters I analyzed similar phenomena in the context of worksites and educational settings. In fact, my analysis reveals that the structural similarities that are evident in these diverse contexts are not a historical accident, but rather the outcome of mutually reinforcing processes. In the following chapter I return to the academic sphere in order to document how these larger currents of sociotechnical mediation and change inflected the

ongoing evolution of educational programs in computer engineering, computer science, and related areas. My analysis also speaks to the importance of educational contexts as crucial sites for both the building and differentiation of disciplines. In fact, the present chapter hints at the extent to which the Computer Society's sociotechnical settlement expanded well beyond the domain of computer design and engineering, especially through the 1970s and 1980s. As a result, university departments and programs became increasingly important sites for establishing, defining, and growing computer engineering as a distinct educational and disciplinary domain.

Chapter 7

Bridging the Tar Pit?:

Constructing CSE and Computing Education, circa 1974-1991

Imagine for a moment that the year is 1973. The setting is one of the offices of the Sperry Rand corporation. Down one long corridor we find the office of Michael Mulder, a mid-level manager in the company's UNIVAC division. Peering into the office, we find that Mulder is in the midst of an interview – one of many that he has conducted in recent months with fresh college graduates. In its ongoing efforts to assume and maintain its position as a key player in the fast-moving and cutthroat computer business, Sperry Rand has an almost insatiable appetite for new technical staff. Yet despite the fact that the current interviewee has been trained in a computer-oriented degree program within a department of electrical engineering or computer science, he appears woefully unprepared for the type of work that he will face as a new Sperry Rand employee. Mulder is increasingly discouraged.[217]

In light of his own academic and professional background – including a Ph.D. in electrical engineering earned a few years prior, as well as extensive stints in computer system design and development at Sperry Rand and elsewhere – Mulder had a sense for what it takes to be successful in the field, and many of his interviewees simply did not seem to have the right types of knowledge and skills. As Mulder later explained in an article he authored, "Potential new hires … lacked adequate breadth of training for industry. All too often their academic background appeared to be confined to one or the other of the two major divisions of our profession – hardware or software" (Mulder, 1977, p. 70). While preceding chapters reveal that similar concerns over the computer field's sociotechnical schisms can be traced all the way back

[217] This introductory narrative is inspired by Mulder's own accounts of how he came to get involved in a variety of curricular reform movements in the mid-1970s (Mulder, 1977, p. 70; Jones and Mulder, 1984, p. 24). I have taken an educated guess at both the year of this event and setting in which it unfolds. My gendering of the interviewee as male in this passage also reflects an unfortunate historical reality, namely that most of the candidates for such positions would have been men.

to the 1950s, Mulder's comments suggest that the divide between the hardware- and software-oriented factions had become particularly acute in the educational sphere. And as a rising leader in the engineering profession, Mulder was eager to probe the cause of this problem and begin working toward a solution.

Recognizing that the relevance of this issue extended well beyond the bounds of his own company, Mulder took his concerns to the IEEE Computer Society. He quickly found the sympathetic ear of C. V. Ramamoorthy, who at the time was serving as chair of the group's newly-formed Education Committee (Mulder, 1977, p. 70). As a result of their exchange, Mulder soon found himself serving as the chair of a new Model Curricula Subcommittee. Mulder's timing was right. The Computer Society's interests and activities in the educational arena were beginning to gain momentum, especially as the group started to pick up where the COSINE Committee had left off. Yet in contrast to COSINE – which in the early 1970s had promoted the development of computer engineering programs and options within electrical engineering – Mulder and the rest of the Computer Society's new school of educational reformers quickly adopted the boundary-spanning phrase "computer science and engineering" (CSE) as they worked to improve the training of future generations of computer-oriented professionals.

The Computer Society's embrace of the term "computer science and engineering" might appear a somewhat surprising turn of events, especially given the significant fragmentation of computer science and computer engineering education through the 1960s and into the early 1970s. On the other hand, the previous chapter hinted at some of the unifying trends that were sweeping through the field beginning in the 1970s. The scope and identity of the Computer Society, for example, expanded to encompass an ever-wider swath of "computer science and engineering." In fact, it was increasingly rare to find the Computer Society framed as primarily a "computer engineering" or even "engineering-oriented" organization, especially as the Society's publications and activities gradually expanded to cover "much of the total hardware, software, and applications spectrum" (Smith, 1977a).

The present chapter therefore sheds additional light on how the emergence of the "CSE movement" was not a coincidental development, but rather part of a larger array of mutually reinforcing processes and forces, many of which also played a role in the parallel expansion of the Computer Society's membership and scope. Even more importantly, my account reveals the extent to which the efforts of Mulder and other CSE reformers was both inspired and constrained

by the sociotechnical milieu in which their work was situated, just as the Computer Society was but one node in a larger system of professional organizations. To put it another way, these educational reformers faced what institutional theorists call a pre-existing "organizational field" that on the whole tended to preference conservatism, standardization, and incrementalism over the more fundamental types of reforms that the Computer Society's Education Committee was ostensibly promoting.[218] In fact, Mulder forcefully hinted at this theme when he later adopted the metaphor of the "tar pit" to describe the challenges that he and others faced as they worked to develop curricula that "mesh computer science and engineering" (Mulder, 1975, p. 28).

Below I document a variety of efforts to achieve this meshing, beginning with the Computer Society's promotion of CSE from the mid-1970s to mid-1980s, and culminating with the rise of the "Computing as a Discipline" movement in the late 1980s and early 1990s. This chapter also aims to provide a much richer description of the "tar pit" context in which these reforms developed. More specifically, my analysis places particular emphasis on a series of foundational "axes of similarity/difference" on which ongoing debates over the professional and disciplinary boundaries of the computing field were frequently constructed during this time period. To be sure, prior chapters have already shed some light on this theme, especially through my analysis of the evolving relationships between hardware and software, and science and engineering. Further, my discussions about the persistent tensions between fragmentation and unification in the various fields and subfield of computing – as aptly captured by the two-faced Janus metaphor – provides additional framing for this chapter. Yet the sections that follow look even more closely at how the privileging of "core" versus "peripheral" concepts and concerns led various groups and authors to very different understandings of what it means to adopt or privilege various sociotechnical identity markers, whether it be computer scientist or computer engineer, systems engineer or software engineer, or others.

In contrast to prior chapters, the sections that follow also place somewhat greater emphasis on yet another important axis of similarity/difference, namely that of discipline-profession. As a caveat, it is worth noting that concerns over disciplinarity and professionalism

[218] For more on the concept of "organizational field" – as well as a discussion about the tendency for such fields to promote standardization and homogeneity over variation and diversity – see DiMaggio and Powell (1983). As the authors summarize, "Once a set of organizations emerges as a field, a paradox arises: rational actors make their organizations increasingly similar as they try to change them" (p. 147). DiMaggio and Powell also offer valuable discussions about the many ways in which professionalization often encourages institutional isomorphism in a given organizational field (pp. 152-154).

were often noted in prior chapters, especially given the extent to which this particular distinction can often be mapped onto another axis, namely that of science and engineering. Yet the present analysis brings into further relief some of the key tensions between disciplinary and professional outlooks or perspectives, tensions that became increasingly evident in the 1970s and 1980s. More specifically, I document the difficulties and instabilities that emerged from the fact that the domain of computers and computing has long been partially shared by engineering – which is frequently viewed as a profession first and a discipline second – and computer science, which since its inception has primarily been conceived as an independent academic discipline.

As further background, the rich body of scholarship reviewed in the introductory chapter of this dissertation revealed contrasting perspectives on the organization of discipline and professions. More specifically, Abbott has promoted the "settlement" metaphor to describe the relatively loose and intertwined claims of disciplines, while he uses the term "jurisdiction" to frame the more rigid and entrenched character of the sociotechnical boundaries between professions. Further, many studies of professions tend to place primary emphasis on worksites and working practices, while studies of disciplines often focus on the academic context and the development of theory and abstract knowledge.

Hence, this chapter helps reveal how ongoing efforts to unify or integrate computer science and engineering challenged some deeply entrenched boundaries, not only between hardware and software, or even science and engineering, but also between discipline and profession. The success of these efforts therefore rested in large part on their ability to both work against a pre-existing and deeply entrenched organizational field and bring into a stable state of alignment a number of major axes of similarity/difference. Further, my analysis speaks to the importance of discourse in the construction, maintenance, and blurring of professional and disciplinary boundaries. In fact, the present chapter provides further support for the claim that disciplines and professions can be viewed as heterogeneous ensembles that are constructed out of diverse sociotechnical elements, ranging from discursive markers and abstract bodies of knowledge to technological artifacts and institutional infrastructures.

In light of this introduction, is it any wonder that the subject(s) in question appear persistently elusive and unstable? Yet no matter how messy the tar pit, my goal for this chapter is to develop a reasonable likeness of this complex and dynamic field, including its ongoing

evolution and development. And by focusing on the academic context in recent decades, this chapter also begins to point toward opportunities for critically engaged intervention and reform.

Claiming CSE: The Computer Society Makes Moves in Education

At the annual Spring Joint Computer Conference (SJCC) in 1971, the Computer Society's Education Committee was established in ad hoc form ("Computer Society Starts," 1971). The Committee achieved full standing committee status later in the same year ("Education Committee Added," 1971).[219] These were important developments, as they suggested that the Society was getting more serious about its role in the educational arena. And as the group was gaining its initial footing and establishing a tentative agenda, the last of the COSINE Committee reports graced the pages of *Computer* in 1972 and 1973, reflecting the reasonably cordial relationship between the Computer Society and this alternate body of educational reformers (Denning et al., 1972; Booth et al., 1973). When the COSINE Committee disbanded around 1972, commentators such as Martha Sloan raised concerns about the ability of the Computer Society to pick up the torch of curricular development and reform (Sloan, 1974). Yet by the mid-1970s such concerns were beginning to look overstated, especially given the 1975 publication of one of the first major reports of the Computer Society's Education Committee on "A Course of Study in Computer Hardware Architecture" (Rossman et al., 1975).[220]

Even more importantly, Mulder's prodding helped lead to the establishment of the Model Curricula Subcommittee in mid-1974 a branch of the Computer Society's Education Committee (Mulder, 1977, p. 70). Mulder himself served as the first chair of the group, and former COSINE contributors such as David Robinson and Martha Sloan joined as members.[221] Mulder and the rest of the Subcommittee quickly went to work on the development of new model curricula for four-year undergraduate degree programs in the area they suggestively dubbed "Computer Science and Engineering." Their first progress report was published in early 1975, and the final, full draft of the curriculum was completed in late 1976 and disseminated more widely in 1977 through derivative and supplemental publications. The scale of the undertaking was impressive.

[219] In comparison, evidence suggests that the ACM's Education Committee was formed and active by at least 1960, with Louis Fein serving as one of the group's first chairmen (Huskey, 1960b). By 1970, the relative maturity of the Education Committee was clearly reflected in its structure, which included a total of four special interest groups (SIGs) and six sub-committees (Carlson, 1970b).

[220] In fact, this six-member group included C. Gordon Bell, who had been extensively involved with the COSINE Committee in the early 1970s.

[221] Robinson was a member of the COSINE Task Force on Minicomputers (also known as Task Force VII). Sloan's work in and around the COSINE Committee are documented in significant detail in the preceding chapter.

Producing the final version of the report involved roughly two and one-half years of effort, 20 primary authors, 15 solicited contributors, 19 reviewers, approximately 20,000 man-hours of work, and more than $20,000 of Computer Society funds (Mulder, 1977, p. 70). In recognition of this fact, Computer Society President Merlin G. Smith bestowed upon the twelve main members of the Subcommittee a "Group Special Award" in late 1977 (Smith, 1977b). Contrary to Sloan's earlier concerns, the Subcommittee had carried forward the torch of curricular development and reform with an impressive level of ambition and enthusiasm.

Yet it is worth taking a closer look at the development of the group's recommendations, especially to highlight some of the major themes – and tensions – that surfaced in their work. In fact, many of these themes were evident in the group's first presentations, delivered at the Spring 1975 COMPCON meeting (Mulder et al., 1975). As background framing, Mulder and his co-authors articulated their concerns about the adequacy of computer-oriented degree programs:

> The voiced opinion from the computer industry is that the academic community is failing to provide the blend and the depth of computer-oriented instruction necessary to allow these new hires to be productive without considerable additional training. Students voice the opinion that they were not properly prepared to meet the demanding challenges of the computer industry (p. 33).

As this statement reveals, many students in computer-oriented degree programs and their prospective employers were no longer satisfied with the historical status quo, in which undergraduate education in fundamentals was supplemented by considerable amounts of formal and informal training in industry. And discontent with this educational model was only exacerbated by perpetual increases in both the technical complexity of computer technology and the rate of technological change, which made the provision of training by industry ever-more extensive, time-consuming, and costly. Just how did this mismatch between the educational sector and the needs of the computer industry come about?

Working toward an explanation, the authors noted the emergence of two distinct educational camps, one that preferred the development of degree programs grounded in science and theory, and another that privileged programs that were more pragmatic and engineering-oriented. More specifically, the report noted the long-standing tendency for Computer Science Departments to focus on abstracts and theory while neglecting subjects related to hardware, hardware/software interfaces, and systems. Conversely, they chided Electrical Engineering

Departments for being "slow to extend their programs beyond the hardware or electronic aspects of computer systems" (p. 33). As documented in preceding chapters, such comments echoed critiques that had been circulating since at least the mid-1960s.

Yet as this preliminary report made clear, this new group of reformers largely followed their predecessors when they placed primary blame on computer scientists – rather than engineers – for the inadequate training of computer professionals. In fact, a 1968 COSINE conference paper by C. L. Coates on "University Education in Computer Engineering" was particularly influential on the group's position.[222] In a section suggestively labeled "The Problem," the authors followed Coates rather directly when they noted that computer science was hamstrung by two "fundamental limitations," namely the arts and science background of most computer science faculty and the institutional location of computer science departments with arts and science colleges. In light of these factors, they argued that computer science departments were ill-suited to provide the more practical and engineering-oriented flavor of education that the subcommittee favored – and that industry was supposedly clamoring for.

Much of the remainder of this same report was dedicated to outlining the subcommittee's platform for reform. Yet unlike Coates – who was instrumental in shifting the agenda of the COSINE Committee away from the development of the "computer sciences in electrical engineering" and toward degree programs and options in "computer engineering" – the group adopted a more Janus-faced position that was focused on the domain they called "computer science and engineering." As they explained:

> It is the conclusion of this committee and others that have preceded it that we must recognize that computer science education and computer engineering education are not the same and that there is a need for both. The solution may well be the definition of model curricula that are interdisciplinary in nature with more emphasis placed on computer engineering (p. 33).

On the one hand, computer science and computer engineering education were marked by the authors as "not the same." On the other hand, their use of the phrase "computer science and engineering" suggested the need for educational programs in which the two domains were somehow blended or merged to form a single and more coherent curriculum. While this vision

[222] This particular interim report also featured a list of nine other COSINE Committee reports. Yet interestingly enough, the COSINE Committee's 1967 report on *Computer Sciences in Electrical Engineering* was omitted.

looked like throwback to the mid-1960s efforts of Zadeh and other engineers to embrace the term "computer science," it was also clear that the agenda of this new committee was unique, in no small part due to the changing sociotechnical context in which they worked.

Fleshing out what an interdisciplinary curriculum in "Computer Science and Engineering" might look like, the group explained that such academic programs should cover three intersecting areas, namely hardware systems technology, software systems technology, and processors/logic technology (p. 34). The report also argued that such programs needed to place "more emphasis on computer engineering, proper emphasis on computer science, and a flexible structuring of the curricula" (p. 34). The subcommittee's preference for the term "technology" and phrase "computer engineering" looked like a corrective to the dominant mode of computer science education, which tended to emphasize theory, programming, and software. Further, their strategic and guarded use of the term "computer science" looked like an attempt by these engineers to selectively and strategically claim portions of this adjoining disciplinary domain. In fact, the authors followed Coates by arguing that computer science departments were not amenable to the development of programs in "computer science and engineering," especially given their association with faculties and colleges of "arts and science."[223] And elsewhere the report reiterated that education in the area of "computer science and engineering" was "best provided in the domain of engineering (i.e., Electrical or Computer Engineering)" (p. 34).

Given this overview, one might also wonder how the Subcommittee came to couch their work under the aegis of "Computer Science and Engineering." In fact, this particular phrase was never promoted in earlier COSINE Committee reports, and it appeared only occasionally in professional publications through the early and mid-1970s, even as the Computer Society expanded its purview beyond computer engineering and into other areas. The Model Curricula Subcommittee also failed to offer an explanation for its choice of words. My own research suggests that the aforementioned Anthony Oettinger was one of the first individuals to widely promote this particular phrase. As documented in Chapter 5, Oettinger served from 1966 to 1968 as President of the ACM, and during this time he expressed significant ambivalence about the status of computer science as a discipline. In a 1967 commentary, he explicitly claimed that the term "computer science" was a misnomer, and in as early as 1966 he was using the alternate

[223] In fact, Coates' original 1968 comments about the affiliation of computer science with the arts and sciences appeared in near-verbatim form in this committee report.

phrase "computer science and engineering" (Oettinger, 1966b, p. 839). A 1968 letter by Oettinger also referenced "Computer Science and Engineering," with capitalization used to suggest that this was indeed a recognized and distinct field of activity (Oettinger, 1968b, p. 293).

Even more importantly, Oettinger spearheaded the establishment of a Computer Science and Engineering Board (CSEB) at the National Academy of Sciences in 1968, amidst growing concerns about the relatively low visibility and influence of computer professionals and their interests in Washington ("Computer Science and Engineering Board," 1968; Titus, 1968). In addition to acting in an advisory capacity to both the Academy and the government on a wide range of computer-related issues, the group also promoted the interests of the computer field, especially in areas such as funding for research. Yet in spite of both Oettinger's efforts and the formation of the CSEB, the application of the term "computer science and engineering" remained rather infrequent and scattered through the early 1970s.[224] In fact, the CSEB itself was disbanded by late 1973, which might have been the demise of the phrase had the Model Curricula Subcommittee not embraced it shortly thereafter (Ralston, 1973b, p. 725).

But even as "CSE" was being brought back to life by this other group of actors, numerous questions remained about the extent to which these engineers were claiming computer science, or some portion thereof. Through a series of additional papers and presentations, the group gradually refined its vision and provided a more detailed picture of what educational programs in "Computer Science and Engineering" might look like. And over time, the work of the subcommittee started to look more like a discipline-building project, albeit in ways quite distinct from Oettinger's prior efforts. In the sections that follow, I review a number of the group's subsequent publications. My analysis brings into further relief the group's evolving agenda, as well as the numerous barriers that stood in the way of realizing educational programs bearing the mark of CSE, much less an entire discipline.

Bridging the Tar Pit?: Toward a Curriculum in Computer Science and Engineering

Another snapshot of the Education Committee's activities appeared in the December of 1975 issue of *Computer*, which was topically dedicated to "Computer Education." In a lead article, Mulder reviewed the ongoing work of the Model Curricula Subcommittee. And while the

[224] As documented in the preceding chapter, Lotfi Zadeh also adopted this particular phrase when he noted in a 1971 paper that "electrical engineering has a special responsibility to train its students in both the basic and applied aspects of computer science and engineering" (1971, p. 153).

general message of the report followed prior documents closely – at times in verbatim or near-verbatim form – Mulder embellished his account with an evocative analogy. As the author explained, prior efforts to develop "model curricula that mesh computer science and engineering" were akin to the "tar pits" of prehistoric lore, where great beasts engaged in mortal struggle (Mulder, 1975, p. 28). Further, he explained that the committee's work toward a new set of model curricula for Computer Science and Engineering programs represented an effort to "bridge the tar pit" (p. 28). Emphasizing the value of a more integrated or unified approach to the education of computer professionals, Mulder explained that the work of the subcommittee was "the first effort to bridge the gap between computer science and computer engineering. And *this* is the difference between past and current efforts" (p. 31).

On the one hand, individuals such as Zadeh and some of the early COSINE reports can be credited with taking some tentative first steps toward this type of bridging. On the other hand, both Mulder's tar pit analogy and the committee's use of the "computer science and engineering" moniker suggested that this new reform movement was more explicitly concerned with working toward some sort of curricular unification that spanned these two sociotechnical domains. In fact, the author noted that the model curricula project was focused on an "integration" of the "hardware and software disciplines," and he added that the subject areas identified by the subcommittee represented "the domain of computer science and engineering" (p. 29). Such statements reveal the extent to which the development of curricula can quickly shade into a form of discipline-building. And indeed, the subject areas and courses identified in Mulder's article covered a broad array of topics that were at least partially germane to the work of computer engineers and computer scientists, ranging from digital logic and computer organization to operating systems, software engineering, and computing theory.[225] The author also explicitly declared that "a merging of computer science and computer engineering disciplines is both desirable and possible" (p. 31).

Yet in spite of Mulder's ambitious remarks, the uptake of this early draft of the subcommittee's model curricula was probably limited, especially given that it was short on detail and marked as "interim." Further, building metaphorical bridges on paper was far easier than

[225] The minimal "core" and more extensive "typical" curricula presented in the report also covered a wide range of material, albeit with greater emphasis on boundary-spanning topics such as computer organization, operating systems, and software engineering. The peripheries of hardware and theory received somewhat less attention.

realizing a "unified approach to education in computer science and engineering" in the "tar-pit" milieu of actual academic institutions. Two additional papers published during this same time period provide additional insights regarding the historical context for – as well as the Janus-faced character of – the Computer Science and Engineering movement. The first of these was published alongside Mulder's article in the same 1975 issue of *Computer,* and it presented the results of a survey of electrical engineering and computer science departments (Sloan, 1975). Spearheaded by Model Curricula Subcommittee member Martha Sloan, this project was described as an update to the 1972 COSINE survey, which I reviewed in Chapter 5.

Given that it was mailed to 222 electrical engineering and 95 graduate-level computer science departments, the survey revealed the extent to which the latter discipline was well-established at a large numbers of schools. And the results of the survey provided other important insights about the departmental and curricular boundaries that had grown up in the realm of computer-oriented education. Sloan's data showed, for example, that the vast majority of the 160 responding departments carried the name "Electrical Engineering" or "Computer Science," while only three used the combined "Electrical Engineering and Computer Science" (p. 36). The author also noted the tendency for many departments to fall rather predictably on one side or the other of the hardware-software divide: "The distinction between CS departments predominating in software and EE departments predominating in hardware is well established, at least in the aggregate" (p. 40). Along similar lines, Sloan reported that joint faculty appointments and joint course offerings between the two types of departments were rare, and she added that "[E]ven when a department does teach a course traditionally belonging to the other department, it colors the course with its own orientation" (p. 40). These deeply entrenched curricular and departmental bifurcations help reveal the salience of Mulder's "tar pit" analogy.

With regard to the field of computer engineering more specifically, Sloan indicated that only 3 of the 160 responding departments carried the name "Electrical and Computer Engineering," although a much more impressive 51% of all responding electrical engineering departments offered some type of CS or CE degree program or option (p. 36). As these data reveal, computer engineering remained largely positioned as a branch or "dimension" of electrical engineering education, rather than as a distinct discipline unto itself. In addition, Sloan's analysis suggested that this situation was not likely to change, at least in the near term. She noted, for example, that the growth of computer science and computer engineering courses

and options was slowing within EE departments, although she added many opportunities remained to "consolidate and improve curricula" for computer science and/or computer engineering programs (p. 40).

And indeed, the term "consolidation" appeared an apt characterization of the curricular reform efforts that were underway around this time. Yet her survey also revealed major barriers to the development of academic programs in the area of "computer science and engineering," including the persistent division of computer science and computer engineering programs, courses, and faculties, which were often situated in entirely different departments and even colleges. By contrast, computer science departments and graduate programs – both of which serve as key markers for disciplinary identity and crucial sites for disciplinary development in the American academic context – had proliferated from the mid-1960s onward. Truly unifying the disciplines of computer science and computer engineering therefore appeared a formidable task, especially given the various schisms that had grown up between these two fields in the educational arena. On the other hand, those at the forefront of the CSE movement appeared increasingly interested in developing their own flavor of computer science and engineering education *within* schools and departments of engineering, and with little concern for the ultimate role or fate of computer science departments and programs.

Many closely related themes were evident in a 1976 review article titled "Computer Science and Engineering Education," authored by Education Committee chair and Model Curriculum Subcommittee member C. V. Ramamoorthy (Ramamoorthy, 1976). Published in a special 25th anniversary issue of the *IEEE Transactions on Computers*, Ramamoorthy's article started by briefly chronicling "the evolution of CSE education," from the development of logic design and programming courses at a handful of pioneering institutions in the 1950s to the rise of Computer Science departments and programs in the 1960s. And with regard to more contemporary matters, Ramamoorthy acknowledged the many different institutional realizations of "CSE education," including through separate Computer Science and Electrical Engineering Departments (Illinois at Urbana, Northwestern, Stanford, and Texas at Austin), combined Departments of Computer Science and Electrical Engineering (MIT and California at Berkeley), and the relatively recent emergence of Electrical and Computer Engineering Departments (University of Michigan and University of Wisconsin at Madison). Turning to the historical development of curricula, the author reviewed a series of key milestones, including the ACM's

Curriculum 68, various COSINE Committee reports, and the ongoing work of the Computer Society. He also discussed a number of major trends and key issues with regard to CSE education, including accreditation and certification efforts, the development of laboratory facilities, and debates over the value of theory versus practice in computer-oriented degree programs.

On the one hand, the author's wide-ranging review revealed a desire among reformers to strategically frame "Computer Science and Engineering" as a single domain, in spite of the extensive historical segregation of computer science and computer engineering in the academic context. On the other hand, Ramamoorthy clearly delineated the computer field's two sociotechnical faces. He explained, for example, that "the computer scientist is interested in the theory and science of computation and programming," while "the computer engineer is interested in the specification, design, implementation, and utilization (operation) of data processing systems including both hardware and software" (pp. 1200-1201). Even more suggestively, the author noted that "[t]he computer engineer (including the software engineer) uses the principles of computer science and/or electrical engineering in specifying, designing, implementing, and utilizing computer systems for specific applications" (p. 1201).

As suggested by these remarks, engineers such as Ramamoorthy were inclined to frame computer engineers as claiming an ever wider swath of sociotechnical territory, including relevant portions of software, software engineering, and even computer science. And even though he adopted the phrase "computer science and engineering," his comments also perpetuated the idea that computer scientists and computer engineers were associated with distinct disciplinary and professional identities, even if he admitted that the former might draw on the theoretical knowledge and principles developed by the latter. In fact, the reasons for holding onto these distinct identities were both historic and pragmatic. That is, the term "computer engineer" emerged in the early 1950s and was used with increasing frequency through the 1950s and beyond, while "computer scientist" was coined in the late 1950s and widely applied in the 1960s. These monikers had seeped deeply into the discursive infrastructure of the computer field, and replacing them would likely require a more compelling alternative than an awkward and seemingly paradoxical phrase such as "computer scientist and engineer."

In summary, the three articles reviewed here shed important light on the fundamental tensions that came with calling for the establishment of integrated educational programs in the

area of "Computer Science and Engineering." Even more specifically, I have documented the Janus-faced character of this movement, where various reformers framed computer science and computer engineering as distinct fields, albeit somehow united under the aegis of CSE. Further, my analysis hints at the extent to which the discursive construction of CSE as a unified or integrated domain was significantly in conflict with the persistent structural segregation of computer science and computer engineering courses, programs, and faculties in separate academic departments and even colleges. And Ramamoorthy's comments in particular reveal the deeply entrenched distinction between the respective identities of computer scientists and computer engineers.

While the members of the Model Curricula Subcommittee were likely cognizant of at least some of these issues, the following sections reveal that the group's continued optimism regarding the development of educational programs dedicated to Computer Science and Engineering. My analysis also brings into further relief the extent to which these reformers were concerned with encouraging and overseeing the development of CSE *within* colleges and departments of engineering.

A(n Engineer's) Curriculum in Computer Science and Engineering

Following a long series of interim reports, conference presentations, workshops, and revisions, the final version of *A Curriculum in Computer Science and Engineering* was dubbed "Revision 1" and initially published in early 1977 (Education Committee, 1977). As background for this lengthy report, the authors explained that their recommendations were designed to provide sufficient curricular breadth and depth, bridge the gap between hardware and software, and be suitable for implementation in a variety of institutional contexts (pp. 1-2). The organization of the curriculum had also evolved considerably since the group's first reports, and the authors ultimately settled on recommending four main subject areas for coursework, namely digital logic, computer organization and architecture, software engineering, and theory of computing.[226] The report also featured detailed outlines, instructional objectives, and lists of

[226] These four subject areas were the result of a gradual evolution that started with the aforementioned division of CSE into hardware systems technology, software systems technology, and processor/logic technology. In late 1975, an intermediate report by Mulder revised this spectrum to include hardware systems, software systems, and computing theory (Mulder, 1975, p. 30). This report also added a new, parallel set of categories that included digital logic, computer organization, operating systems and

reference materials for a total of 22 courses and 6 laboratories. A smaller subset of courses and labs were identified as a minimal "core curriculum" for schools in the early stages of developing and/or implementing CSE programs.

Acknowledging the ambitious character of their proposed curriculum, the authors explained that the recommendations represented "the global continuum of computer science and engineering" (p. 3). Yet despite such discipline-building rhetoric, the authors were strategic in what they included, as well as what they left out. For example, only four courses were recommended in the subject area marked "Theory of Computing," and these tended to emphasize the theoretical foundations of analysis and design rather than the more abstract and theoretical topics that were typically associated with computer science education.[227] In addition, only a relatively small portion of the Theory of Computing subject area fell within the prescribed core curriculum, which seemed to reflect the committee's aforementioned commitment to placing a "proper emphasis on computer science." The recommended coverage of the hardware-oriented "Digital Logic" area was also comparatively minimal. In fact, this part of the curriculum featured just four classes, and only a fraction of this content was included in the core curriculum. Both the Software Engineering and Computer Organization and Architecture subject areas, on the other hand, dominated the report generally and the core curriculum specifically. These emergent boundary fields were therefore implicitly framed as unifying cornerstones for CSE education, as well as for CSE writ large.

Yet the development of the model curriculum was not happening in isolation, and it is worth noting that the group's recommendations paralleled larger trends in the computer field. As noted in the preceding chapter, for example, the impressive expansion of the Computer Society's size and scope in the 1970s was in part linked to the group's strategic movement into various subfields on the boundaries of computer science and computer engineering. In fact, many of the group's new technical committees, workshops, conferences, and publications were focused on precisely those domains that were at the core of the new Model Curriculum, such as Computer Architecture, Operating Systems, and Software Engineering. The shifting boundaries of both the

---

software engineering, and theory of computing. The original categories were eventually dropped and replaced by the new category names as listed here.

[227] In fact, the authors of the report noted that the area of "Theoretical Computer Science and Engineering" had historically developed in a two-pronged manner, with "one concerned with hardware design (automata theory) and the other with programming language and compiler design (formal languages)" (p. 64).

Computer Society and the proposed model curriculum therefore reflected and reinforced one another, as well as the technological state of the art. Further, the expanding claims of engineers in these boundary-spanning subfields helped legitimize their adoption of the phrase "computer science," as well as their selective borrowing of subjects and topics that had historically been associated and/or shared with computer science.

This same report also hinted at the extent to which colleges and departments of engineering remained the authors' preferred site for the development of CSE programs. To begin with, the minimal core curriculum appeared well-suited for implementation as a new option within existing engineering programs. And even more importantly, an Appendix tacked on at the end of the report provided an outline for a four-year "Electrical and Computer Engineering Curriculum" that was designed to meet accreditation guidelines that had been established by the Engineers Council for Professional Development (p. 98). While the authors offered little in the way of explanation regarding this supplemental documentation, it revealed this group's preference for the development of "Computer Science and Engineering" education within colleges and departments of engineering. This appendix also hinted at the rising importance of accreditation, a point I discuss in more detail below.

The work of the Subcommittee received even wider distribution in late 1977 through the publication of a special issue of *Computer* on the topic of "Computer Science and Engineering Education." In addition to a series of articles on the proposed curriculum and related topics, this same issue also included an article that compared and contrasted the Computer Society's Model Curriculum with another set of curricular recommendations that were being developed by the ACM's C^3S, and which were first published in draft form in mid-1977 (Engel, 1977). Author Gerald Engel – who at the time maintained close ties with the ACM, including through his role as liaison between the ACM's Education Board and the Computer Society's Education Committee – explained that the two reports reflected "differences in background and philosophy between the computer engineer and the computer scientist trained in the liberal arts tradition" (p. 121).

Yet rather than belaboring this point, Engel instead emphasized the overlap between the two sets of recommendations, especially in core subject areas such as software engineering and "programming design." Noting the potential for closer cooperation between the Model Curriculum Subcommittee and C^3S, Engel concluded on an optimistic note: "One day, perhaps,

computer science and computer engineering will no longer exist as separate entities, but instead as a single program representing options of a common core of fundamental material. This common core is the essence of our profession" (p. 123). While Engel's claim that the so-called "core" represented the unity of the computer field, the continued development of separate curricular recommendations by the ACM and Computer Society revealed the persistence of the field's two sociotechnical faces, as well as the continued importance of "peripheral" domains of technology and knowledge for the various actors and groups in this story.

However, various committees and initiatives seemed to bring the educational activities of these two professional societies into closer alignment, especially in the late 1970s and through the 1980s. This time period was also marked by expanded efforts to establish and promote the discipline of "Computer Science and Engineering." These trends hinted at the tentative emergence of a more unified discipline. Yet the ultimate success of this movement was far from assured, especially given that it required the assembly of rather complex – and fragile – ensemble of heterogeneous sociotechnical elements. In the following sections I continue to trace out this complex tangle of interests and forces, both within the Computer Society's Education Committee and beyond.

Supporting Curricular Reform in Electrical Engineering Education

While the mid-1970s ambitions of the Model Curricula Subcommittee were impressive, other groups and events provided the group with both direct and indirect forms of support. For example, the participants in the Digital Systems Education (or DISE) Project spearheaded a variety of activities that were synergistic with both the work of the Subcommittee and the earlier efforts of the COSINE Committee.[228] After meeting for the first time in 1974, this inter-university and inter-industry working group secured three years of NSF funding for the development of educational materials in the "digital systems" area (Cain and Hoelzeman, 1977). As explained in one of the group's reports, the project was largely prompted by the difficulties that educators faced as they tried to keep pace with rapid technological and theoretical change (Cain and Hoelzeman, 1977, p. 145). In support of their goals, the group established a newsletter and a repository for instructional materials, and a variety of DISE task force groups led the

[228] Taylor Booth was the only former COSINE member to participate in the DISE Project. J. T. Cain and Ronald Hoelzeman, on the other hand, were members of both the Model Curriculum Subcommittee and DISE Committee.

development and collection of materials in areas ranging from digital systems to software engineering (Cain, 1975). Another DISE task force concerned itself with improving communication and cooperation between universities and industry (Cain, 1975). The group also organized a Workshop on Microprocessors and Education in 1976, and select papers from this event were published in the January 1977 issue of *Computer*.

On the one hand, the DISE Project was significant because it continued one of the major areas of reform led by the COSINE Committee, namely the reorganization of electrical engineering education around digital systems and related topics. DISE efforts to promote the development of software engineering education also fit into this larger movement, and one DISE report even framed software engineering as a "'new' area or approach within the digital systems area" (Cain, 1975, p. 15). On the other hand, the overall impact of DISE was likely limited, especially given that the group appears to have been disbanded after NSF funding ran out in 1977. Further, the concept of "Digital Systems Education" as promoted by DISE was largely a subset of larger, parallel efforts to develop educational programs in computer science and engineering. In fact, Ramamoorthy noted in his 1976 review of computer science and engineering education that the efforts of the DISE Committee to collect and develop educational materials were in part being guided by the curricular recommendations being developed by the Computer Society's Model Curricula Subcommittee (Ramamoorthy, 1976, p. 1202).

Two additional branches of the Computer Society's Education Committee provided other types of support for the work of the Model Curricula Subcommittee. The Regional HELP Subcommittee, to begin with, was formed to provide assistance to those schools wishing to establish programs or departments dedicated to computer science and/or computer engineering (Ghosh, et al., 1975). This group was therefore designed to overcome the challenges of implementation and reform at actual institutions, a problem that the COSINE Committee had tried to address with its less-than-successful site visit program.[229] The specific activities of this group were not widely publicized, which makes it difficult to document its overall impact and agenda. However, a 1976 report of the group's activities revealed that its members were working closely with a handful of schools, with the goal of developing computer engineering programs that were in agreement with the recommendations of the Model Curricula Subcommittee (Rine,

[229] By 1979 the Regional HELP subcommittee was renamed "Curriculum Implementation and Assistance," and by 1982 it was titled "Curriculum Assistance." Additional research is needed to determine the overall role and impact of this evolving group.

et al., 1976). The authors also noted that a series of regional HELP Workshops were being planned, where interested faculty could begin to think about and plan for the implementation of the Computer Society's model curricula on their own campuses. And finally, this report identified a number of programs that might serve as "models" for CSE education, and the list featured a number of well-known institutions such as UC-Berkeley, Stanford, Illinois, Carnegie-Mellon, and MIT (p. 210).

The Subcommittee on Coordination, on the other hand, was established to coordinate matters of mutual interest between the Computer Society's Education Committee and other groups and organizations (Salisbury, Snyder, and Smith, 1975). In fact, the ACM's Standing Committee on Curriculum in Computer Sciences (C^3S) was the group's sole initial focus (p. 41). Noting that the "common educational concerns between the Computer Society and the ACM mirror in large measure the overlapping areas of interest of the two organizations," the authors a 1975 subcommittee report put forward the "oversimplified" view that the ACM tended to focus on the development of the "computer science" curriculum, while the Computer Society was concerned with "computer engineering" (p. 41). This same report also noted that one of the major functions of the subcommittee was to "eliminate duplication of effort in those areas where the two curricula can properly follow a common core" (p. 41). In order to work toward this goal, "observers" from the ACM and Computer Society were selected to attend relevant meetings in each counterpart organization. The report also noted the close relation of the Coordination group with both the Model Curriculum and Survey Subcommittees.

As suggested by this overview, the Subcommittee on Coordination was framed as an "interface" and "channel of communication," both within the Computer Society and beyond. It therefore acted as a sociotechnical mediator, in a manner akin to many of the other ACM SIGs and Computer Society TCs that maintained overlapping interests. Further, it is worth noting that the relation of these two professional societies on educational manners was generally cooperative around this time, especially given that individual institutions rather than professional societies were the primary sites for battles over discipline building and curricular reform. In fact, the 1978 publication of *A Library List on Undergraduate Computer Science, Computer Engineering, and*

Information Systems – which was prepared by a joint committee of the ACM and the Computer Society – provides further evidence for this cordial relationship (Joint Committee, 1978).[230]

A series of Workshops on Computer Science and Engineering (CSE) Curricula provided additional support for the Education Committee and its various subcommittees. The first three of these events – which took place in 1976 and 1977 – were primarily concerned with the overall development and site-specific implementation of the Computer Society's Model Curriculum in CSE.[231] The third workshop was also noteworthy for its explicit concern with both bridging the gap between hardware and software and incorporating microprocessor technology in CSE education. The fourth and final workshop was organized in early 1978 and emphasized the adaptation of CSE materials for use by smaller institutions, community colleges, and introductory courses at large schools. The impact of these workshops was likely significant, given that each event attracted anywhere from 80 to 130 or more participants. Digests of papers were also published for the latter two events, further disseminating the content of these meetings.

While the various committees and events reviewed in this section provided various types of support for the development of CSE in the academic context, the term "computer science and engineering" was also being adopted for other purposes and projects, especially in the late 1970s and early 1980s. In fact, these projects were projecting the outward image of a full-blown discipline-building project, supported by an impressive cadre of actors and groups. Yet as my analysis reveals, these projects managed to only thinly veil the underlying sociotechnical tensions that came with them.

Disciplining CSE: Taxonomies, COSERS, and Encyclopedias, oh my!

As noted above, many of the publications and presentation of the Model Curricula Subcommittee indicated that the group's proposed curriculum was a sort of outline or map of the boundary-spanning discipline dubbed Computer Science and Engineering. This is perhaps not surprising, especially given the extent to which the building of curricula – not to mention associated departments and degree programs – is often deeply intertwined with the building of

[230] The overall organization of this extensive list of books and other reference materials was itself an act of boundary-work, in that it involved the development of a taxonomic organization scheme for computer science, computer engineering, and information systems. The report probably provided inspiration some of the subsequent taxonomy projects that I discuss in more detail below.

[231] My summary of these events is based primarily based on the remarks of Rine and Lee (1978) as a part of their introduction to the published proceedings of the fourth such workshop in 1978.

disciplines. Yet in the middle and late 1970s, at least three major projects got underway that were quite explicitly concerned with outlining the historical trajectory, contemporary contours, and future research horizons of the discipline ostensibly dubbed "computer science and engineering." These large and ambitious undertakings spanned multiple years, required significant financial backing, and involved large numbers of coordinators, authors, and reviewers. These three projects were also explicitly concerned with producing final reports that were accessible to those outside of the field, thereby reflecting growing concerns about the proper image and perceived legitimacy of the computer field and its branches, both in society generally and in communities of scientists, technologists, and policy-makers specifically.

Initiated by the American Federation of Information Processing Societies (AFIPS) in 1977 and published in 1980, the *Taxonomy of Computer Science and Engineering* report was the result of three years of work by a committee of eleven, along with seventy additional authors and reviewers, many of them well-known in the field (Ralston, 1980). The project also received financial support from the Institute of Computer Science and Technology of the National Bureau of Standards and the Air Force Office of Scientific Research and the Office of Naval Research. Committee chairman Mathematician, computer scientist, and former ACM President Anthony Ralston explained in a preface to the group's final report that the development of the taxonomy involved "a study of the structure of a discipline which appears to be unique among the sciences and almost unique among all disciplines" (p. v). Further outlining the motivations behind the project, Ralston added that the project was designed as a response to persistent misunderstandings about "what computer science and engineering is," especially by those outside of the field, and especially in light of its rapid growth and development (p. v). Among a variety of possible uses, the authors suggested that the taxonomy could serve as a reference for definitions or bibliographic sources, or as a guide for allocating research grants, classifying jobs, or organizing publications.

The taxonomy itself was organized as a "tree" with nine major branches, namely hardware, computer systems, data, software, mathematics of computing, theory of computation, methodologies, applications/techniques, and a residual category dubbed "computing milieux." As the authors explained, the organization of the taxonomy in this manner reflected the committee's view that "Hardware and Software are the 'pure' endpoints of a core computer science and technology continuum" (AFIPS Taxonomy Committee, 1980, p. 416). Yet as in

previous reports and commentaries, the taxonomy came to reflect various disciplinary and professional bifurcations, in spite of its stated objective of codifying the structure of "Computer Science and Engineering." It included, for example, a series of suggestive taxonomic descriptions of occupational titles and organizations, including the following:

> 9.6.1 Occupational Titles
> 9.6.1.1 Computer scientist
> [A person involved in computer research or advanced development.]
> 9.6.1.2 Computer engineer
> [A person involved in design or development of computer hardware.]
> …
> 9.6.2 Organizations
> 9.6.2.1 Technical Societies
> …
> 9.6.2.1.2 ACM
> [The Association for Computer Machinery whose members are mainly computer scientists and programmers.]
> …
> 9.6.2.1.4 IEEE Computer Society
> [The Institute of Electrical and Electronic Engineers Computer Society whose members are mainly computer engineers.]
>
> (AFIPS Taxonomy Committee, 1980, p. 11)

To begin with, these entries once again acknowledged the existence of distinct social and professional identities for computer scientists and computer engineers. Further, the accompanying organizational descriptions revealed the respective and persistent linking of these two identities with the field's two main professional societies, in spite of increasing overlap and coordination between the two groups.

In summary, the dominant image of the field's social and professional boundaries stood in marked tension with the authors' claim that Computer Science and Engineering could indeed be viewed as a single discipline. Yet in a discussion of philosophical and technical issues included at the end of the taxonomy, the authors acknowledged some of the tensions that came with their project. They noted, for instance, that CSE was a "new and rapidly changing field," thereby impeding the ability of this group to establish a high degree of "conceptual unity and stability" in the proposed taxonomy (p. 415). As a more specific example of these challenges, the

report acknowledged that significant controversy had erupted over the "Computer Systems" node, which reviewers had apparently criticized as a "misconceived hybrid" (p. 416). While certainly a suggestive turn of phrase, in the context of the present analysis it quite naturally leads to the question of whether "computer science and engineering" was itself a misconceived hybrid, especially for those uneasy about blurring the disciplinary and professional boundaries that traditionally separated science from engineering.

One finds similar tensions evident in COSERS, or the "Computer Science and Engineering Research Study," which was conceived in 1974 and launched in 1975 with financial support from the NSF (Arden, 1980, Preface). The COSERS steering committee featured thirteen well-known computer scientists and engineers, including university, government, and industry affiliates. The committee also included three members of the aforementioned AFIPS Taxonomy Committee.[232] According to Bruce Arden – who chaired the project while serving as the head of Princeton's Department of Electrical Engineering and Computer Science – the major objectives of the COSERS study involved identifying and describing the boundaries of research in the domain marked "computer science and engineering" (Preface). According to the group's final report, this description would also serve as an "operational definition" for the formative field. Revealing one of the main motivations for their ambitious undertaking, Arden noted in one his earlier reports that the development of such a definition "will have a salutary, self-organizing effect on this relatively new research area and … the resulting report will be useful for technical administrators in their task of research-support allocation" (Arden, 1976, p. 673). Such comments reveal some overlap between the goals of this project with both the AFIPS Taxonomy work and prior projects, including Oettinger's efforts to promote research in "Computer Science and Engineering," including through the establishment of the short-lived CSEB.

The final COSERS report – titled *What Can Be Automated: The Computer Science and Engineering Research Study* – was published in 1980 by The MIT Press and featured the work of eighty contributing authors. In most general terms, the report was organized around eight primary subject areas. In "decreasing order of longevity," these subjects included numerical computation, theory of computation, hardware systems, programming languages, artificial intelligence, operating systems, database systems, and software methodology. The group also

[232] The three overlapping members included Bernard Galler, Jean Sammet, and Stephen Yau. The two projects also involved a number of overlapping authors and reviewers.

described research in a diverse variety of "application" areas, ranging from algebraic computation and computational linguistics to computer applications in medicine and air traffic control. Yet in contrast to the AFIPS taxonomy, the COSERS project was somewhat more explicitly concerned with identifying and describing future directions for research. Following Mahoney, the COSERS participants were attempting to codify the disciplinary *agenda* of computer science and engineering, especially around the question "What can be automated?"[233] Yet the impressive breadth of the final report revealed that the actual research agenda was both wider in scope and more fragmented than suggested by any single, unifying question.

In his introductory remarks, Arden also acknowledged ongoing debates regarding the respective boundaries around the science and engineering aspects of computing, and he reviewed a series of justifications for maintaining this division. For example, he noted that computer scientists tended to focus on explanatory models and "understanding," while computer engineers were more concerned with applications and implementation. Arden countered, however, that common concerns with efficiency made it difficult to maintain the science-engineering distinction. The author also noted that the two fields were frequently divided according to their relative proximity to physical equipment: "In short, there is currently an operational difference between computer science and computer engineering, which corresponds roughly to how close interests are to the levels of physical implementation of algorithms, or the machine level" (Arden, 1980, p. 7). Yet he questioned this rationale as well, arguing that concerns about "the cost of algorithms at all levels" tended to blur this difference. Following another closely related theme, the report also summarized ongoing efforts to develop a succinct and universal definition of computer science, while noting both the inherent difficulties with such a task and the author's preference for an "operational" rather than "simple" definition.[234] As suggested by this overview,

[233] As Mahoney explains, "The agenda of a field consists of what its practitioners agree ought to be done, a consensus concerning the problems of the field, their order of importance or priority, the means of solving them, and perhaps most importantly, what constitutes a solution" (2000).

[234] Interestingly enough, Arden added: "Since computer engineering, no matter how it is distinguished from computer science, rests ultimately on the same definition, it has not generated independent candidates for definition" (Arden, 1980, p. 7). While perhaps overstated, this remark once again revealed the extent to which computer science and computer engineering had evolved in relation to one another. This comment also hints at the extent to which the development of computer engineering as a distinct field tended to involve implicit rather than explicit discipline-building activities, such as the ongoing development of curricula, programs, social identities, etc.

Arden was intent on framing "Computer Science and Engineering" as a single field, in spite of both its broad span and history of segmentation and fragmentation.

Various subsequent publications also perpetuated the image of a unified field, at least on the surface. For example, the *Encyclopedia of Computer Science* – which was first released in 1976 – was published in second edition form in 1983 under the modified title *Encyclopedia of Computer Science and Engineering* (Ralston and Meek, 1976; Ralston and Reilly, 1983).[235] This name change was not entirely surprising, given that the aforementioned Anthony Ralston served as an editor for both editions. As Ralston explained in a Preface to the second edition, "This change is both an attempt to describe the contents of this book more accurately and an explicit recognition of a greater emphasis on this edition than in the last on computer technology" (Ralston and Reilly, 1983, p. xi). In addition, Ralston noted that the major categories used to classify the articles in the encyclopedia largely corresponded to those presented in the AFIPS *Taxonomy*. Hence, these two projects were in part mutually reinforcing.

Yet a closer look at these tomes reveals a continued privileging of the computer science outlook. The first and second editions, for instance, included entries for "computer science," but none for "computer engineering" or even "computer science and engineering." However, the 1983 edition acknowledged that computer science maintained significant overlap with both mathematics and electrical engineering, and at one point even claimed that computer science "is also considered an *engineering science*," especially given the role of design-oriented activities in many phases of the field (p. 366). An appendix in the second edition also included a list of universities in the United States and Canada that offered the Ph.D. degree in computer science or closely related fields (pp. 1598-1600). Of 83 such schools, more than three-quarters offered Ph.D. degrees that were situated in departments or programs of computer science, computing science, or similar. On the other hand, just eleven of the listed programs and departments included the word "engineering" in their title, and these varied widely in naming. This appendix therefore put forward the image of computer science as a bona fide discipline, complete with well-established departments and programs that were steadily churning out doctorate degrees in computer science.

While Oettinger can be credited with first popularizing the term "computer science and engineering" and the Model Curricula Subcommittee later promoted the development of

[235] For a recent, article-length history of the *Encyclopedia of Computer Science*, see Ralston (2004).

educational programs bearing this moniker, the AFIPS, COSERS, and Encyclopedia projects reveal that the disciplinary development of CSE had entered a new phase. More specifically, these projects variously attempted to identify and in part codify the *structure* of the discipline, including via hierarchically ordered lists of various constituent categories, sub-categories, and subjects. In addition, these projects were framed as contributing to ongoing efforts to garner additional support for the field of computer science and engineering, with particular emphasis on improving how uninitiated outsiders understood the character and contours of the field. In fact, the COSERS project in particular placed significant emphasis on identifying a unifying *agenda* for the field, as well as a variety of more specific research horizons. These projects were strategic and political, as they promoted an outward image of unity in the domain marked computer science and engineering, despite the various internal schisms that divided the field.

My discussion of these projects also reveals that the discipline-building project of CSE had at least partially transcended the Computer Society and its cadre of educational reformers. In fact, these projects brought together a variety of actors who maintained close ties with the ACM and the Computer Society, computer science and computer engineering, and industry and the academy. On the other hand, the Computer Society's Education Committee largely retained control of the phrase "Computer Science and Engineering" for its own model curricula, and these recommendations tended to privilege the needs and perspectives of the many engineers and engineering educators who dominated the group. In the following sections, I document how this tension persisted through much of the 1980s, and then partially reversed in the late 1980s through the "Computing as a Discipline" movement. As my account makes plain, the shifting disciplinary landscape of the computer field was persistently suspended between the abstract ideals of discipline builders and the extant realities of pre-existing institutional structures, professional identities, divisions of labor, accreditation processes, discursive constructs, and technological developments. In the following sections I take a closer look at some of the currents and undercurrents that accompanied the emergence of "Computer Science and Engineering."

Managing Complexity: The Hybridization of Hardware and Software Engineering

While my analysis has hinted at the importance of technological change as a backdrop for the evolving professional and disciplinary landscape of computing, it is worth bringing this theme into further relief. More specifically, this section reveals the persistent blurring of not only

the software-hardware boundary, but also the various bodies of knowledge and techniques associated with these domains. In doing so, my goal is not to paint technology is a cause or prime mover behind the various movements documented in this chapter, but rather as one important factor among many. As background, I also engaged with a number of closely related themes in the preceding chapter, including through my review of the emergence of some new domains of expertise. More specifically, the development of new technologies, design techniques, and bodies of expertise contributed to the establishment of new subfields such as computer architecture and microprogramming. Further, work in these areas revealed a continued blurring of the boundaries between software and hardware, especially at the intermediate levels of computer design.

However, the preceding analysis tended to gloss over another important aspect of this history, namely the spectacular increases in the complexity of the technological art of computer design, especially from the 1970s onward. One finds, for example, dramatic increases in both the density and scale of integrated circuits, leading from medium-scale integration (or MSI, with hundreds or thousands of transistors on a chip) to large-scale integration (or LSI, many thousands of transistors on a single chip) to very large-scale integration (or VLSI, with many tens-of-thousands to hundreds-of-thousands transistors on a chip) within the span of just over a decade.[236] As one might suspect, designing and producing reliable devices of this scale – not to mention incorporating them into even larger and more complex systems – proved a formidable challenge, even for the most seasoned individuals and design teams.[237]

In fact, in his 1989 article on the co-evolution of electronics technology and computer science from the 1940s to 1970s, historian Paul Ceruzzi framed "the management of complexity" as a common concern of both electronics engineers and computer scientists, and he largely credited the field of computer science for providing engineers with the ability to cope with the task of building chips that consisted of hundreds of thousands of individual transistors (Ceruzzi, 1989). And while Ceruzzi's assumptions in this article about the respective boundaries around computer science and computer engineering are certainly debatable, it is worth probing in more

[236] *Wikipedia* provides a reasonable overview of the defining characteristics and major time periods for the historical development of integrated circuit technologies, including MSI, LSI, and VLSI ("Integrated circuit," n.d.).
[237] Tracy Kidder's Pulitzer Prize-winning *The Soul of a New Machine* (1981) provides a persuasive account of how two design groups at a well-known computer company grappled with these types of challenges in the mid-1970s.

detail the movement of knowledge and technology between the more "hardware" and "software" ends of the computing spectrum, especially through the 1970s and into the 1980s.

For example, hardware and system designers benefited greatly from the continued development and application of new hardware description languages (HDLs), especially from the 1970s onward. In addition to revealing how the knowledge and techniques of programmers and computer scientists could usefully be applied to the engineering of ever-more complex computing devices, HDLs were often viewed as providing an important bridge between hardware and software design. More specifically, these languages provided computer engineers and designers with a much greater appreciation for the *behavorial* as well *structural* aspects of hardware.[238] To put it another way these languages provided not only the ability to describe the physical structure and interconnections of a given integrated circuit and/or system, they also improved the ability of designers to specify in detail how that circuit or system would behave. Further, the development of common specification languages for both hardware and software helped enable the simulation of combined hardware-software systems. Such simulations also helped enhance the ability of designers to analyze the increasingly important trade-offs that came with shifting functionality between hardware and software.

HDLs – coupled with a variety of supporting technologies and developments – seemed to provide crucially important scaffolding for the emergence of a truly integrated approach to hardware and software design. In fact, it didn't take long for observers to sense the larger significance of these developments. In as early as 1974, for example, F. J. Mowle of Purdue noted the potential impact of HDLs in the educational sphere. As Mawle explained, the emergence of hardware description languages, new semiconductor technologies, and shifting divisions of system design labor pointed toward "an integrated education in hardware and software principles by use of a suitable combination of hardware description language and high-level programming language … This will be a promising approach to overcoming the pedagogic efficiencies of the conventional methodological segregation and its consequences" (as quoted in Chu, 1974, p. 20). Mowle's comments help bring into further relief the currents of change that seemed to be afoot as the Computer Society's educational reformers went to work on developing their ambitious new boundary-spanning curriculum in CSE, and as other groups worked more generally to outline the contours of CSE.

[238] I draw significant inspiration here from the remarks of R. Hartenstein, as quoted by Chu (1974, p. 20).

On the one hand, the rise of HDLs seems to provide support for Ceruzzi's claim that the engineers had greatly profited from the prior innovations of computer scientists. Yet the movement of knowledge between these two groups clearly flowed both ways. In fact, just as the engineers learned much from the more *behavioral* or *procedural* perspective of computer scientists and programmers, so too did computer scientists learn from the *structural* outlook of the engineer-cum-designer. At risk of overgeneralizing, the emergence of the field of "software engineering" provides evidence for this trend, where the tools and techniques of engineers were used to cope with the increasing complexity of *software*. For example, one finds the application various design methodologies and cycles to the domain of software. Further, many of the metrics long-privileged by engineers for the design of physical artifacts – such as simplicity, reliability, modularity, economy, and adaptability – were imported wholesale into the realm of software design. Hence, if the increasing complexity of integrated circuit technology showed the limits of a purely *structural* approach to engineering design, so too did the increasing complexity of large-scale software projects begin to reveal the limits of both computer science theory and purely procedural approaches to programming.

The emergence of the next major generation of integrated circuit technology – dubbed very large-scale integration, or VLSI – sheds additional light on the continued interplay between the realms of software and hardware development.[239] Commentator Ben Spaanenburg, for instance, suggestively summarized that the "VLSI 81" conference held in Edinburgh, Scotland was largely organized around the question: "Will VLSI be solved by (computer) science or (electrical) engineering?" (1982). Describing the event as a "staging area for a clash" between science and engineering, the author presented a somewhat lopsided view of the event by emphasizing the continued influx of computer science into engineering and microelectronics. In fact, he noted in that one participant in the event had put forward the view that "software is the key to VLSI design, with the crucial skill being complexity management."

While such comments seem to support Ceruzzi's claims, other evidence reveals that the flow of technology and knowledge in and around the domain of VLSI was altogether more

[239] As other commentators have noted, the difference between LSI and VLSI was not only a matter of the number of transistors that could be packed onto a chip, but also associated differences in design philosophies. LSI technologies allowed computer systems to be developed by wiring together standard LSI modules. VLSI, on the other hand, opened the way for entire systems to be placed on a single chip. As this overview suggests, the design of systems based on VLSI rather than LSI technology required very different design methodologies and approaches.

complex. Later in the same year, for example, a three-day workshop "on the engineering of VLSI and of Software" was organized by the Computer Society. According to one announcement, the workshop was conceived to examine "the role, utility and value of [s]oftware engineering practices applied to VLSI" *and* "VLSI technology and engineering practices applied to software" ("Call for Participation," 1982). Rather than promoting the tired image of persistently fragmented field, this particular event suggested that many of the technical boundaries between software and hardware had been breached. On the other hand, Spaanenburg's remarks revealed that images of science and engineering as distinct domains continued to have much currency in the field, even if it was increasingly difficult to determine where the tools, techniques, and knowledge of the computer scientist ended and the computer engineer began.

Research Directions in Computer Engineering: (Re)Defining the Discipline

While many of the technological currents outlined above appeared largely synergistic with the philosophy and motivations of the "Computer Science and Engineering" movement, other 1980s era developments simultaneously undermined it. In order to document this trend I begin in the early 1980s, when a handful of engineers worked to both (re)establish a definition for computer engineering and promote the field as a distinct discipline and domain of research. Evidence for this movement can be found in a published report on a 1981 NSF workshop on "Research Directions in Computer Engineering" (Freeman, 1982; 1983). In terms of composition, the fourteen participants at this event included six industry affiliates, while another eight hailed from the academy (p. 80). Electrical engineer Herbert Freeman of Rensselaer Polytechnic Institute served as chairman, and C. V. Ramamoorthy was among the participants (Freeman, 1983, p. 81).

As Freeman noted in a follow-up report – which was published in *Computer* – the workshop participants were ostensibly gathered to discuss the future of research in the area of computer engineering, yet they quickly surmised that it was necessary to first establish the field's definition, scope, position, and goals. More specifically, this same report indicated that the group spent considerable time on the topic of "What is computer engineering?" "Computer engineering has never been clearly defined" (Freeman, 1983, p. 80), Freeman explained, and he suggested that this condition largely stemmed from both the relative newness and rapid development of

computer technology. The author then summarized the participants' efforts to clear up some of this ambiguity, including via their own attempt at a definition:

> [C]omputer engineering is the discipline that deals with the design and development of computer systems and emphasizes such factors as function, performance, cost, size, power requirements, reliability, maintainability, and societal impact. Intrinsic to computer engineering is the concept of *design* as it applies to all aspects of a computer system – the hardware, the software, the algorithms used – and the application for which it is intended (p. 80, authors' emphasis).

The report also emphasized that computer engineering was neither science nor mathematics, especially given the orientation of engineers toward applying theory and focusing on matters of implementation. As this overview reveals, the workshop participants leveraged the historical links between engineering and design to frame computer engineering as covering a wide swath of sociotechnical territory. In fact, the authors used their rather expansive definition to claim that "most people working in the computer industry (other than pure science) are computer engineers" (p. 80).

To be sure, many self-described computer scientists probably recoiled at such claims. The suggestion that the group's definitional work was largely novel was also questionable. As documented in prior chapters, many prior commentators had made implicit and explicit claims about the scope and contours of computer engineering. For example, the COSINE Committee's widely-distributed 1970 recommendations for computer engineering options quite explicitly defined computer engineering, while also hinting at how the education and work responsibility of computer engineers differed from their computer science and electrical engineering ilk. And as noted above, Ramamoorthy himself put forward a rather succinct definition for computer engineering in his 1976 review article on "Computer Science and Engineering Education."

In light of this evidence, how do we account for this collective case of amnesia – this apparent lack of disciplinary memory – which led this group to claim that computer engineering had never been defined? To begin with, many earlier publications had framed computer engineering as a branch of engineering or dimension of electrical engineering, rather than as a discipline unto itself. In fact, one finds a similar tendency among the NSF workshop participants, especially as they worked to distinguish computer engineering and computer science. As

Freeman summarized: "[c]omputer science tends to stress understanding and insight," while "[c]omputer engineering emphasizes practical, economic systems" (p. 80). This statement implied that the workers in each of these domains possessed their own distinct outlooks or perspectives, and these roughly cleaved along the science-engineering boundary. Freeman added that this difference was especially apparent in the academic context, where computer engineering students were schooled not only in computer hardware and software, but also in basic science, engineering science, and engineering design. The author quite suggestively added that computer engineering students enrolled in ABET accredited programs were educated as "engineers first and computer engineers second" (p. 81). Such comments support the claim that the development of computer engineering as a recognized *discipline* was often and persistently overshadowed by ongoing efforts to turn engineering undergraduates into appropriately qualified *professional* engineers.

The parallel promotion of "Computer Science and Engineering" as an all-encompassing disciplinary and professional moniker only caused further confusion about both the past and present landscape of the computer-oriented disciplines. And in line with this alternate view of the field, Freeman acknowledged that computer engineers and computer scientists maintained overlapping interests in many of the same epistemological and technological territories. "Except for the very theoretical aspects of computer science and the very strong hardware aspects of computer engineering," the author explained, "the domain of interest of computer science and computer engineering are virtually the same" (p. 80). Echoing many prior commentators, Freeman ultimately concluded that the two disciplines should be regarded as "complements" rather than "competitors" (p. 82). And the workshop group even went so far as to identify the "integration of hardware and software disciplines" as one of 16 major five-year research goals for the field (p. 82).

As this overview reveals, over the span of roughly three decades the field of computer engineering had largely failed to achieve an independent disciplinary identity, even for those who self-identified with the field. In fact, computer engineering appeared stubbornly suspended between electrical engineering and computer science, profession and discipline, and the NSF workshop hinted at some of the potentially deleterious consequences that came with the field's boundary-spanning position. The report raised questions, for example, about the extent to which university administrators, students, funding agencies, and the lay public understood what

computer engineering encompassed. Perhaps even more suggestively, Freeman added that "the choice of name for a university department can have far-reaching implications on the types of students it attracts, its faculty, the kind of research support it will receive, and its ultimate growth and development" (p. 81). While the author failed to elaborate on this comment, its probable meaning is easily inferred, namely that working the phrase "computer engineering" into departmental titles was an important step toward its recognition as an academic discipline.

Yet as the following section makes clear, promoting the institutional, disciplinary, and discursive legitimacy of computer engineering stood in at least partial tension with not only this group's comments about integrating the hardware and software disciplines, but also with those who continued to lobby for a more unified approach to computer science and engineering education. And indeed, the Computer Society's next major curriculum development project was once again framed under the boundary-spanning guise of CSE, even as others were beginning to raise questions about the extent to which the group's work actually promoted the blurring or crossing of disciplinary boundaries.

From Curriculum to Program: The Engineers Revisit CSE Education

Through the late 1970s and into the 1980s the Computer Society's Education Committee continued to expand, both in terms of its membership and level of activity. In 1979, for example, chairman David Rine boasted that the group consisted of about 300 members, and he added that approximately 35 papers related to the committee's activities had recently been presented at conferences (Rine, 1979, p. 3; 4). Around this time the group was involved with assisting educators with the implementation of model curricula, developing new curriculum reports for other educational levels (including pre-college, community college, and graduate), and establishing new recommendations for more specific subfields, including graduate-level programs in software engineering (Rine, 1979). Committee members were also increasingly involved in accreditation-related work, as discussed in more detail below. The committee's institutional prominence was further elevated when it officially became the Educational Activities Board (EAB) in the early 1980s (Booth, 1982). As a result of this change, the group's leader was recognized as a voting Vice President on the Computer Society's governing board. Within the span of a decade, education had clearly emerged as a centrally important domain of activity for the Computer Society and many of its members.

Not content to rest on its laurels, by late 1981 the group was pondering a review of its 1977 curricular recommendations for CSE. Reports suggest that Ramamoorthy initiated this undertaking, and he and the other reviewers quickly concluded that a major revision of the model curriculum was needed, especially in light of ongoing and rapid technological changes (Cain, Langdon, and Varanasi, 1983, p. vi). When Taylor Booth took over as head of the EAB in 1982, he was instrumental in expanding the project to more explicitly address faculty and resource issues as well as curriculum (p. vi). This expansion of the project strongly reflected the full range of implementation challenges that many institutions faced as they attempted to establish educational programs in CSE and related areas. The group's final report was first published in late 1983 and also appeared in summary form in an April 1984 issue of *Computer* that was dedicated to "Computers in Education" (Model Program Committee, 1983; Cain, Langdon, and Varanasi, 1984).

In summary, the new model program took an evolutionary rather than revolutionary step beyond the group's earlier work.[240] One change worthy of noting centers on the core curriculum, where outlines for individual courses were replaced with descriptions of 13 core subject areas, with 9 of these marked as lecture/recitation and 4 as laboratory.[241] The report also identified a diverse assortment of 15 advanced subject areas, and recommended that any given program should provide in-depth coverage of at least two. Elsewhere in the report, the authors stressed the importance of striking a balance between hardware and software-oriented CSE programs: "The curriculum component of the program is intended to provide potential graduates with a well-balanced education in fundamental principles of hardware and software design, reinforced with experiential skills" (16).

The Committee also echoed the COSERS and AFIPS Taxonomy projects by explicitly framing Computer Science and Engineering as a discipline. The authors of the report described electrical engineering and mathematics as the main "sister fields" (p. 99) or "sister disciplines"

[240] Engineer V. Rao Vemuri – who served as a member of the committee that produced the 1983 Model Program in CSE – nicely captured in a 1993 article the tendency for model curricula to develop in a conservative manner: "Indeed, there has never been a shortage of studies on model curricula. … What seems to impede progress is that curricular recommendations have the tendency to exhibit tremendous 'implementation inertias.' They resist all but the most incremental changes" (Vemuri, 1993, p. 108).
[241] The nine lecture/recitation areas included Fundamentals of Computing, Data Structures, System Software and Software Engineering, Computing Languages, Operating Systems, Logic Design, Digital Systems Design, Computer Architecture, and Interfacing and Communication. The subject area of Discrete Mathematics was also identified as a crucial supporting topic for CSE.

(p. 120) of computer science and engineering, for example, and elsewhere they noted that "computer science and engineering is now recognized as a separate, identifiable discipline" (p. 98). Such comments once more reveal the strategic importance of building disciplines through discourse, especially given that only a handful of departments and programs actually carried the title of "computer science and engineering" by this time. In fact – and like the advocates of computer science before them – the authors of the model program report were forced to acknowledge the potential for wide variation in how CSE might be realized in diverse institutional contexts. "The administrative structure used to support programs in this area," the report explained, "can take a variety of forms" (p. 98).

Once again, we find a persistent and unresolved tension in the disciplinary landscape of computing. To whit, the authors' insistence on the existence of CSE as a distinct disciplinary field was accompanied by a lack of consensus regarding the prevailing or even preferred location for its associated educational programs. And elsewhere in this report, the authors revealed their preference for linking CSE with engineering, even as they emphasized that the field spanned the hardware-software spectrum:

> The undergraduate program in computer science and engineering must contain a core that gives each student a comprehensive understanding of the hardware and software principles underlying the computer area. In addition, the student must also have a strong background in mathematics, the basic sciences, and the engineering sciences (p. 123).

As one might suspect, the "core" summarized in this passage seemed to have much in common with other types of engineering programs. That is, engineering students in diverse sub-fields were expected to pass through a reasonably standard sequence of foundational courses in math, science, and engineering science before moving on to more specialized engineering subjects.

Still other publications revealed that various actors were eager to promote and oversee the development of CSE departments and programs *within* colleges and departments of engineering. In a 1984 review article on the topic of "Computer Education," for example, former COSINE Committee member and Computer Society EAB Vice-President Taylor Booth echoed the Model Program report when he noted that "computer science and engineering has matured into a well-defined disciplined" (p. 64). Yet even more suggestively, the author added:

> [M]any schools are reconsidering their departmental structures and creating computer science and engineering departments, apart from electrical engineering. This trend should accelerate in the next few years, and by the 1990's, the computer science and engineering department will be considered key to any engineering school that wishes to offer a full-spectrum program" (Booth, 1984, p. 64).

Booth's comments reveal growing recognition among educational reformers regarding the importance of building the discipline computer science and engineering through the establishment of thusly-named departments and programs, albeit within colleges and schools of engineering. In fact, Booth noted that recent revisions to ABET criteria for computer science and engineering programs had been "revised to reflect the fact that it [CSE] is a distinct engineering discipline" (p. 64).

On a closely related note, Booth was also a member EAB task force that was formed in 1984 to analyze the role of "design education" in computer science and engineering. In addition to dovetailing with both the Computer Society's 1983 model program and ABET accreditation criteria that I discuss below, the group's recommendations – which were published in 1986 – emphasized the integral role of engineering design in the field of computer science and engineering, especially given "the many conceptual levels involved in information systems, from hardware components to complex software systems" (Booth, et al., 1986, p. 26). This same report revealed the potential for engineers to strategically leverage their historical monopoly on "design education" in order to retain control of the domain dubbed Computer Science and Engineering. "In the world of technology," the authors explained, "design has been the traditional province of the engineer and differentiates the engineer from the scientist" (p. 21). And while the ongoing development of software engineering courses and programs in a variety of departments suggested that both engineers and non-engineers maintained overlapping interests and claims in the arena of software design, the following section reveals the importance of accreditation structures and processes in ongoing efforts to establish and police the boundaries around engineering and design.

Engineering Accreditation and The Discursive Politics of Professional Certification

Given its mediating role betwixt profession and discipline, industry and the academy, accreditation is an important topic. In fact, the present case helps illustrate the pivotal role that professional societies often assume at the intersection of profession and discipline, especially as their members undertake tasks such as developing accreditation criteria and reviewing individual academic programs. As further background on the topic, it is worth briefly reviewing some relevant history.[242] In the wake of the 1930 publication of the influential Wickenden report on engineering education, a climate of cooperation helped enable the 1932 founding of the Engineers' Council for Professional Development (ECPD). With the early support and active participation of seven major engineering professional societies, the group went to work, approving its first set of accreditation criteria for engineering programs in 1933 and issuing its first accreditations in 1936. The group was quickly recognized as *the* accreditation body for U.S. educational programs in engineering, engineering technology, and related fields.

The ECPD maintained a single set of accreditation criteria for all types of programs for many decades, and these grew longer and more complex through the 1960s and 1970s as the various fields and subfields of engineering continued a long historical pattern of evolution and diversification. And indeed, computer engineering was one among many fields that emerged and were eventually recognized by the ECPD. As reported by Jones and Mulder (1984, p. 25), in 1971 the computer engineering program at Case Western Reserve University became the first accredited engineering program with the word "computer" in its title. Syracuse University and the University of Connecticut followed close behind, with the former named "computer engineering" and the latter dubbed "computer science" (p. 25).

The accreditation of these programs was an important step in the recognition of computer engineering as a partially or perhaps even wholly distinct field or discipline, although the existence of Connecticut's engineering-oriented computer science program revealed continued uncertainty over how such programs should be named. Further, these programs were accredited under "special" EPCD guidelines since no specific criteria existed for these relatively new types of degrees. Amidst growing concerns that more specific guidelines were needed for computer-oriented engineering programs, a Computer Society committee chaired by Ramamoorthy went to work on the problem in 1975 (Jones and Mulder, 1984, p. 25). The ECPD approved the group's

[242] This brief historical review is largely based on the accessible account developed by Stephan (2002).

recommendations for "Computer Engineering" in 1978, and they were first used for accreditation visits in 1979 (p. 25). In addition to "amplifying" and "interpreting" the ECPD's general criteria, the IEEE guidelines stressed the importance of education in the areas of engineering design and the engineering sciences (IEEE Educational Activities Board, 1978).[243]

The EPCD was renamed the Accreditation Board for Engineering Technology (ABET) in 1980, and by October of the same year the organization had accredited a total of 10 bachelor's level and 3 master's level programs in the "Computer" program area (ABET, 1980). In the early 1980s the Board also spearheaded the development of more specific "program criteria" that would be used for the accreditation of programs in various sub-fields of engineering, thereby reducing ongoing confusion over the publication of separate, supplemental guidelines by various professional societies (Jones and Mulder, 1984, p. 25; ABET, 1982). The IEEE responded with a set of program criteria for "computer and similarly named engineering programs," and these were approved and in active use by the mid-1980s (Jones and Mulder, 1984, p. 26).

As noted above, the Computer Society's development and promotion of *A Curriculum in Computer Science and Engineering* in the mid- and late-1970s dealt with the matter of accreditation rather lightly, although the inclusion of an "Electrical and Computer Engineering Curriculum" designed to meet EPCD guidelines revealed that this group was at least nominally interested in making their recommendations accreditation-friendly. The Computer Society's 1983 Model Program, on the other hand, revealed the increasing importance of explicitly dovetailing the group's educational recommendations with ABET criteria. The first major section of the program report, for instance, featured detailed information about the ABET general criteria, as well as draft criteria for what the group variously referred to as "Computer Science and Engineering Programs" or "Computer and Similarly Named Engineering Programs" (Model Program Committee, 1983, pp. 3-6). The curricula section of the report also featured three sample course-by-course implementations of four-year CSE programs, all assumed to be situated in schools of engineering, and all satisfying ABET accreditation criteria (pp. 89-97). Interestingly enough, no other sample implementations were presented, which further suggested that the group's primary interests did indeed center on the development of CSE or CE programs within departments and schools of engineering.

[243] According to one preliminary report, these guidelines were intended for "computer engineering, computer science, information science, and similar programs for which ECPD accreditation is request" (IEEE Educational Activities Board, 1978, p. 67).

In 1984, a *Computer* article co-authored by Michael Mulder and Edwin C. Jones also dealt rather extensively with the matter of accreditation. In addition to summarizing the historical development and contemporary status of accreditation in the area of "Computer Science and Engineering," the authors outlined a series of "Issues and Concerns" that they felt warranted further study. It is worth quoting the authors at length here, as they nicely captured many of the persistent concerns and tensions that accompanied both the ongoing development of computer-oriented curricula and the changing disciplinary landscape of the computer field:

---

1) What is the role of basic science in computer science and engineering education, and what are the appropriate basic sciences for CSE programs?
2) What is the role of programming courses? The general trend today is not to allow programming courses to be considered in any of the five major curriculum classifications because they are considered skills, not course material in engineering science or design.
3) What is computer engineering? How does it differ from computer science? Are the distinctions worth noting? Could the discipline be called computer science engineering [sic] or some other title? Should the professional societies move toward combining these disciplines if they are separate?
4) What is software engineering and is it really "engineering"? Some people tend to look at the software problem and argue that, since it does not involve hardware, it is other than an engineering concern. This view is, of course, disputed by those who see the decision to make a trade-off in design between hardware and software as purely an engineering problem.
5) What should we do with model programs? New model programs have been prepared to set goals and provide guidance. In the near future, we should try to incorporate these ideas into the program criteria.

(Jones and Mulder, 1984, p. 27)

In most general terms, this list of issues revealed many points of instability in the milieu of computer-oriented education. In fact, one is struck by the extent to which these authors were questioning the fundamental nomenclature and identity of their own disciplinary field and its

related subfields – especially in contrast to other reports published around this same time period. As noted above, for example, the 1983 Model Program had rather confidently declared that "computer science and engineering" was indeed a "separate" and "identifiable" discipline.

Three additional and more specific themes are also worth highlighting here. First, one finds tensions running through this passage regarding what should or should not be counted as engineering, and how engineering was related to the sciences. Second, the authors once again adopted a Janus-faced position, especially by implying that there was such a thing called "computer science and engineering education," and then raising questions about whether computer science should be distinguished from computer engineering. Third, Jones and Mulder questioned whether the professional societies might somehow "move toward" a merger of the computer engineering and computer science disciplines, thereby putting forward an image of the professional societies as key loci of disciplinary development. While my account certainly speaks to the role of such societies in the emergence and building of disciplines, the present chapter also emphasizes both the heterogeneous nature of discipline building in general and the importance of the academic context in particular.

Data culled from a series of ABET annual reports provides a more detailed view of the ongoing development of computer-oriented engineering degrees and programs in the early and mid-1980s. To begin with, the number of accredited programs in the so-called "computer area" continued to rise. By October of 1985, for example, the number of ABET recognized programs at the bachelor's-level had risen to 34 (ABET, 1985, p. 37). On the surface, these numbers may appear rather small, especially given that many hundreds of electrical engineering and computer science departments and programs were in existence by this time. However, this data only reflected those programs that were *specifically* accredited by ABET in the computer area. In fact, there were surely many ABET-accredited electrical engineering programs that offered *options* in computer engineering, computer science, and related areas. Yet the incorporation of computer engineering as part of a multi-option degree structure suggested that computer engineering was a branch or subfield of electrical engineering, rather than a discipline unto itself.[244]

These same ABET reports also revealed important trends in the naming of programs. To begin with, the term "Computer Engineering" remained popular and influential, and by 1985 it

[244] As noted above, Sloan's data from 1974 showed that 51% of EE departments responding to her survey offered CS or CE degrees or options. And as noted in Chapter 5, a similar survey conducted by Sloan in 1972 revealed that 49% of responding EE departments maintained Computer Engineering degree options.

was being used to describe 18 of 34 accredited programs (ABET, 1985, p. 60). The phrase "Computer Science and Engineering" also became more widespread in the early part of the 1980s, appearing in just two accredited program titles in 1982 and a more impressive seven in 1985 (p. 60). By 1985, other program names included "Computer Science" (3 programs), "Computer Systems Engineering" (3 programs), and a smattering of more unusual one-offs (p. 60).[245] And on a closely related note, a new ABET guideline in the mid-1980s mandated that all accredited engineering programs must include the word "engineering" in their titles after 1985 (Jones and Mulder, 1984, p. 26). According to one account, the phrase "and engineering" was simply appended to many program names in order to satisfy this requirement (Yeargan, 2002, p. 111). Nonetheless, this anecdote forcefully reveals that the policing of discourse can play an important role in ongoing efforts to construct and maintain professional and disciplinary identities.

CSAB and CSAC: Independent Accreditation for an Independent Discipline

No matter how any given degree program was named, it was clear that the criteria and guidelines published and used by ABET were by definition intended for *engineering* programs. Hence, computer science and other computer-oriented programs situated outside of colleges and departments of engineering lacked a suitable and widely-recognized accreditation processes. Yet the need for such an accreditation system had been recognized much earlier. In fact, interest in the matter among ACM members and leaders can be traced back to at least the late 1960s and early 1970s, with ACM President Walter Carlson identifying accreditation as one of his top goals for the organization in 1969. More specifically, he called on the group to issue curricular recommendations for all levels of computer education by 1972, and accredit at least fifty percent of all computer-oriented educational programs by 1980 (Carlson, 1969).

Yet many within the ACM were skeptical about the potentially stifling effect of accreditation on computer science programs, even if they were willing to support some sort of certification process for courses of study in programming and related areas, many of which were being offered by trade and technical schools. As summarized by the Secretary of the ACM SIGCSE, "In view of the developing nature of computer science it was observed that an

[245] These other program names included Computer and Electrical Engineering (Purdue), Computer and Information Engineering Sciences (University of Florida), Computer and Systems Engineering (RPI), and Computer Science Engineering (San Jose State University) (ABET, 1985, p. 60).

accrediting committee might serve to stifle rather than encourage the natural development of computer science curriculae. … After lengthy interchange the only consensus was that accreditation in computer science similar to accreditation procedures in other more established disciplines is not in the immediate future" (Matula, 1969).

In striking contrast to the engineering community – where accreditation had for many decades played a pivotal role in the development and recognition of educational programs – the computer science camp privileged their independence, thereby stalling Carlson's ambitious agenda. In fact, it wasn't entirely clear what it would mean to "professionalize" computer science through accreditation programs, especially since the field's dominant image was largely based on its status as an academic discipline. Certifying professional programmers, on the other hand, was a somewhat more palatable prospect since programming was often viewed as closely linked to – but also partially distinct from – computer science.

These tendencies were reflected in the 1973 establishment of the Institute for the Certification of Computer Professionals (ICCP) by eight professional organizations, including the ACM and IEEE Computer Society (McCracken, 1979, p. 145). Through the 1970s the ICCP initiated and administered the Certificate in Computer Programming (CCP), yet this credential was primarily aimed at individual programmers. It therefore had only indirect bearing on both computer science generally and computer-oriented degree programs specifically. In 1977 the ACM finally approved its first set of accreditation guidelines for "Bachelor's Degree Programs in Computer Science," but even these were primarily intended for use by individual institutions for informal, self-study, and/or for use in connection with regional accreditation procedures (ACM Accreditation Committee, 1977).

As both the number of computer science programs continued to expand the concerns about their quality persisted, around 1981 the ACM and the Computer Society finally took the first steps toward developing a new accreditation process that was tailored for computer science education (Mulder and Dalphin, 1984, p. 30). A joint task force co-chaired by Michael Mulder of the Computer Society and John Dalphin of the ACM went to work on the problem, and in 1982 they recommended the establishment of a new accreditation body (p. 30). This led to the 1984 founding of the Computing Sciences Accreditation Board (CSAB) as an independent, non-profit organization, with the Computing Sciences Accreditation Committee (CSAC) established shortly thereafter to oversee the actual accreditation process. During the first accreditation cycle in the

Fall of 1985, a total of 31 schools were reviewed and 23 approved (Booth and Miller, 1987, p. 379). On a closely related note, this same report noted that the ACM maintained a master list of more than 450 institutions that offered some type of four-year undergraduate degree in computer science. The CSAB clearly had a long list of prospective clients (p. 378).

The joint creation of the CSAB by the ACM and Computer Society stands as another testament to the reasonably close relation of these two professional societies in the early and mid-1980s. In fact, this same time period was marked by other signs of cooperation, including renewed discussions about the possible advantages of merging the two groups. Yet just as the Computer Society remained institutionally suspended between the IEEE on the one side and the ACM on the other, the accreditation criteria and processes for computer-oriented degree programs had developed in a similarly bi-furcated manner, with both ABET and the CSAB pursuing partially independent goals and certifying different types of programs.

In fact, Figure 7.1 – which was originally presented by Mulder and Dalphin in their 1984 article on Computer Science accreditation – suggestively depicts these tensions. On the one hand, we find Electrical Engineering and Computer Science Engineering explicitly aligned with both Schools of Engineering and ABET Accreditation, while Computer Science and Information Science were linked to "Liberal Arts and Science." Yet the anticipated "program range" for CSAC accreditation was framed as covering a broad span of the so-called "Computing Sciences," even reaching into the domain of "Computer Science and Engineering." As I note below, reconciling the overlapping jurisdictions of ABET and CSAC emerged as an increasingly important issue, especially in the late 1980s and into the 1990s.

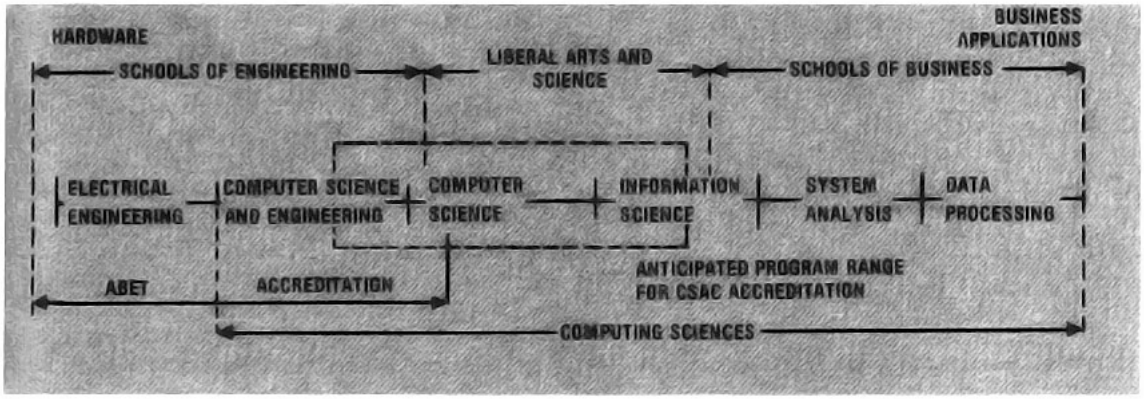

Figure 7.1 – Distribution of Computer Science Programs with Present and Projected Accreditation (Mulder and Dalphin, 1984, p. 31) © 1984 IEEE

In following section I document how the close relation of the ACM and Computer Society led to another unprecedented project, namely the joint development of a common curriculum. On the other hand, my analysis once again reveals a variety of incongruities between the development of new model curricula, the historically dominant structure of academic departments and programs, and the ongoing evolution of accreditation criteria and processes.

The Diversification of Computer Science Curricula

If the development of the CSAB/CSAC hinted at a continued independent streak among computer science programs and departments, so too did a series of 1980s-era publications on computer science curricula. In fact, many of these documents allowed various groups and actors to refine their definition of computer science, while also resisting or simply ignoring the alternate educational agenda that was being developed by the Computer Society. The ACM, to begin with, selectively updated its curricular recommendations in the early and mid-1980s. In 1981, for instance, the ACM C^3S published a set of "Recommendations for Master's Level Programs in Computer Science" (Magel, et al., 1981). The report offered little in the way of surprises, as the group's recommendations were significantly informed by a number of M.S. programs in Computer Science that were already in existence at various schools.

More specifically, the report proposed curricular coverage in a number of predictable subject areas, such as programming languages, theoretical computer science, and data and file structures. Courses in a fourth topical area, namely "Operating Systems and Computer Architecture," provided students with some exposure to hardware and systems, although it was clear that these programs were more generally rooted in computer science rather than engineering. And a pair of subsequent ACM task force committees developed revised curricula for the first two courses recommended in *Curriculum '78*, namely "CS1" and "CS2." Respectively published in 1984 and 1985, these new recommendations were framed as a responding to both an increase in computer science knowledge and the need for greater emphasis on software engineering (Koffman, Miller, and Wardle, 1984; Koffman, Stemple, and Wardle, 1985).

Other groups were also leading the development of curricula in the early and mid-1980s, and these tended to reflect some of the schisms that were growing both within computer science and between computer science and computer engineering. *The Carnegie Mellon Curriculum for*

Undergraduate Computer Science (Shaw, 1985), to begin with, was the result of a three-year effort by an eight-member group affiliated with Carnegie Mellon's Computer Science Department. This undertaking was largely prompted by the decision to establish an undergraduate Bachelor's degree in Computer Science at the school, although Carnegie Mellon's Computer Science Department and associated Ph.D. program had been in existence since 1965 (Preface). The authors also framed their work as forward-looking, and they complained that the ACM's prior recommendations were overly conservative, disunified, and lacking in mathematics content (pp. 18-19).

The authors of this report also critiqued the Computer Society's curriculum as "heavily biased toward hardware," and they noted that it "fails to expose the important common fundamentals in joining hardware and software" (p. 19). They went on to conclude that the proposed curriculum "might be reasonable for a curriculum directed purely at the electrical engineering side of the discipline, but the designers claims that the curriculum is suitable for computer science" (p. 19). As suggested by these remarks, these computer scientists clearly viewed the Computer Society's curriculum as ultimately focused on "computer engineering," despite the discursive garb of "computer science and engineering" that surrounded it.[246] The group responded with their own nearly 200-page report. In addition to once again revisiting questions about the nature and definition of computer science, the authors emphasized the balanced integration of theory and practice, outlined a total of 30 courses, and proposed overall requirements for undergraduate degree programs.

"A Model Curriculum for a Liberal Arts Degree in Computer Science" (Gibbs and Tucker, 1986), on the other hand, was based on another partially unique set of interests and philosophies. In part developed through two major workshops, this project was funded by the Alfred P. Sloan foundation. In their final report, co-authors Norman E. Gibbs and Allen B. Tucker noted the increasing obsolescence of the ACM's *Curriculum '78*, and they critiqued the more recent CSAB accreditation standard "for its inflexibility and for its strong bias toward a professional engineering education" (p. 203). Along similar lines, they credited the Carnegie Mellon report for promoting a "liberal professional education," but nonetheless faulted its "significant engineering point of view" (p. 203). The authors of the report went on to stress that

[246] The authors of the Carnegie-Mellon report even referenced the "IEEE computer engineering curriculum," in spite of the fact that it was actually titled "computer science and engineering" (19).

"computer science *is* science," and they emphasized that "[i]n defining *computer science*, we should be able to distinguish it from *computer engineering*, just as *chemistry* is distinguished from *chemical engineering*, and *physics* from *mechanical and electrical engineering*" (p. 204, authors' emphasis). The curriculum outlined by the group – which was largely organized around just four core courses – was explicitly framed as leading to B.A. degree within a liberal arts setting. As suggested by this overview, this curriculum represented a growing bifurcation of computer science education, where the needs and interests of particular schools and faculties appeared increasingly divergent from other types of institutions. The Gibbs and Tucker report also revealed an explicit resistance to the educational imperatives of industry, which by this time were increasingly influential on the curriculum development and accreditation activities of groups such as the Computer Society and ACM.

One therefore finds in these various reports and recommendations a continued diversification of computer science curricula. Perhaps not surprisingly, this posed challenges to those who preferred a more unified model for a wide range of educational programs, ranging from computer science programs oriented to the liberal arts to computer engineering options deeply rooted within engineering. Yet as the following sections make clear, these challenges did not stop the actors and groups involved with the "computing as a discipline" movement from making their own moves toward unifying computer-oriented education.

From Discipline in Crisis to Computing as a Discipline

While the aforementioned Liberal Arts report was clearly inflected by the location of its authors in a particular type of institutional location, this group also expressed hope that their efforts might be viewed as a contribution to a complete overhaul of *Curriculum '78*. And while a liberal arts point of view was explicitly included in the ACM's next set of curricular recommendations, it would be a number of years before this important next chapter in history of computer-oriented curricula got underway. As background, the "Computing as a Discipline" movement laid the initial foundations for a complete revision of the ACM's recommended curricula. Largely led by well-known computer scientist Peter Denning, the origins of this movement can be traced back to the biennial Snowbird conferences, which were organized by and for the leaders of doctorate-granting computer science programs. Designed to grapple with the major issues that were facing the field at any given time, the 1980 and 1982 meetings were

noteworthy for their explicit emphasis on the so-called "crisis in computer science," which involved an ongoing and acute shortage of trained personnel in the computer field, especially at the Ph.D. level. The reports that came out of these two events included a number of recommended improvements to the environment in computer science departments, in hopes of attracting more students and turning out more graduate degrees (Denning, et al., 1981; Yau, et al., 1983).

While the 1984 Snowbird report trafficked in much the same territory, it also grappled more explicitly with the topic of "computer science as a discipline." Noting "[t]he continued skepticism of scientists from other disciplines concerning the substance of computer science," the authors added that "computer scientists have no single picture of the nature of their own field. … [N]o core description is universally accepted" (Tartar, et al., 1985, p. 102). Such comments reveal the somewhat anomalous character of computer science in the academic landscape. In spite of two decades worth of history – not to mention the existence of approximately 1200 undergraduate degree programs bearing the stamp of computer science – the discipline seemed to lack a widely-recognized definition or description. And as the report acknowledged, computer-related courses and programs occupied diverse positions within the university structure, and the authors complained that "this situation fragments resources and weakens the cases made by these departments for additional funding" (p. 102). In light of these issues, the authors urged improved cooperation among the leading departments in the field. Just as importantly, they called for the development of a "unifying image of computer science" (p. 105).[247]

One finds striking parallels here with the aforementioned 1981 workshop on "Research Directions in Computer Engineering." But while the NSF workshop seemed to generate relatively little in the way of follow-up action, the Snowbird report helped stimulate the establishment of the ACM's Task Force on the Core of Computer Science in 1985, with Denning acting as chair (Denning, et al., 1989a, p. 9). The Computer Society cooperated enthusiastically in the undertaking, and no less a figure than Michael Mulder served as a member of the group (p.

[247] Denning likely played a leading role in the development of this agenda. In a commentary piece that was both passed on his opening address at Snowbird 84 and later published in both the CACM and Computer, Denning urged his academic colleagues to revisit faculty salaries, equipment and facilities, promotion and tenure, and the treatment of junior faculty. However, he also stressed the importance of long-range planning, revisiting the core curriculum, improving relations with other disciplines, and moving into new research areas.

9).[248] Originally charged with developing a description for computer science, proposing a teaching paradigm for the field, and outlining an introductory course sequence, the group ultimately developed what they described as a "new intellectual framework for our discipline and a new basis for our curricula" (Denning, et al., 1989a, p. 10). And while originally focused on the domain of "computer science," the group quickly extended its work to cover computer engineering, reasoning that "no fundamental difference exists between the two fields in the core material" (p. 10). In fact, the group's final report was titled "Computing as a Discipline," reflecting their desire to "embrace all of computer science and engineering" with a new moniker that was boundary-spanning, catchy, and succinct. The work of this task force was also distributed widely. In addition to being published in stand-alone form, condensed versions of their final report also appeared in *Communications of the ACM* and *Computer* (Denning, et al., 1988; 1989a; 1989b).

One central feature of the group's report was its emphasis on the three fundamental "paradigms" or "cultural styles" of computing, namely theory, abstraction (or "modeling"), and design. As the authors explained, these three paradigms were respectively rooted in mathematics, experimental science, and engineering. And although they acknowledged that so-called computer scientists tended to focus on theory and abstraction while computer engineers were more concerned with the abstraction and design, the authors developed a boundary-spanning definition for work in all phases of the field:

> The discipline of computing is the systematic study of algorithmic processes that describe and transform information: their theory, analysis, design, efficiency, implementation, and application. The fundamental question underlying all of computing is, "What can be (efficiently) automated?" (Denning, et al., 1989a, p. 12).

The concluding question presented in this passage reveals the influence of the aforementioned COSERS report on this group's work. The authors' efforts to position "computing" within a larger disciplinary landscape also advanced arguments that had been variously trotted out in previous publications, such as the COSERS report. They emphasized, for example, that the roots of the field extended deeply into both mathematics and engineering, and elsewhere they claimed

[248] In a reciprocal gesture, the Computer Society established a task force on computing laboratories, with the cooperation of the ACM (Denning, et al., 1989a, p. 9).

that “The science and engineering [of computing] are inseparable because of the fundamental interplay between the scientific and engineering paradigms within the discipline” (p. 16). The authors were clearly focused on the core of this settlement rather than its many peripheries.

Further fleshing out their vision for the discipline of computing, the authors proposed the segmentation of the field into nine distinct sub-areas, including algorithms and data structures, programming languages, architecture, numerical and symbolic computation, operating systems, software methodology and engineering, database and information retrieval systems; artificial intelligence and robotics, and human-computer communications (Denning, et al., 1989a, p. 12). And given that each of these areas could also be viewed in terms of theory, abstraction, and design, the authors presented the discipline writ large as a nine by three matrix, and in an attached appendix they included summary descriptions for all twenty-seven of the constituent boxes. These were then used to inform the development of a new curriculum for an introductory course sequence, which was also elaborated in significant detail in an Appendix to the group’s full-length report (Denning, et al., 1988, pp. A-II-1-18).

At least on the surface, “Computing as a Discipline” looked like an important document. It was developed under the auspices of the ACM but with the cooperation of the Computer Society, and it put forward an innovative new integrative structure for all phases of the computing field. The authors had carefully articulated a definition for the proposed discipline that both built on prior work and pointed the way toward the further development of curricula. The efforts of the task force also helped stimulate another unprecedented development, namely the establishment of an ACM/IEEE-CS Joint Curriculum task Force in early 1988 (Tucker, et al., 1991). This new group was co-chaired by Allen B. Tucker – who was a key player in the development of the aforementioned liberal arts computer science curriculum – and Bruce Barnes, an NSF division director with both longstanding ties to the computer science community and significant earlier experience with curriculum development. And while the chairmen of the group seemed to reflect a bias toward computer science, the other members of the fourteen-member task force included well-known engineering reformers such as Michael Mulder and J. Thomas Cain.

The activities of the task force spanned a period of roughly two years, and included eight major working meetings and numerous panel presentations. The group’s recommendations also went through three major rounds of reviews, involving dozens of educators. Ultimately dubbed

"Computing Curricula 1991" (CC1991), the group's final recommendations were first published in late 1990, and were summarized in both *CACM* and *Computer* in 1991 (Tucker, et al., 1991; Tucker, 1991; Tucker and Barnes, 1991). In summary, the authors' recommendations were extensively informed by the results of the "Computing as a Discipline" project. They explained, for example, that their curricular recommendations were intended "for baccalaureate programs in the discipline of *computing*, which includes programs with the titles 'computer science,' 'computer engineering,' 'computer science and engineering,' and other similar titles" (Tucker, et al., 1991, p. v).[249] The group also reiterated the importance of theory, abstraction, and design as the three main processes or "point of view" in the computing field, and they explained that the nine major subject areas identified by their predecessors "cover the entire discipline." In organizing the common requirements for all undergraduate curricula in the discipline, these nine areas were further broken down into smaller "knowledge units."

Yet the CC1991 report went beyond prior efforts, including by identifying a set of twelve "recurring concepts" that were framed as fundamental for the discipline. These diverse, boundary-spanning themes ranged from complexity and efficiency to security and tradeoffs, and the report noted that they could help play a unifying role in the development of courses and curricula. "By pointing out and discussing the recurring concepts as they arise," the authors explained, "the conscientious instructor can help portray computing as a coherent discipline rather than as a collection of unrelated topics" (Tucker, et al., 1991, p. 15). Perhaps more than any prior author or group, this task force can be credited with articulating a unified core of knowledge, skills, and concepts that truly spanned the full spectrum of the field in question.

But even as the authors worked to present a single, coherent set of underlying principles for the design of their curricula, their recommendations once more reflected the Janus-faced character of "computing." In outlining the motivations behind their work, for example, the authors pointed out that "the discipline and its pedagogy have changed significantly in recent years," and they went on to note "growing recognition of substantial curricular commonalities among programs, despite strong and fundamental differences" (Tucker, et al., 1991, p. 2). The tension between commonalities and differences was particularly evident in an appendix featuring twelve detailed sample curricula. Nine of these curricula were framed as preparatory for entry

[249] The authors went on to clarify that "[p]rograms in related areas, such as information systems, were not considered by the Task Force" (Tucker, et al., 1991, p. 2).

into the “computing profession,” and these were further broken down into three specific implementations for Computer Engineering programs, four for Computer Science, one for a liberal-arts-oriented Computer Science, and one for Computer Science and Engineering (Appendix A). These nine implementations were also predictably linked to corresponding accreditation criteria, with Computer Engineering programs designed to satisfy EAC/ABET guidelines, the Computer Science programs designed to satisfy CSAC/CSAB criteria, and the CSE implementation designed to meet both.

The report also included three additional implementations that largely ignored accreditation criteria in order to meet goals other than the training of so-called “computing professionals” (pp. 136-154). These implementations included two computer science programs that were even more explicitly oriented toward the liberal arts, clearly reflecting the influence and interests of task force members such as Tucker. The third such program, on the other hand, emphasized mathematics, theoretical foundations, and formal methods, thereby providing foundations for graduate studies in computer science or related areas. This twelfth curriculum once again reflected an emphasis on disciplinarity among many computer scientists.

Perhaps more than any other report or document from this time period, *Computing Curricula 1991* captured in a single document the Janus-faced character of educational programs in the domain of “computing.” On the one hand, the underlying philosophy of CC1991 centered on the idea that all phases of the computing field were united via both a common concern with abstraction and a shared interest in various overlapping subject areas and recurring concepts. Placing further emphasis on the “core” of the field, the developers of CC1991 carefully and strategically crafted a set of “common requirements” that were framed as foundational for educational programs in all phases of the field. On the other hand, the twelve sample implementations revealed the extent to which the multiple faces of computing were stubbornly persistent in the academic context, with ABET-accredited computer engineering programs at one end of this spectrum to liberal arts-oriented Computer Science programs at the other. CC1991 therefore reflected both the core and the peripheries of computing, and it rather uneasily straddled some of the major axes of difference – such as science-engineering, theory-design, and discipline-profession – that constituted yet simultaneously divided the field. In spite of the premise and promise of CC1991, its authors remained partially constrained by the organizational field in which their work was situated. And continued instability in the disciplinary and

professional boundaries of the field profoundly shaped this work, while also setting the stage for later developments.

The Shifting Institutional Landscape of Computing

It is worth taking another step back to more generally assess the institutional backdrop for the development and publication and CC1991. As noted above, available data suggests that more than half of all electrical engineering departments were offering programs or options in computer engineering by the late 1970s, while the number of departments offering ABET accredited undergraduate degrees in computer engineering and closely related areas had risen to 34 by late 1985 (ABET, 1985, p. 37). Yet as subsequent ABET annual reports revealed, there was a very noticeable uptick in the number of undergraduate programs in the computer area in the late 1980s and early 1990s. More specifically, by November of 1988 there were 55 such programs at the bachelor's level, and by November of 1990 there were 58 (ABET, 1989, p. 37; ABET, 1990, p. 34). These reports also reveal important naming trends. In 1990, for example, 43 programs carried the title "Computer Engineering," 9 were dubbed "Computer Science and Engineering," and the handful of remaining programs carried designations ranging from "Computer Systems Engineering" to "Computer Science" (ABET, 1990, p. 66). Not only do these data reveal the continued establishment of a partially distinct professional and disciplinary identity for computing engineering, they also suggest that the term "Computer Science and Engineering" had gained only modest traction in the academic context, despite of the Computer Society's prior promotional work.

These same reports also hint at the extent to which ABET-accredited programs in the computer area were primarily focused on the training of future professionals, especially via the granting of bachelor's degrees. In fact, only two computer programs at the master's level maintained ABET accreditation during this time period. Another perspective on these trends can be gleaned from the results of the Taulbee survey, which annually solicits data from those U.S. and Canadian departments that grant doctoral degrees in computer science and computer engineering. According to the 1989-1990 Taulbee report, the survey was sent to a total of 136 CS and 34 CE Ph.D.-granting departments (Gries and Marsh, 1992, p. 133). These numbers alone reflect the ongoing tilting of computer science toward the trappings of disciplinarity, where doctoral degrees and independent departments reign supreme. By contrast, computer engineering

remained more professionally oriented, although the existence of 34 CE graduate programs suggested an ongoing expansion of the field's disciplinary identity, especially through academic research and same-named graduate degrees.

This same Taulbee survey also revealed continued instability in the naming of departments – and a scattered assortment of other institutional homes – for graduate programs in CS and CE. First, the authors reported that an impressive 96 of the surveyed departments (and other Ph.D.-granting academic units) carried the title of "Computer Science" or "Computer Sciences," revealing the continued salience of this disciplinary identifier (Gries and Marsh, 1992, p. 134). Second, the term "Computer Engineering" and its closest variations were counted a total of 42 times in the survey, including a further breakdown of 23 instances of the name "Electrical and Computer Engineering," 12 of "Computer Science and Engineering," 4 of "Computer Engineering," and a handful of one-off variations such as "Computer Engineering and Science," "Electrical, Computer, and Systems Engineering," and "Electrical Engineering and Computer Engineering" (p. 134).[250] As suggested by these data, the frequent concatenation of the term "computer engineering" with other titles revealed the continued lack of a distinct and independent disciplinary identity for the field. Further, the twelve occurrences of the name "Computer Science and Engineering" once again reflected the relatively meager take-up of this boundary-spanning moniker.

In fact, the existence of departments and programs that included both "computer science" and "engineering" in their titles led to concerns in the late 1980s about which organization(s) should handle their accreditation. Following a 1989 directive from the Council on Postsecondary Accreditation (COPA), the directors of ABET and CSAB jointly declared that "[a] program whose title implies that it could be accredited by both ABET and CSAB must be evaluated and accredited by both agencies simultaneously" (Yeargan, 2002, p. 112). Once again, it was clear that a dense web of professional and disciplinary politics surrounded the naming of computer-oriented educational programs, regardless of their precise location or even content. And in even more pragmatic terms, this ruling imposed significant logistical and financial burdens for the

[250] The other department/unit names reported in this same survey included Computer and Information Science(s) (10 instances), Electrical Engineering and Computer Science (8), Electrical Engineering (3), Computer Science and Operations Research (2), Mathematical and Computer Sciences (2), Computing Science (2), Information and Computer Science (1), Advanced Computer Studies (1), Applied Sciences (1), and Computer Science and Electrical Engineering (1) (Gries and Marsh, 1992, p. 134).

institutions that wished to maintain accredited programs with boundary-spanning names. In fact, no less a school than MIT petitioned ABET in 1992 to retain their "Computer Science and Engineering" program while foregoing CSAB accreditation (Yeargan, 2002, p. 112). With strong encouragement from the IEEE, ABET ultimately approved MIT's "grandfathering" request, and it also extended this policy to cover a handful of similarly named programs. These developments clearly flew in the face of the ongoing efforts of reformers to promote the establishment of educational programs dedicated to "Computer Science and Engineering" or even "Computing." The discussion and debate around this issue helped prompt discussions regarding a possible merger of the ABET and CSAB, a topic to which I return below.

Conclusion

In an important sense, CC1991 looked like an important metaphorical bridge over the so-called tar pit of computer-oriented curricula. In addition to involving engineers and computer scientists and bearing the mark of the Computer Society and ACM, this report and its authors were working to bring together all phases of computer science and engineering education, especially by emphasizing an array of core common subjects, processes, and concepts. The unifying potential of CC1991 was therefore significantly premised on identifying and describing a shared body of knowledge and common set of educational approaches. Commentators recognized the significance of this strategy. As reviewer N. S. Coulter noted in late 1991, for example, "The success if Curricula 91 will depend greatly on the truth of the conjecture that the diverse field of computing has a common core" (Coulter, 1991). In support of this agenda, the CC1991 project also carried forward a discipline-building project that was based on the boundary-spanning moniker of *computing*. In fact, the discursive shift from "computer science and engineering" to "computing" was another key strategy, as it helped counter persistent suspicions that the CSE movement was much more focused on computer engineering rather than computer science.

Yet the CC1991 report could not fully escape a larger organizational field that both powerfully inflected its development and set the stage for its subsequent diffusion and uptake. In fact, the inclusion of a series of sample implementations at the end of the document revealed the extent to which CC1991 remained situated in a larger disciplinary and professional context. And as my analysis reveals, fragmentation was a dominant feature of the computer field generally –

as well as in the area of computer-oriented education specifically – through the historical period covered by this chapter. Evidence for this theme includes the persistence of multiple accreditation bodies, the continued existence of diverse types of educational departments and programs, and the ongoing use of distinct types of sociotechnical identity markers by different types of professionals.

I have also focused on the extent to which CC1991 and many prior documents were implicitly and explicitly linked to a number of foundational "axes of similarity/difference." For instance, axes such as software-hardware, science-engineering, and theory-design have powerfully inflected and informed ongoing debates over the sociotechnical boundaries of the field. In the present chapter, I also placed particular emphasis on the discipline-profession axis as providing another way to understand the persistent instabilities of the computer field, especially in the educational arena. More specifically, my analysis has documented how the dominant image of computer science as an academic discipline stands in significant conflict with the image of computer engineering as a profession. These two different outlooks or perspectives lead various actors and groups to privilege very different types of educational programs, career pathways, institutional structures, and even identity markers.

In light of this overview, the underlying assumptions on which CC1991 was built may appear deeply naïve. And indeed, my remarks in the epilogue that follows reveal something of a resurgence of fragmentation and factionalism in the computer field, especially in the late 1990s and early 2000s. Yet I also document a number of countervailing trends that suggest a continued blurring of the sociotechnical boundaries of computing, including in the educational context. Whether or not the tar pit described in this dissertation can – or even should – be bridged remains largely an open question. Further, the stakes that came with answering this question one way or another continue to loom large, not only for future generations of computer professionals, but also for future generations of computer technology. In even more general terms, these debates potently exemplify the ongoing emergence of technoscience as a dominant mode of practice in both the industrial and educational sectors, where demarcating science, technology, and engineering often looks like an increasingly futile exercise in boundary-work. By comparison, the dominant images of engineering-as-profession and science-as-discipline remain rather stubbornly entrenched.

Epilogue

Computing Curricula and Codesign: Divergent Pathways?

In the immediate wake of the Computing Curricula 1991 project, it appeared as though forces of integration were sweeping through computer field. In 1992, for example, a lengthy report titled *Computing the Future: A Broader Agenda for Computer Science and Engineering* (Hartmanis and Lin, 1992) was released by the National Research Council. As indicated by its title, this document embraced the idea that Computer Science and Engineering (CSE) was indeed a single "intellectual discipline" (pp. 19-24; pp. 213-214), and the term "computing" was also used extensively throughout the report. Pointing to key challenges in the field such as increasing demand for "more powerful and easier to use" computing technologies, as well as a continued blurring of the boundaries within and around CSE, the authors of the report made many recommendations for improving research and teaching in CSE. In fact, their plan included sustaining the field's "traditional core activities," while also broadening its "intellectual agenda" (p. 18). In summary, the authors of *Computing the Future* therefore framed CSE as a wide-ranging yet contiguous disciplinary settlement. In fact, much of the report took the disciplinary status of the field as a given. Yet as subsequent developments helped reveal, the authors' views of CSE were increasingly out of tune with the actual disciplinary milieu of the computer field.

One important piece of evidence for this theme surfaced in late 2004, when a joint task force of the ACM and IEEE Computer Society released the final draft of a report titled *Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering* (Computing Curricula for Computer Engineering Joint Task Force, 2004). As even its title suggested, this document looked like a rather direct affront to many of the prior projects and reports that had called for the development of more integrated or even unified educational programs in "computer science and engineering" or even "computing." Further, the dedication of major chapters of this report to topics such as "Computer Engineering as a Discipline" (Ch. 2)

and “Professionalism” (Ch. 6) suggested that new moves were afoot to assert that computer engineering was indeed both a distinct disciplinary and professional domain. But how do we account for this turn of events, and what is its larger significance?

To begin answering these questions it is necessary step back to 1998, when the IEEE Computer Society and ACM established a new “Computing Curricula” (CC) joint task force. According to one report, this group was originally chartered “[t]o review the Joint ACM and IEEE/CS Computing Curricula 1991 and develop a revised and enhanced version that addresses developments in computing technologies in the past decade and will sustain through the next decade” (The Joint Task Force on Computing Curricula, 2001, p. 1). As suggested by such statements – as well as frequent references to the project as “Computing Curricula 2001” (CC2001) – it initially looked like this new task force was charged with developing an updated set of recommendations for what their predecessors had dubbed the “discipline of computing.”

Yet as some of the participants admitted in a later report, “That task has proved much more daunting than we had originally realized” (p. 1). In addition to noting dramatic changes in the computing field since the release of CC1991, the authors explained that “the scope of what we call computing has broadened to the point that it is difficult to define it as a single discipline” (p. 1). Emphasizing the multi-disciplinary character of the field, the authors asserted: “[C]omputing in the 21st century encompasses many vital disciplines with their own integrity and pedagogical traditions” (p. 2). Such claims were surely debatable, but they provided the authors with an explanation and justification for a rather impressive fragmentation of CC2001 project.

In fact, entirely new committees were ultimately formed to develop five separate computing curricula reports for Computer Science (CCCS), Computer Engineering (CCCE), Information Systems (CCIS), Software Engineering (CCSE), and Information Technology (CCIT). Representatives of these groups were also tapped to develop a post-hoc overview report, with the major goal of somehow reviewing and linking five separate sets of recommendations, including by identifying associated commonalities and differences. It was also noted that the overview project might help reveal new or emergent curricular areas. As of late 2006, final curriculum reports had been released for all of the areas except for Information Technology. The overview project also remained in process, although interim drafts revealed that this group was making substantial headway.

The splintering of the CC2001 project is itself a noteworthy development, especially when viewed against the longer historical backdrop presented in this dissertation. In fact, CC2001 seemed to once again point the way toward an expanding gulf between computer designers and programmers – as well as between their tangible outputs, in the form of machine hardware and software code – thereby perpetuating a trend that had been the subject of periodic critique since at least the 1950s. On the other hand, many parts of this story are perhaps not entirely surprising, especially given my claims about the unique and somewhat unstable position of computer engineering betwixt the realms of discipline and profession. In order to highlight these themes, it is worth taking a closer look at the *Computer Engineering 2004* report.

To begin with, the authors of this document worked to both define computer engineering as a distinct academic discipline and position it with respect to other fields. This is a particularly significant development, as this was one of the most explicit and extensive efforts of this type to date. The authors' executive summary provided a brief yet rather general definition for the field:

> Computer engineering is a discipline that embodies the science and technology of design, construction, implementation, and maintenance of software and hardware components of modern computing systems and computer-controlled equipment. Computer engineering has traditionally been viewed as a combination of both computer science (CS) and electrical engineering (EE) (p. iii).

While rather wide-ranging, this statement provided an overview of the field's disciplinary settlement. Subsequent chapters expanded this description, while also marking computer engineering as a distinct disciplinary and professional domain.

In a chapter titled "Computer Engineering as a Discipline," for instance, the authors explained that computer engineering had "evolved from" the disciplines of electrical engineering and computer science, albeit often within EE programs (p. 5). While such statements hinted at the authors' lack of awareness for the deeper origins and longer trajectory of their own field, these remarks helped set up a historical review of ABET-accredited degree programs in computer engineering (p. 5). In addition to noting that 1971 marked the recognition of the first such program at Case Western Reserve University, they charted the accreditation of as many as 170 computer engineering and closely related programs. More specifically, the report indicated that 10 new Computer Engineering programs were accredited prior to 1980, 32 during the 1980s, 44 in the 1990s, and 54 from 2000 to 2004, all leading to a grand total of 140. The authors also

noted the accreditation during this same time period of a smattering of programs with related titles, such as Computer Systems Engineering (5 total), Electrical and Computer Engineering (11 total), and Computer Science and Engineering (12 total). For the sake of comparison, they added that there were about 300 accredited electrical engineering programs in the United States.

Such statistics supported the authors' claims that computer engineering was an "independent discipline" and "a discipline in it [sic] own right" (p. 37). In fact, one finds notable parallels here with the development of other fields such as computer science, where the bottom-up establishment of degree programs and departments provided crucial evidence for the claim that computer science was indeed a discipline. Further fleshing out the disciplinary settlement of computer engineering, in another chapter the authors presented a detailed map of the field's "body of knowledge" (BOK) which included a total of 18 major "knowledge areas" (Ch. 4). In fact, the same conceptual framework of the BOK appeared in all of the computing curricula reports, suggesting that much of the larger Computing Curricula project was premised on the idea that disciplinary settlements are largely and ultimately based on knowledge claims, albeit with room for extensive overlaps and interpenetrations between disciplines.

While this somewhat more nuanced image of disciplinarity seemed to provide a more realistic view of the "computing" field and its various disciplinary branches, the computer engineering report revealed the continued importance of other, non-epistemological factors, including both the expected abilities and preferred identities of the field's students and practitioners. Admitting that the distinctions between computer engineers, electrical engineers, computer scientists, and other computer professionals and technologies were often somewhat "ambiguous," the authors of the report explained that computer engineers possessed three key characteristics, namely: the ability to design computer systems, including software and hardware; the possession of a breadth of knowledge of mathematics and engineering sciences; and a preparation for "professional practice in engineering" (p. 5). These three characteristics nicely captured how the authors strategically framed computer engineering as not only a distinct academic discipline, but also an unambiguous part of the engineering profession. In fact, it was reasonably clear that engineering education was the only widely accepted pathway by which students could be introduced to the engineering sciences, schooled in the basics of engineering design, and finally ushered into the fold of the engineering profession.

If there remained any doubts about these underlying assumptions, the rest of the report largely erased them. The authors repeatedly claimed, for example, that the "ability to design" was a key aspect of computer engineering. They also noted that "[p]rofessionalism should be a constant theme that pervades the entire curriculum" (p. 8), and they even devoted a chapter of the report to this topic (Ch. 6). And finally, the report placed significant emphasis on the importance of accreditation processes and criteria in ongoing efforts to develop computer engineering curricula and programs. As the authors explained, "The computer engineering core [body of knowledge] acknowledges that engineering curricula are often subject to accreditation, licensure, or governmental constraints" (p. 10). For better or worse, such statements help perpetuate the image of computer engineering as an engineering discipline to its "core."

Software/Hardware Codesign: Blurring the Sociotechnical Boundaries

Even as the authors of the *Computer Engineering 2004* report promoted a vision of their field as both an academic discipline and unambiguous branch of the engineering profession, many of the developments cited in prior chapters call into question many of the underlying technical and espistemological justifications for these types of boundary-work. In fact, my application of the "disciplinary settlement" concept to the body of knowledge outlined in this same report reveals the extent to which interpenetrating and overlapping knowledge claims between computer engineering and adjacent fields is a taken-for-granted reality. Further, in recent decades it has become increasingly common to find practicing and prospective computer engineers and computer scientists working side-by-side, in contexts ranging from university classrooms and labs to private-sector offices and research facilities. This trend also seems to continue apace, despite the partially distinct disciplinary backgrounds and professional identities maintained by the major actors and groups in question. I claim that one important enabling factor that helps us account for this trend centers on the continued blurring of the boundaries between "hardware" and "software," a topic that runs through large parts of this dissertation. In order to both bring this theme into relief and point the way toward some possible reform movements, I turn to a brief history of the "software/hardware codesign" movement.

As noted in preceding chapters, comments about the ultimately nebulous character of the boundary between computer hardware and software – or "machine" and "code" – can be traced back to the 1940s and 1950s. In fact, by the 1970s this blurred view of the software-hardware

relations was almost axiomatic for many commentators, as reflected in period textbooks such as *Structured Computer Organization* by well-known computer scientist Andrew Tanenbaum (1976). In an introductory chapter, the author summarized the historical development of computer organization and architecture, leading him to cleverly note that “one man's hardware is another man's software” (p. 11). He also went on to describe the boundary between computer software and hardware boundary as “arbitrary and constantly changing” (p. 11).

On the one hand, one detects in such remarks a strong resonance with the writings of a host of prior commentators, ranging from Mauchly and Hopper to Carr and Gorn. On the other hand, Tanenbaum’s comments tended to hide some of the thorny practical realities that came with the actual practice of computer software and hardware design. For instance, differently trained professionals were clearly involved in different aspects of computer system design, and they brought with them their own partially unique tools, techniques, and cultures of design. And indeed, these types of distinctions were even more evident when one moved from intermediate levels of computer design to the opposite ends of the spectrum, where the design of the electronic components of computer “hardware” seemed quite distinct from the development of operating systems and programming tools, much less end-user applications.

Nevertheless, the mid-1970s were a time when a handful of forward-looking researchers were tentatively seeking out and developing the appropriate tools and techniques that would allow them to better grapple with design trade-offs that surfaced at the intersection of software and hardware. In a 1975 paper, for example, C. W. Rose and M. Albarran noted the long-standing tendency for the design of hardware and software to proceed “quite differently and separately” (p. 421). Yet they noted two major trends that were upsetting the status quo. The first of these centered on new technological developments that were providing system designers with the ability to implement a wide variety of functions “in either hardware, software, or a combination of both” (p. 421). The second trend – which the authors described as more “philosophical” in nature – centered on new design methods that were systematic, hierarchical, and capable of dealing with multiple levels of abstraction using a common design language.

As suggested by these remarks, changes in both technology and design culture were pointing to new possibilities in the area of computer system design. But these authors also noted a number of difficult limitations with existing software and hardware design tools, including a lack of suitable hardware description languages (HDLs) and an overall inflexibility with regard

to determining the boundaries between software and hardware. As an alternative approach, they called for the development of a "computer system description language" that could accommodate the hierarchical description of hardware *and* software. Their own LOGOS design environment was offered as a tentative step toward a system that could help automate the design of "hierarchical, integrated hardware/software systems" (p. 429).

The LOGOS project looked like an important step toward a more integrated approach to computer systems design, its immediate impact appears to have been minimal. As suggested by other commentators, the dominance of "layered" and "hierarchical" models of computer systems architecture through the 1970s may have hampered these alternative approaches. In fact, the dominant culture of computer design at the time tended to either insulate hardware and software specialists from one another or position engineers as the vanguards of computer design decisions. It would take roughly a decade before some of these barriers started to fall, especially as evidenced through the emergence and development of the "codesign" movement.

While the phrase "software/hardware codesign" can be traced back to at least 1985, the concept gained significant momentum through the 1990s. And by some measures, co-design has attracted a great deal of attention in a relatively short span of time. Many hundreds of papers on codesign have been published since the early 1990s, and a search of the ACM and IEEE archives reveals increasing interest in the topic. In addition, the First International Conference on Hardware-Software Codesign was held in 1992, and international workshops and conferences dedicated to this topic have been held annually since 1996. Published proceedings reveal upward trends in the size and scope of these events.

In most general terms, a central tenet of the co-design approach to developing specialized computing devices centers on the idea that the design of such systems must start with no *a priori* boundaries around the software and hardware components. A 1991 article on the topic provides one early and rather succinct description of the how and why of codesign. Authors Franke and Purvis start this piece with a brief review of historically dominant approaches to computer design. "Computer systems development has been ordinarily characterized," they explained, "by the notion that hardware engineers supply general-purpose computing systems, which are then programmed by software engineers" (p. 344). The authors pointed to the relative independence of software and hardware development activities under this model, and they suggested that

"layered" or "hierarchical" models of computer architecture create an environment in which software specialists can avoid grappling with "low-level" hardware concerns.

Franke and Purvis went on to discuss how a number of important technological developments were making it more feasible and desirable to call into question the boundaries around hardware and software engineering. And as suggested by the title of their paper, the authors proposed an alternative approach to computer design that "combine[s] the hardware and software perspectives from the earliest stages of the design process" (p. 344). In a more recent paper, computer scientists Micaela Serra and William Gardner offer a concise summary of four key characteristics of the co-design approach: "the cooperative design of hardware and software components; the unification of currently separate hardware and software paths; the movement of functionality between hardware and software; the meeting of system-level objectives by exploiting the synergism of hardware and software through their concurrent design" (1998, p. 1).

These and other codesign proponents point to a number of advantages of this model, many with appeal to the profit-motivated private sector. In terms of the design process, for example, codesign promises to significantly streamline the coordination of large system design projects, leading to reductions in development time and cost. Many others argue that codesign can lead to the development of "better" technologies, at least in terms of technical metrics like performance, reliability, and/or flexibility.

As this overview reveals, hardware/software codesign shares much in common with earlier historical movements. Hence, one might wonder why something like codesign failed to gain traction earlier. After all, this dissertation reveals that interchangeability of "hardware" and "programs" was recognized in as early as the 1950s, and the topic of hardware/software equivalence has resurfaced regularly for decades. But during the 1980s, a unique confluence of trends helped enable the emergence of a more recognizable movement of co-design proponents and practitioners. On rare occasion, these individuals even note that their work is part of a longer tradition. In an introduction to the proceedings of the Fifth International Conference on Hardware/Software Codesign (CODES/CASHE '97), two workshop chairs explained that "[d]esigners have practiced co-design since the first microprocessors were used for implementing digital control" (Ernst and Borriello, 1997).

Yet many codesign commentators and proponents are quick to follow the typical outlook of technologists when they point to the numerous technical trends that helped set the stage for their work. Some relevant and oft-cited developments include:

- the increasing diversity, complexity, and number of embedded systems in use or development;
- the growth of hardware development languages, especially in tandem with Very Large Scale Integration (VLSI) technologies for integrated circuits;
- the development of CAD systems that support the simultaneous design of both software and hardware; and
- the emergence of new types of programmable chips, including application-specific integrated circuits (ASICs).

Yet we might be wary of overemphasizing technological factors as we account for the rise of co-design. As many scholars have taught us, such expressions of technological determinism are both commonplace and easy to debunk. Indeed, the preceding overview of the pioneering work by Rose and Albarran suggests that, by at least the mid-1970s, some of the pivotal technological and "philosophical" developments were starting to lead toward new design approaches, and these had much in common with codesign.

Further, one cannot help but notice other trends that were afoot around the time that the codesign movement really started to take off. The "computing as a discipline" movement described in the preceding chapter, for example, was gaining momentum around this same time. Hence, it is highly plausible that the CC1991 effort both reflected and reinforced a culture of "integration" in the computer field, thereby helping to enable the emergence and growth of the codesign movement. This thesis is further supported by the observation that many codesign researchers maintained university affiliations or appointments. Perhaps one of Carr's early assertions rings true – namely that universities do indeed play crucial roles in stimulating research and development activities at the cutting-edge of computing.

As Serra and Gardner's efforts reveal, introducing co-design to future generations of computer science and computer engineering students remains something of an experiment. As they admit, "different design cultures hamper integration," and their own agenda is framed in terms of developing a more "appropriate curriculum" for computer science students. Yet the benefits of such reforms are increasingly clear. They explain, for instance, that the

*inter*disciplinary linking of computer science and engineering – as well as *intra*disciplinary explorations within computer science – were novel features and a "source of great strength" in their own codesign course (p. 8). Emphasizing the extent to which hardware/software co-design challenges insular computing curricula, they add that "hardware related topics were tremendously empowering to the mainly software students in Comp. Science, who found the demistification [sic] of the whole area of VLSI design and CAD software useful to their breadth" (p. 8).

These initial results are certainly encouraging, but recent currents in the educational sphere reveal that proponents of codesign may continue to face formidable challenges, especially as they work to introduce these alternative design approaches into computer science, computer engineering, and related curricula. On the other hand, my analysis suggests that the codesign movement itself may point the way toward a more integrated "discipline of computing," both in educational contexts and beyond.

From Software/Hardware Codesign to Sociotechnical Codesign

Significant barriers must be overcome before codesign methods move to the forefront of the computing curricula specifically and the computer field generally. In fact, my analysis suggests that the most recent Computing Curricula project and the codesign movement increasingly look like two alternative pathways for the future of computing, with the former preserving sociotechnical fragmentation and factionalism and the latter tending toward greater integrating and unification. And while my analysis frames codesign as a promising development that may help in the realization of a reform movement with deep historical roots, I close with an even more ambitious vision for the future.

As the preceding overview makes clear, software/hardware codesign remains significantly focused on technology, both in terms of its associated enabling factors and anticipated outcomes. However, I contend that the boundary-blurring characteristics of codesign can provide inspiration for other important types of reform. In their 1991 paper, codesign proponents Franke and Purvis cite a 1986 article in which well-known computer researcher Elliott Organick described the emergence of a new breed of "heterosystems" engineers. Skilled in working with large systems comprised of "diverse, interacting components," Organick explained that these designers "are no longer just software engineers or just hardware engineers"

(quoted in Franke and Purvis, 1991, p. 347). Franke and Purvis tentatively worked in similar directions when they note that computer system professionals are increasingly involved in the design of "reactive" systems that involve hardware, software, and "users and objects from the real world" (p. 346).

Some may notice parallels here with contemporary research in a variety of fields, including Science and Technology Studies (STS). As described by Donald MacKenzie, for example, successfully developing new technologies often requires heterogeneous engineering, or "the engineering of the social as well as the physical world" (1990, p. 28). Still others may recognize that terms such as "codesign" and "meta-design" have been used to describe new approaches to technological design that center on open and extensible systems and active efforts to blur the boundaries between the designers and users of various technologies. Hence, by challenging the drawing of *a priori* boundaries around hardware and software, codesign methods can inspire us to call into question other boundaries, such as those that divide computer scientists from computer engineers, or those that separate computer technologies from users, applications, and even society.

The history of the software-hardware boundary presented in this dissertation forcefully reveals the extent to which the social and technical are deeply intertwined. It is hoped that ongoing moves to put back together the Humpty and Dumpty of software and hardware may also point us toward a more thoroughly contextualized, reflexive, and socially responsible culture of computer design and use. Doing so, however, will require that computer experts from a variety of backgrounds acknowledge and critically engage other axes of similarity/different that have played a profoundly influential role in the computing field. In fact, preceding chapters reveal that challenging the boundaries around software and hardware can quickly raise thorny questions about the relation of science and engineering, as well as the respective dominant images of disciplines and professions.

Appendix A

Acronyms and Abbreviations

ABET	Accreditation Board for Engineering and Technology
AC	Alternating Current
ACM	Association for Computing Machinery
AFIPS	American Federation of Information Processing Societies
AIEE	American Institute of Electrical Engineers
ASCC	Automatic Sequence Controlled Calculator
ASIC	Application-Specific Integrated Circuit
ASEE	American Society for Engineering Education
BOK	Body of Knowledge
C^3S	Curriculum Committee on Computer Science (of the ACM)
CACM	*Communications of the ACM*
CAD	Computer-Aided Design
CC1991	Computing Curricula 1991
CCP	Certificate in Computer Programming
CDC	Computing Devices Committee (of the AIEE)
CE	Computer Engineering
COPA	Council on Postsecondary Accreditation
COSERS	Computer Science and Engineering Research Study
COSINE	Computer Science(s) in Electrical Engineering
CS	Computer Science
CSAB	Computing Sciences Accreditation Board
CSAC	Computing Sciences Accreditation Committee
CSE	Computer Science and Engineering
CSEB	Computer Science and Engineering Board (of the NAS)
CUPM	Committee on the Undergraduate Program in Mathematics (of the MAA)
DC	Direct Current
DISE	Digital Systems Education
DPMA	Data Processing Management Association
EAB	Educational Activities Board (of the IEEE)
ECE	Electrical and Computer Engineering
ECPD	Engineers' Council for Professional Development
EDSAC	Electronic Delay Storage Automatic Calculator
EDVAC	Electronic Discrete Variable Automatic Calculator
EE	Electrical Engineering
EECS	Electrical Engineering and Computer Science
EJCC	Eastern Joint Computer Conference

FJCC	Fall Joint Computer Conference
ENIAC	Electrical Numerical Integrator And Calculator
HCI	Human-Computer Interaction
HDL	Hardware Description Language
IAS	Institute for Advanced Study (at Princeton University)
IBM	International Business Machines
ICCP	Institute for the Certification of Computer Professionals
IEEE	Institute of Electrical and Electronics Engineers
IEEE CS	Institute of Electrical and Electronics Engineers – Computer Society
IFIP	International Federation for Information Processing
IRE	Institute of Radio Engineers
IS	Information Systems
IT	Information Technology
ITG	Institute Technical Group
JCC	Joint Computer Committee
JCC	Joint Computer Conference
JEE	*Journal of Engineering Education*
LoC	Library of Congress
LSI	Large-Scale Integration
MAA	Mathematical Association of America
MANIAC	Mathematical Analyzer Numerical Integrator and Computer
MIT	Massachusetts Institute of Technology
MSI	Medium-Scale Integration
NAE	National Academy of Engineering
NAS	National Academy of Sciences
NATO	North Atlantic Treaty Organisation
NBS	National Bureau of Standards
NCR	National Cash Register
NJCC	National Joint Computer Committee
NJCC	National Joint Computer Conference
NMAA	National Machine Accountants Association
NSF	National Science Foundation
ONR	Office of Naval Research
PGEC	Professional Group on Electronic Computers (of the IRE)
PGIPS	Professional Group on Information Processing Systems
PTGEC	Professional Technical Group on Electronic Computers (of the IRE)
RCA	Radio Corporation of America
SDC	System Development Corporation
SIAM	Society for Industrial and Applied Mathematics
SIC	Special Interest Committee
SICARCH	Special Interest Committee on Computer Architecture (of the ACM)
SICSOFT	Special Interest Committee on Software Engineering (of the ACM)
SIG	Special Interest Group
SIGCSE	Special Interest Group on Computer Science Education (of the ACM)
SIGMICRO	Special Interest Group on Microprogramming (of the ACM)
SIGSOFT	Special Interest Group on Software Engineering (of the ACM)

SJCC	Spring Joint Computer Conference
STS	Science and Technology Studies
TC	Technical Committee
TCCA	Technical Committee on Computer Architecture (of the IEEE Computer Society)
TCSE	Technical Committee on Software Engineering (of the IEEE Computer Society)
TIC	Technical Interest Council
UCLA	University of California – Los Angeles
UNESCO	United Nations Educational, Scientific, and Cultural Organization
UNIVAC	Universal Automatic Computer
VLSI	Very Large-Scale Integration
WJCC	Western Joint Computer Conference

Bibliography

"1946 National Electronics Conference." (1946, September). *Proceedings of the IRE*, 34(9): 665-667.

"1947 IRE National Convention." (1947, May). *Proceedings of the IRE*, 35(5): 499-503.

"1948 IRE National Convention Program – Summaries of Technical Papers." (1948, March). *Proceedings of the IRE*, 36(3): 365-380.

"1949 IRE National Convention Program – Summaries of Technical Papers." (1949, February). *Proceedings of the IRE*, 37(2): 160-178.

"1949 Engineering Developments – Reviewed by AIEE Technical Committees." (1950, January). *Electrical Engineering*, 69(1): 1-11, 24-25.

"1993 OECE Recipients – Clarence L. Coates." (1993, November). Retrieved on April 4, 2006 from Purdue School of ECE web site: https://engineering.purdue.edu/ECE/People/Alumni/OECE/1993/coates.whtml

"A Datamation Staff Survey: Computer Components '61." (1961, August). *Datamation*, 7(8): 36-40.

"A Matter of Degrees." (1965, June). *Datamation*, 11(6): 23.

Abbott, Andrew. (1988). *The System of Professions: An Essay on the Division of Expert Labor*. Chicago: University of Chicago Press.

Abbott, Andrew. (2001). *Chaos of Disciplines*. Chicago and London: The University of Chicago Press.

ABET (Accreditation Board for Engineering and Technology). (1980). *Forty-eighth Annual Report (1979/1980)*. New York, NY: ABET.

ABET (Accreditation Board for Engineering and Technology). (1982). *Fiftieth Annual Report (1981/1982)*. New York, NY: ABET.

ABET (Accreditation Board for Engineering and Technology). (1985). *Fifty-third Annual Report (1984/1985)*. New York, NY: ABET.

ABET (Accreditation Board for Engineering and Technology). (1989). *Fifty-seventh Annual Report (1988/1989)*. New York, NY: ABET.

ABET (Accreditation Board for Engineering and Technology). (1990). *Fifty-eighth Annual Report (1989/1990)*. New York, NY: ABET.

Abrahams, Paul. (1987, November). "A Farewell to NCC." *Communications of the ACM*, 30(11): 899.

ACM Accreditation Committee. (1977, November). "Accreditation Guidelines for Bachelor's Degree Programs in Computer Science." *Communications of the ACM*, 20(11): 891-892.

ACM C^3S (Curriculum Committee on Computer Science). (1965, September). "An Undergraduate Program in Computer Science – Preliminary Recommendations." *Communications of the ACM*, 8(9): 543-552.

ACM C^3S (Curriculum Committee on Computer Science). (1968, March). "Curriculum 68: Recommendations for Academic Programs in Computer Science." *Communications of the ACM*, 11(3): 151-197.

"ACM: Association for Computing Machinery, the world's first educational and scientific computing society." (n.d.) Retrieved April 13, 2006 from http://www.acm.org/

"ACM and IEEE-CS Launch Fall Joint Computer Conference." (1987, January). *Computer*, 20(1): 117.

"ACM Special Interest Group for Mathematical Programming." (1961, September). *Communications of the ACM*, 4(9): 368. "ACM Special Interest Group for Mathematical Programming." (1961, September). *Communications of the ACM*, 4(9): 368.

Acton, Forman S. (1957). "Supply and Demand in Computational Mathematics." In Preston C. Hammer (Ed.), *The Computing Laboratory in the University* (121-125). Madison, WI: The University of Wisconsin Press.

Ad Hoc Group, AIEE Computing Devices Committee. (1963, April). "Developments and Trends in Computing Devices During 1962." *Electrical Engineering*, 82(4): 269-274.

Ad Hoc Group, AIEE Computing Devices Committee. (1963, May). "A Summary of Recent Advances in the Computer Field." *Computers and Automation*, 12(5): 32-40.

Adams, Charles W. (1957). "The Contribution of the Computing Laboratory to the University Curriculum." In Preston C. Hammer (Ed.), *The Computing Laboratory in the University* (139-143). Madison, WI: The University of Wisconsin Press.

"Affiliate Status." (1960, June). *IRE Transactions on Electronic Computers*, EC-9(2): Back cover.

"AFIPS Appoints Public Affairs Directors." (1962, August). *Communications of the ACM*, 5(8): 425.

AFIPS Taxonomy Committee. (1980). *Taxonomy of Computer Science and Engineering*. Arlington, VA: American Federation of Information Processing Societies, Inc.

"AIEE Forms Committee on Computing Devices." (1948, March). *Electrical Engineering*, 67(3): 271.

"AIEE Power Industry Computer Application Conference." (1958, September). *Electrical Engineering*, 77(9): 848-849.

"AIEE Officers and Committees for 1946-47." (1946, September). *Transactions of the AIEE*, 65(9): 1217-1228.

"AIEE Officers and Committees for 1948-49." (1948, September). *Transactions of the AIEE*, 67(9): 1784-1798.

"AIEE Officers and Committees for 1949-50." (1949, September). *Transactions of the AIEE*, 68(9): 799-811.

"AIEE Officers and Committees for 1954-55." (1954, September). *Electrical Engineering*, 73(9): 834-860.

"AIEE Officers and Committees for 1955-56." (1955, September). *Electrical Engineering*, 74(9): 837-850.

"AIEE Officers and Committees for 1956-57." (1956, September). *Electrical Engineering*, 75(9): 841-856.

"AIEE Officers, Departments, and Committees for 1957-1958." (1957, September). *Electrical Engineering*, 76(9): 832-850.

"AIEE Officers, Departments, and Committees for 1958-1959." (1958, September). *Electrical Engineering*, 77(9): 870-888.

"AIEE Technical Subcommittees, 1950-1951." (1950, September). *Electrical Engineering*, 69(9): 937-944.

"AIEE Winter General Meeting, New York, NY, February 2-7, 1958 (Tentative Technical Program)." (1958, January). *Electrical Engineering*, 77(1): pp. 71-80.

Aiken, Howard H., and Grace M. Hopper. (1946). “The Automatic Sequence Controlled Calculator.” *Electrical Engineering*, 65: 384-391, 449--454, 522-528.

Aiken, Howard H. (1951). [Opening Address.] In *Proceedings of a Second Symposium on Large-Scale Digital Calculating Machinery*, The Annals of the Computation Laboratory at Harvard University, Volume XXVI, Harvard University Computation Laboratory, September 13-16, 1949. Cambridge, MA: Harvard University Press.

Akera, Atsushi. (1998). *Calculating a Natural World: Scientists, Engineers, and Computers in the United States, 1937-1968*. Unpublished Dissertation. The University of Pennsylvania.

Akera, Atsushi. (2002). “The Early Computers.” In Atsushi Akera and Frederik Nebeker (Eds.), *From 0 to 1: An Authoritative History of Modern Computing* (63-75). Oxford, England and New York, NY: Oxford University Press.

Akera, Atsushi. (2004a). “The Circulation of Knowledge and Disciplinary Formation: Modern Computing as an Ecology of Knowledge.” Conference paper presented at 3Societies Conference, Halifax, NS, August 2004. Retrieved October 9, 2006 from http://www.rpi.edu/%7eakeraa/3Soc-Paper.doc

Akera, Atsushi. (2004b). “Peripatetic Careers, Institutional Ecologies, and the Multiple Foundations of New Technology: John W. Mauchly and the Origin for the Digital Electronic Computer.” Conference paper presented at SHOT 2004 Annual Meeting, Amsterdam, Netherlands, October 6-9, 2004. Retrieved October 9, 2006 from http://www.rpi.edu/%7eakeraa/SHOT-Paper.doc

Akera, Atsushi. (2006). *Calculating a Natural World: Scientists, Engineers, and Computers During the Rise of U.S. Cold War Research.* Cambridge, MA and London, England: The MIT Press.

Akera, Atsushi, and Frederik Nebeker. (2002). *From 0 to 1: An Authoritative History of Modern Computing*. New York: Oxford University Press.

Alt, Franz L. (1952). “Forward.” *Proceedings of the Association for Computing Machinery*. Pittsburgh, PA, May 2-3, 1952.

Alt, Franz L. (1958). *Electronic Digital Computers: Their Use in Science and Engineering.* New York, NY and London: Academic Press.

Alt, Franz L. (1962, June). “Fifteen Years ACM.” *Communications of the ACM*, 5(6): 300-307.

Alt, Franz L. (1965, January). “Some Unorthodox Predictions.” *Computers and Automation*, 14(1): 11-12.

“Alumni: Obituaries.” (2004, July-August). *The Pennsylvania Gazette.* Retrieved November 7, 2006 from http://www.upenn.edu/gazette/0704/0704obits.html

Amdahl, Lowell. (1965, November). “Gothic Computer Architecture: A Guest Editorial.” *Datamation*, 11(11): 23.

American Society for Engineering Education. (1958, October). “Report on the Engineering Sciences, 1956-1958.” *Journal of Engineering Education*, 49(1): 36.

Anderson, Walter L. (1962, June). “The Chairman’s Letter.” *IRE Transactions on Electronic Computers*, EC-11(3): 441.

Anderson, Walter L. (1963a, February). “The Chairman’s Letter.” *IEEE Transactions on Electronic Computers*, EC-12(1): 55.

Anderson, Walter L. (1963b, April). “The Chairman’s Letter.” *IEEE Transactions on Electronic Computers*, EC-12(2): 176.

Anderson, Walter L. (1963c, June). “The Chairman’s Letter.” *IEEE Transactions on Electronic Computers*, EC-12(3): 352.

Anderson, Walter L. (1964a, February). “The Chairman’s Letter.” *IEEE Transactions on Electronic Computers*, EC-13(1): 81.

Anderson, Walter L. (1964b, April). “Chairman’s Newsletter.” *IEEE Transactions on Electronic Computers*, EC-13(2): 180.

Anderson, Walter L. (1976, December). “The Middle Years.” *Computer*, 9(12): 45-53.

“Announcing a Major New Publication in the Field of Computer Science: IEEE Transactions on Software Engineering.” (1975, February). *Computer*, 8(2): 82.

“Anthony Oettinger’s Home Page.” (1998, April 31). Retrieved October 23, 2006 from http://people.deas.harvard.edu/users/faculty/Anthony_Oettinger/Anthony_Oettinger.html

Arden, Bruce W. (1976, December). “The Computer Science and Engineering Research Study (COSERS).” *Communications of the ACM*, 19(12): 670-673.

Arden, Bruce W. (Ed.). (1980). *What Can Be Automated: The Computer Science and Engineering Research Study (COSERS).* Cambridge, MA and London, England: The MIT Press.

Armer, Paul. (1959, January). [Letter to the Editor]. *Communications of the ACM*, 2(1): 2-4.

Armer, Paul, Morton M. Astrahan, Isaac L. Auerbach, Walter M. Carlson, Arnold A. Cohen, Margaret R. Fox, Claude A.R. Kagan, Morris Rubinoff, Jack Sherman, and Willis H. Ware. (1986, July-September). “Reflections on a Quarter-Century: AFIPS Founders.” *IEEE Annals of the History of Computing*, 8(3): 225-256.

Aspray, William. (1985). “Introduction.” In *Proceedings of a Symposium on Large-Scale Digital Calculating Machinery*, Harvard University, January 7-10, 1947 (ix-xxiii). Cambridge, MA and London, England: MIT Press and Los Angeles, CA and San Francisco, CA: Tomash Publishers.

Aspray, William. (1993, April). “Edwin L. Harder and the Anacom: Analog Computing at Westinghouse.” *IEEE Annals of the History of Computing*, 15(2): 35-52.

Aspray, William. (2000, July-September). “Was Early Entry a Competitive Advantage: U.S. Universities That Entered Computing in the 1940s.” *IEEE Annals of the History of Computing*, 22(3): 42-87.

Astrahan, Morton M. (1976, December). “In the Beginning there was the IRE Professional Group on Electronic Computers.” *Computer*, 9(12): 43-44.

Atchison, William F. (1960, June). “Numerical Analysis and Computers in Engineering Education.” *Journal of Engineering Education*, 50(10): 856-859.

Atchison, William F., and John W. Hamblen. (1964, April). “Status of Computer Sciences Curricula in Colleges and Universities.” *Communications of the ACM*, 7(4): 225-227.

Atchison, William F. (1968). “The Position of Computing Science in the University Structure: A Report of the Workshop.” In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (169-175). New York and London: Academic Press.

Atchison, William F. (1971). “Computer Science as a New Discipline.” *International Journal of Electrical Engineering Education*, 9: 130-135.

Atchison, William F. (1985). “The Development of Computer Science Education.” In M. C. Yovitz (Ed.), *Advances in Computers* (319-377). New York, NY: Academic Press.

Auerbach, Isaac L. (1986a, April). “The Start of IFIP – Personal Recollections.” *Annals of the History of Computing*, 8(2): 180-192.

Auerbach, Isaac L. (1986b, July/September). “Harry H. Goode, June 30, 1909-October 30, 1960.” *Annals of the History of Computing*, 8(3): 257-260.

Autonetics, a Division of North American Aviation, Inc. (1956, February). “Engineers and Scientists… Help us solve today’s most advanced problems.” *Electrical Engineering*, 75(2): 91A.

Bagley, Philip R. (1959, May). [Letter to the Editor]. *Communications of the ACM*, 2(5): 3-4.

Baker, W. R. G. (1957, June). “The IRE ‘Affiliate’ Plan – A New Venture in Engineering Society Structure and Service.” *IRE Transactions on Electronic Computers*, EC-6(2): 71.

Baldwin, Carliss Y. and Kim B. Clark. (2000). *Design Rules, Vol.1: The Power of Modularity*. Cambridge, MA: The MIT Press.

Barnard, G. A. (1960, May/June). “1960 WJCC: A Look Back.” *Datamation*, 6(3): 23; 52.

Beckman, Frank S. (1968). “Graduate Computer Science Program at American Universities.” In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (39-59). New York and London: Academic Press.

Bendix Aviation Corporation. (1954, November). “Bendix – Senior Electrical Engineer.” *Electrical Engineering*, 73(11): 70A.

Bendix Aviation Corporation. (1955a, August). “Analog Computer Engineers.” *Computers and Automation*, 4(8): 32.

Bendix Aviation Corporation. (1955b, November). “Engineers – Permanant, Creative Opportunities for Electrical Engineers at Bendix.” *Electrical Engineering*, 74(11): p. 78A.

Bendix Aviation Corporation. (1956, January). “Engineers – Permanant, Creative Opportunities for Electrical Engineers at Bendix.” *Electrical Engineering*, 75(1): p. 70A.

Bennett, Arnold A. (1955). “The Impact of Automatic Computing Machines Upon the Undergraduate Curriculum.” In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (40-46). Detroit, MI: Wayne University Press.

Bergstein, Harold. (1962, July). “A Profile of No. 1” [Interview with IBM’s Warren C. Hume and A. L. Harmon]. *Datamation*, 8(7): 33-37.

Bonn, Ted. (1982, October). “A Second Division Director for the IEEE Computer Society.” *Computer*, 15(10): 4-5.

Booth, Taylor L. (1982, July). “Current Activities of the Educational Activities Board.” *Computer*, 15(7): 4-5.

Booth, Taylor L. (1984, October). “Computer Education.” *Computer*, 17(10): 57-68.

Booth, Taylor L. and Raymond E. Miller. (1987, May). “Computer Science Program Accreditation: The First-Year Activities of the Computing Sciences Accreditation Board.” *Communications of the ACM*, 30(5): 376-388.

Booth, Taylor L., C. Gordon Bell, Cecil H. Hoker, Robert M. Glorioso, Edward J. McCluskey, Frederic J. Mowle, David M. Robinson. (1973, January). "Minicomputers in the Digital Laboratory Program." *Computer*, 6(1): 28-42.

Booth, Taylor L., Tom Brubaker, Tom Cain, Ron Danielson, Ron Hoelzeman, Glen Langdon, Dave Soldan, and Muali Varanasi. (1986, June). "Design Education in Computer Science and Engineering." *Computer*, 19(6): 20-27.

Brainerd, John G. and T. K. Sharpless. (1948, February). "The ENIAC." *Electrical Engineering*, 67(2): 163-172.

Brainerd, John G. (1955). "Keynote Address." *Proceedings of the Eastern Joint Computer Conference*, Boston, MA, November 7-9, 1955 (pp. 6-7). New York, NY: Institute of Radio Engineers.

Brainerd, John G. (1960, June). "Setting up a Computing Faculty in a School of Engineering." *Journal of Engineering Education*, 50(10): 846-851.

Brandin, David H. (1982a, November). "ACM President's Letter: The State of the ACM – 1982." *Communications of the ACM*, 25(11): 769-770.

Brandin, David H. (1982b, November). "By invitation – a message from the ACM President: the problems of technology transfer." *Computer*, 15(11): 4.

Brandin, David H. and Oscar N. Garcia. (1983, August). "Where do parallel lines meet? or The common goals of ACM and the IEEE-CS." *Computer*, 16(8): 6-7.

Breslau, Daniel. (2000, July). "Sociology after Humanism: A Lesson from Contemporary Science Studies." *Sociological Theory*, 18(2): 289-307.

Burks, Arthur W., Herman H. Goldstine, and John von Neumann. (1989). "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument." In Zenon W. Pylyshyn and Liam J. Bannon (Eds.), *Perspectives on the Computer Revolution* (39-48). Norwood, NJ: Ablex Publishing Corp. (Originally published in 1946.)

Browne, James C. and John J. Howard, Jr. (1973, November). "The Interaction of Operating Systems and Computer Architecture: A Workshop Survey." *Computer*, 6(11): 16-17.

Buchholz, Werner. (1953, October). "The Computer Issue." *Proceedings of the IRE*, 41(10): 1220-1222.

Burroughs Corporation. (1957, May). "That Certain Man." *Computers and Automation*, 6(5): 43.

Cain, James T. (1975, September). "Report of the Digital Systems Education Committee." *ACM SIGCSE Bulletin*, 7(3): 13-16.

Cain, J. T., and R. G. Hoelzeman. (1977). “DISE Project.” In *Proceedings of the Computer Science and Engineering Workshop*, Williamsburg, Virginia, June 6-7, 1977 (145-146). Long Beach, CA: IEEE Computer Society.

Cain, J. T., G. G. Langdon, Jr., and M. R. Varanasi. (1983). “Foreward.” In *The 1983 IEEE Computer Society Model Program in Computer Science and Engineering* (iii-vi). Silver Spring, MD: IEEE Computer Society Press.

Cain, J. T., G. G. Langdon, Jr., and M. R. Varanasi. (1984, April). “The IEEE Computer Society Model Program in Computer Science and Engineering.” *Computer*, 17(4): 8-17.

“Call for Papers: 17th IMACS World Congress, Budapest, August 25-29, 2003.” (n.d.). Retrieved November 7, 2006 from http://www.ifors.org/panorama/conferences/conf_02_03.html

“Call for Participation: A Workshop on the Engineering of VLSI and of Software.” (1982, April). *Computer*, 15(4): 135.

Callon, Michel. (1999). “Some Elements of a Sociology of Translation: Domestication of the Scallops and the Fishermen of St. Brieuc Bay.” In Mario Biagioli (Ed.), *The Science Studies Reader* (67-83). New York and London: Rutledge. (Original work published 1986, abridged 1998)

Campbell-Kelly, Martin, and Michael R. Williams (Eds.). (1985). *The Moore School Lectures: Theory and Techniques for Design of Electronic Digital Computers*. Cambridge, MA and London, England: The MIT Press/

Campbell-Kelly, Martin, and William Asprey. (1997). *Computer: A History of the Information Machine*. New York: Basic Books.

Campbell-Kelly, Martin. (2000). “Past into Present: The EDSAC Simulator.” In Raúl Rojas and Ulf Hashagen (Eds.), *The First Computers: History and Architectures* (397-416). Cambridge, MA and London, England: The MIT Press.

Carlson, Walter M. (1969, October). “‘There is a tide in the affairs of men…’ (Letter from the ACM Vice-President).” *Communications of the ACM*, 12(10): 537.

Carlson, Walter M. (1970a, September). “Finding the Real Expert (ACM President’s Letter).” *Communications of the ACM*, 13(9): 525.

Carlson, Walter M. (1970b, October). “Rx for Excellence: Better Education (ACM President’s Letter).” *Communications of the ACM*, 13(10): 587.

Carlson, Walter M. and Dick B. Simmons. (1984, March). “Intersociety Cooperation.” *Computer*, 17(3): 88-89.

Carr, John W., III. (1953, November). “Who Will Man the New Digital Computers?” *Computers and Automation,* 2(8): 1-3.

Carr, John W., III. (1956). “Conference Summary.” *Proceedings of the Eastern Joint Computer Conference*, New York, NY, December 10-12, 1956 (147-150). New York, NY: American Institute of Electrical Engineers.

Carr, John W., III. (1952, February). “Discussion.” In *Review of Electronic Digital Computers – Joint AIEE-IRE Computer Conference*, Philadelphia, PA, December 10-12, 1951 (113-114). New York, NY: American Institute of Electrical Engineers.

Carr, John W., III. (1957, January). “Inaugural Presidential Address.” *Journal of the ACM*, 4(1): 5-7.

Carr, John W., III. (1962a, March). “Better Computers.” *International Science and Technology*, No. 3: 35-39.

Carr, John W., III. (1962b. March). “Better Computers.” *Elektronische Rechenanlagen,* 4(4): 157-160.

Carr, John W., III. (1965, January). “The Future of Programming and Programmers.” *Computers and Automation,* 14(1), 15-17; 54.

Ceruzzi, Paul. (1989, October). “Electronics Technology and Computer Science, 1940-1975: A Coevolution.” *Annals of the History of Computing*, 10(4): 257-275.

Ceruzzi, Paul E. (2003). *A History of Modern Computing* (Second Edition). Cambridge, MA: MIT Press.

“Chairmen Named, Activities Planned for Three New TCs.” (1982, December). *Computer*, 15(12): 134.

Chase, W. H. (1961, December). “Merger Discussions Open Opportunities and Challenges.” *Electrical Engineering*, 80(12): 908-912.

Chu, Yaohan. (1974, December). “Why Do We Need Computer Hardware Description Languages.” *Computer*, 7(12): 18-22.

Coates, Clarence L. (1968). “University Education in Computer Engineering.” In *Proceedings of the Meeting on Computer Science in Electrical Engineering of the Commission on Engineering Education*, Stanford University, October 24-25, 1968 (5-11). Washington, DC: National Academy of Engineering.

Coates, Clarence L., Jr., Bruce Arden, Thomas C. Bartee, C. Gordon Bell, Franklin F. Kuo, Edward J. McCluskey, Jr., and William H. Surber, Jr. (1971, June). “An Undergraduate

Computer Engineering Option for Electrical Engineering." *Proceedings of the IEEE,* 59(6): 854-860.

Cohen, Arnold A. (1961, December). "The Chairman's Letter." *IRE Transactions on Electronic Computers,* EC-10(4): 845.

Cohen, Arnold A. (1962a, February). "The Chairman's Letter." *IRE Transactions on Electronic Computers,* EC-11(1): 119.

Cohen, Arnold A. (1962b, April). "The Chairman's Letter." *IRE Transactions on Electronic Computers,* EC-11(2): 319.

Cohen, Arnold A. (1964, June). "Minutes of the Joint Meeting." *IEEE Transactions on Electronic Computers,* EC-13(3): 341.

Committee on Computer Sciences in Electrical Engineering of the Committee on Engineering Education. (1968). *A Program to Stimulate the Development of Electrical Engineering Courses and Curricula To Include the Computer Sciences (Continued Support)* [Proposal].

Committee on the Undergraduate Program in Mathematics (CUPM) of the Mathematical Association of America. (1964, May). *Recommendations on the Undergraduate Mathematics Program for Work in Computing.* Berkeley, CA: Committee on the Undergraduate Program in Mathematics.

"COMPCON." (1972, July/August). *Computer,* 5(4): 30.

Computing Curricula for Computer Engineering Joint Task Force. (2004, December 12). *Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering*. IEEE Computer Society. Retrieved November 28, 2006 from http://www.eng.auburn.edu/ece/CCCE/CCCE-FinalReport-2004Dec12.pdf

"Computing Devices Conference Attracts Interested Audience." (1949, April). *Electrical Engineering*, 68(4): 358.

"COMPSAC 77." (1977, February). *Computer,* 10(2): 4-5.

"Computer Science and Engineering Board Established at Academy of Sciences; Oettinger Named Chairman." (1968, July). *Communications of the ACM,* 11(7): 530.

"Computer Science Curriculum." (1964, April). *Communications of the ACM,* 7(4): 205.

"Computer Society Members Surveyed by AFIPS." (1972, May/June). *Computer*, 5(3): 12.

"Computer Society Starts Education Activity." (1971, July/August). *Computer,* 4(4): 35.

"Computer Society Votes to End National Computer Conference." (1987, August). *Computer*, 20(8): 110.

Concordia, Charles. (1976, December). "In the Beginning there was the AIEE Committee on Computing Devices." *Computer*, 9(12): 42, 44.

Concordia, Charles. (1994). [Oral history interview conducted by Frederick Nebeker]. New Brunswick, NJ: IEEE History Center, Rutgers University. Retrieved October 26, 2006 from http://www.ieee.org/portal/cms_docs_iportals/iportals/aboutus/history_center/oral_history/pdfs/Concordia189.pdf

Condon, Edward U. (1947, April). "Electronics and the Future." *Electrical Engineering*, 66(4): 355-361.

"Conference Report: 2nd International Conference on Software Engineering." (1976, December). *Computer*, 9(12): 68-71.

"Constitution for the IRE Professional Group on Electronic Computers." (1955, September). *IRE Transactions on Electronic Computers*, EC-4(3): 88-92.

Conte, Sam D. (1964). "The Computer Sciences Program at Purdue University." In *Proceedings of the 1964 19th ACM National Conference* (L1.2-1). New York, NY: ACM Press.

Conway, Melvin E. (1968, April). "How Do Committees Invent?" *Datamation*, 14(4): 28-31.

"Conway's Law." (2003, December 29). *The Jargon File 4.4.7*. Retrieved July 11, 2006 from http://www.catb.org/jargon/html/C/Conways-Law.html

Cook, Charles C. (1963). *A Survey of Digital Computer Instruction in Most Major U.S. Engineering Colleges.* Morgantown, WV: The Department of Industrial Engineering, College of Engineering, West Virginia University.

Correll, Quention. (1958, July). [Letter to the Editor]. *Communications of the ACM*, 1(7): 2.

Cortada, James W. (1993). *The Computer in the United States: From Laboratory to Market, 1930 to 1960.* Armonk, NY and London, England: M. E. Sharpe, Inc.

COSINE Committee of the Commission on Engineering Education. (1967a). *Summary of Talks and Discussion Group Recommendations, Conference on Computer Sciences in Electrical Engineering Education,* Princeton University, March 28-29, 1967 (6-8). Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1967b, September). *Computer Sciences in Electrical Engineering*. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1968a, March). "Computer Science in Electrical Engineering." *IEEE Spectrum*, 5(3): 96-103.

COSINE Committee of the Commission on Engineering Education. (1968b, September). *Some Specifications for a Computer-Oriented First Course in Electrical Engineering*. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1968c, October). *An Undergraduate Course On Computer Organization*. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1968d). *Proceedings of the Meeting on Computer Science in Electrical Engineering of the Commission on Engineering Education*, Stanford University, October 24-25, 1968. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1968e, November). *Some Specifications for an Undergraduate Course in Digital Subsystems.* Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1969a, September). *Impact of Computers on Electrical Engineering Education – A View From Industry.* Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Engineering Education. (1969b, December). *Computer-Oriented Electrical Engineering Experiments, 1969-1970.* Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Education. (1970, January). *An Undergraduate Computer Engineering Option for Electrical Engineering*. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Education. (1971a, March). *Digital Systems Laboratory Courses and Laboratory Developments*. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Education. (1971b, June). *An Undergraduate Course on Operating Systems Principles*. Washington, DC: National Academy of Engineering.

COSINE Committee of the Commission on Education. (1972, April). *Minicomputers in the Digital Laboratory Program*. Washington, DC: National Academy of Engineering.

Coulter, N. S. (1991, December 1). "Computer Curricula 1991 (Review)." *Computing Reviews*. Retrieved September 15, 2006 from http://www.reviews.com/review/review_reviewprint.cfm?review_id=115546

Cruz, J. B., Jr. (Ed.). (1963, June). “What is System Theory and Where is it Going? – A Panel Discussion.” *IEEE Transactions on Circuits and Systems*, 10(2): 154-160.

Dataman Associates. (1962a, November). “Careers in Computing.” *Datamation*, 8(11): 121.

Dataman Associates. (1962b, November). “Careers in Computing.” *Datamation*, 8(11): 123.

Davis, Malcom R. (1969, September). “IEEE Computer Group Personnel Survey.” *Computer Group News*, 2(11): 4-41.

Denning, Peter J. and Jack B. Dennis, Butler Lampson, A. Nico Haberman, Richard R. Muntz, and Dennis Tsichritzis. (1972, January/February). “An Undergraduate Course on Operating Systems Principles.” *Computer*, 5(1): 40-59.

Denning, Peter J. (Ed.), Edward Feigenbaum, Paul Gilmore, Anthony Hearn, Robert W. Ritchie, and Joseph Traub. (1981, June). “A Discipline in Crisis (The Snowbird Report).” *Communications of the ACM*, 24(6): 370-374.

Denning, Peter J., Douglas E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. (1988). *Report of the ACM Task Force on the Core of Computer Science.* New York, NY: ACM Press.

Denning, Peter J., Douglas E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. (1989a, January). “Computing as a Discipline.” *Communications of the ACM*, 32(1): 9-23.

Denning, Peter J., Douglas E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, and Paul R. Young. (1989b, February). “Computing as a Discipline.” *Computer*, 22(2): 63-70.

Dertouzos, Michael L., Theodore R. Bashkow, Herbert J. Carlin, Ernest S. Kuh, Joseph E. Roew, Louis D. Smullin, and M. E. Van Valkenburg. (1971, November). “Insight Versus Algorithms: A Leader’s View.” *IEEE Transactions on Education*, E-14(4): 164-169.

Dickmann, Robert A. (1971, October). “Summary report on the 1971 AFIPS Information Processing Personnel Survey.” Montvale, NJ: AFIPS Press.

DiMaggio, Paul J. and Walter W. Powell. (1983, April). “The Iron Cage Revisited: Institutional Isomorphism and Collective Rationality in Organizational Fields.” *American Sociological Review*, 48(2): 147-160.

Douglas Aircraft Company. (1957, October). “Douglas Will Train You to Program Big Computers.” *Electrical Engineering*, 76(10): 98A.

Downey, Gary L. (1998). *The Machine in Me: An Anthropologist Sits Among Computer Engineers*. New York and London: Routledge.

"Editorial." (1953, March). *IRE Transactions on Electronic Computers*, EC-2(1): 1.

"Editorial Prospectus." (1962). *Computer Design*, 1(1): 2-3.

Editors of DATA-LINK (Los Angeles ACM Chapter Newsletter). (1958, April). "What's in a Name?" [Letter to the Editor]. *Communications of the ACM*, 1(4): 6.

"Education Committee Added to Computer Society." (1971, November/December). *Computer*, 4(6): 11.

Education Committee of the IEEE Computer Society. (1977, January). *A Curriculum in Computer Science and Engineering – Committee Report*. Long Beach, CA: IEEE Computer Society.

Edwards, Paul N. (1996). *The Closed World: Computers and the Politics of Discourse in Cold War America*. Cambridge, MA: MIT Press.

"EECS History." (n.d.) Retrieved October 24, 2006 from UC Berkeley EECS web site: http://www.eecs.berkeley.edu/department/history.shtml

"Electronic Digital Computers Considered in Five Papers." (1949, March). *Electrical Engineering*, 68(3): 266.

Engel, Gerald L. (1977, December). "A Comparison of the ACM/C^3S and the IEEE/CSE Model Curriculum Subcommittee Recommendations." *Computer*, 10(12): 121-123.

Engineering Research Associates, Inc. (1952a, July). "Digital Computer Engineers." *Proceedings of the Institute of Radio Engineers*, 40(7): 118A.

Engineering Research Associates, Inc. (1952b, November). "Digital Computer Engineers." *Proceedings of the Institute of Radio Engineers*, 40(11): 129A.

Engineering Research Associates, Inc. (1952c, December). "Digital Computer Engineers." *Proceedings of the Institute of Radio Engineers*, 40(12): 132A.

Engineering Research Associates, Division of Remington Rand, Inc. (1954, May). "Unlimited Opportunities for … Electrical Engineers, and Physicists to do Digital Computer Engineering." *Electrical Engineering*, 73(5): 68A.

Engineering Societies Personnel Service, Inc. (1952, October). "Positions Available." *Electrical Engineering*, 71(10): 86A-87A.

Engstrom, H. T. (1956). “Keynote Address.” In *Proceedings of the Eastern Joint Computer Conference*, New York, NY, December 10-12, 1956 (3-4). New York, NY: American Institute of Electrical Engineers.

Ensmenger, Nathan. (2001, October-December). “The 'Question of Professionalism' in the Computer Fields.” *IEEE Annals of the History of Computing*, 23(4): 56-74.

Ernst, Rolf and Gaetano Borriello. (1997). “Message from the Workshop Chairs.” In *Proceedings of the Fifth International Workshop on Hardware/Software Codesign* (CODES/CASHE '97), March 24-26, 1997 (viii). Washington, DC: IEEE Computer Society.

“Executive Committee – March 2, 1948.” (1948, May). *Proceedings of the IRE*, 36(5): 633.

“Extensive Plans Set for 1947 IRE National Convention.” (1947, February). *Proceedings of the IRE*, 35(2): 172-184.

Fein, Louis. (1959). “The Role of the University in Computers, Data Processing, and Related Fields.” *Communications of the ACM*, 2(9): 7-14.

Fein, Louis. (1961a, June). “The Computer-Related Sciences (Synnoetics) at a University in the Year 1975.” *American Scientist*, 49(2): 149-168.

Fein, Louis. (1961b, September). “The Computer-Related Sciences (Synnoetics) at a University in 1975.” *Datamation*, 7(9): 34-41.

Fein, Louis. (1963, April). “Renaming the PGEC” [Letter to the Editor]. *IEEE Transactions on Electronic Computers*, EC-12(2): 136.

Fein, Louis. (1979). [Oral history interview conducted by Pamela McCorduck, May 9, 1979, Palo Alto, California]. Minneapolis, MN: Charles Babbage Institute, University of Minnesota. Retrieved May 3, 2006 from http://www.cbi.umn.edu/oh/pdf.phtml?id=117

Felker, J. H. (1952a, February). “The Transistor as a Digital Computer Component.” In *Review of Electronic Digital Computers – Joint AIEE-IRE Computer Conference*, Philadelphia, PA, December 10-12, 1951 (105-109). New York, NY: American Institute of Electrical Engineers (AIEE).

Felker, J. H. (1952b, November). “Regenerative Amplifier for Digital Computer Applications.” *Proceedings of the IRE*, 40(11): 1584-1596.

Feng, Tse-yun. (1980, December). “From the President.” *Computer*, 13(12): 3.

Fife, Dennis W. (1968, January). “Session V: Panel Discussion – The Role of Electrical Engineers in Computer Science” (Session Report). *Computer Group News*, 2(1): 20-21.

Fife, Dennis W. (1983, July). “Trends in Membership Development.” *Computer*, 16(7): 6-7.

Finerman, Aaron. (1968). “University Education in Computer Science (Summary).” In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science held at the State University of New York at Stony Brook, June 1967* (193-214). New York and London: Academic Press.

“First Annual IEEE Computer Conference.” (1967, September). *Computer Group News*, 1(8): 1-7.

“Five Sessions Held at Conference on Electron Tubes for Computers.” (1951, February). *Electrical Engineering*, 70(2): 163.

Flamm, Kenneth. (1988). *Creating the Computer: Government, Industry, and High Technology.* Washington, DC: The Brookings Institution.

Flynn, Michael J. (1972, September/October). “How Can Computing Interests Best Be Served” [Letter to the Editor]. *Computer*, 5(5): 64.

“Foreward.” (1952a, February). In *Review of Electronic Digital Computers – Joint AIEE-IRE Computer Conference*, Philadelphia, PA, December 10-12, 1951 (3). New York, NY: American Institute of Electrical Engineers (AIEE).

“Foreward.” (1952b, December). *Transactions of the I.R.E. Professional Group on Electronic Computers*, 1(1), 1.

Forrester, Jay W. (1952, February). “Digital Computers – Present and Future Trends.” In *Review of Electronic Digital Computers – Joint AIEE-IRE Computer Conference*, Philadelphia, PA, December 10-12, 1951 (109-113). New York, NY: American Institute of Electrical Engineers.

Foster, Caxton. (1970a). *Computer Architecture*. New York, NY: Van Nostrand Reinhold

Foster, Caxton. (1970b, September). "Are You Interested in Computer Architecture?" *Communications of the ACM*, 13(9): 526.

Foster, Caxton. (1972, March/April). “Computer Architecture.” *Computer*, 5(2): 19.

Forrester, Jay W. (1957). “Equipmental Aids to Computing.” In Preston C. Hammer (Ed.), *The Computing Laboratory in the University* (15-24). Madison, WI: The University of Wisconsin Press.

Forsythe, George E. (1961, December). “Engineering Students Must Learn Both Computing and Mathematics.” *Journal of Engineering Education*, 52(3): 177-188.

Forsythe, George E. (1963). “Educational Implications of the Computer Revolution.” In W. F. Freiberger and William Prager (Eds.), *Applications of Digital Computers* (166-178). Boston, MA: Ginn and Co.

Forsythe, George E. (1964a, April). “An Undergraduate Curriculum in Numerical Analysis.” *Communications of the ACM*, 7(4): 214-215.

Forsythe, Goerge E. (1964b, October). “Chairmen of ACM Committees.” *Communications of the ACM*, 7(10): 635.

Forsythe, George E. (1967, January). “A University’s Educational Program in Computer Science.” *Communications of the ACM*, 10(1): 3-11.

Forsythe, George E. (1968, May). “What to Do Till the Computer Scientist Comes.” *The American Mathematical Monthly*, 75(5): 454-462.

Foster, Caxton C. (1972, March/April). “Computer Architecture.” *Computer*, 5(2): 19.

The Foxboro Company. (1967, July). “Progress-Minded Programmers…” *Datamation*, 13(7): 130.

Franke, David W. and Martin K. Purvis. (1991). “Hardware/Software CoDesign: A Perspective.” In *Proceedings of the 13th International Conference on Software Engineering*, Austin, TX. Los Alamitos, CA: IEEE Computer Society Press.

Frater, W. H. (1955). “Opening Remarks by the Chairman.” In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (21-22). Detroit, MI: Wayne University Press.

Freeman, Herbert. (1982). *Research Directions in Computer Engineering, Report of a Workshop*, Washington, DC, November 15-16, 1982. Washington, DC: National Science Foundation.

Freeeman, Herbert. (1983, May). “Research Directions in Computer Engineering.” *Computer*, 16(5): 80-82.

Fritz, W. Barkley. (1963, April). “Selected Definitions.” *Communications of the ACM*, 6(4): 152-158.

Galison, Peter. (1997). *Image and Logic: A Material Culture of Microphysics.* Chicago and London: The University of Chicago Press.

Galler, Bernard. (1962, January). “Definition of Software” [Letter to the Editor]. *Communications of the ACM*, 5(1): 6.

Galler, Bernard. (1991). [Oral history interview by Enid H. Galler, August 1991, Sutton's Bay, Michigan]. Minneapolis, MN: Charles Babbage Institute, University of Minnesota. Retrieved May 3, 2006 from http://www.cbi.umn.edu/oh/pdf.phtml?id=126

Galey, J. Michael. (1975, August). "Microprogramming: The Bridge Between Hardware and Software." *Computer*, 8(8): 23.

Gannett, E. K. (1953a, October). "Acknowledgment." *Proceedings of the IRE*, 41(10): 1219.

Gannett, E. K. (1953b, October). [Introduction to guest editorial by Werner Buchholz.] *Proceedings of the IRE*, 41(10): 1220.

Garcia, Oscar N. (1982, January). "From the President." *Computer*, 15(1): 4-5.

Garcia, Oscar N. (1983, January). "From the President." *Computer*, 16(1): 4-5.

Garner, Harvey L. (1964, April). "Critique." *Communications of the ACM*, 7(4): 224-225.

General Motors Research Laboratories. (1960, November/December). "…at the outer edge of computer science." *Datamation*, 6(6): 75.

General Motors Research Laboratories. (1961a, March). "Opportunities at the outer edge of computer science…" *Datamation*, 7(3): 59.

General Motors Research Laboratories. (1961b, August). "Applied Mathematicians, Programmers: Opportunities at the outer edge of computer science…" *Datamation*, 7(8): 83.

Ghosh, S. P., C. Harlaw, M. Tsuchiya, A. B. Salisbury, D. Pessel, D. C. Rine, and E. J. Smith. (1975). "IEEE Computer Education: The Regional HELP Subcommittee." In *COMPCON '75 Digest of Papers: Proceedings of the Spring '75 COMPCON Conference*, February 25-27, 1975, San Francisco, CA (37-39). Long Beach, CA and New York, NY: IEEE Computer Society.

Gibbs, Norman E. and Allen B. Tucker. (1986, March). "A Model Curriculum for a Liberal Arts Degree in Computer Science." *Communications of the ACM*, 29(3): 202-210.

Gieryn, Thomas F. (1983). "Boundary Work and the Demarcation of Science from Non-Science: Strains and Interests in Professional Ideologies of Scientists." *American Sociological Review*, 48: 781-795.

Gieryn, Thomas F. (1995). "Boundaries of Science." In Sheila Jasanoff, Gerald E. Markle, James C. Petersen, and Trevor Pinch (Eds.), *Handbook of Science and Technology Studies*, Revised Edition (393-443). Thousand Oaks, London, and New Delhi: Sage Publications.

Gieryn, Thomas F. (1999). *Cultural Boundaries of Science: Credibility on the Line.* Chicago: University of Chicago Press.

Gilchrist, Bruce. (1959, January). "University Computing Courses." *Journal of Engineering Education*, 49(4): 342-346.

Gilchrist, Bruce. (1961a, March). "Changes in Bylaws." *Communications of the ACM*, 4(3): 136.

Gilchrist, Bruce. (1961b, June). "ACM Membership Survey – January 1, 1961." *Communications of the ACM*, 4(6): 254.

Gilchrist, Bruce. (1962, June). "ACM Membership Survey – January 1, 1962." *Communications of the ACM*, 5(6): 297.

Gilfillan. (1952a, May). "Experienced Radar and Computer Engineers." *Proceedings of the Institute of Radio Engineers*, 40(5): 106A.

Gilfillan. (1952b, June). "Experienced Radar and Computer Engineers." *Proceedings of the Institute of Radio Engineers*, 40(5): 104A.

Gill, Stanley. (1968). "Planning a Profession." In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (117-121). New York and London: Academic Press.

Golinski, Jan. (1998). *Making Natural Knowledge: Constructivism and the History of Science.* Cambridge: Cambridge University Press.

Good, Gregory A. (2000). "The Assembly of Geophysics: Scientific Disciplines as Frameworks of Consensus." *Studies in History and Philosophy of Modern Physics*, 31(3): 259-292.

Goode, Harry H. (1955, June). "PGEC Student Activities and Education in Computers." *IRE Transactions on Computers*, 4(2): 49-51.

Gorn, Saul. (1958, January.) "Letters to the Editor." *Communications of the ACM*, 1(1), 2-4.

Gorn, Saul. (1959, September). "On the Logicial Design of Formal Mixed Languages." *Preprints of Papers Presented at the 14th National Meeting of the Association for Computing Machinery* (25-1). New York, NY: ACM Press.

Gorn, Saul. (1963, April). "The Computer and Information Sciences: A New Basic Discipline." *SIAM Review*, 5(2): 150-155.

"The Great Conference Debate (Editor's Readout)." (1963, March). *Datamation*, 9(3): 25-26.

Green, Judy, Jeanne LaDuke, Saunders Mac Lane, and Uta C. Merzbach. (1998, August). "Mina Spiegel Rees (1902-1997)." *Notices of the AMS,* 45(7): 866-873. Retrieved December 20, 2006 from http://www.ams.org/notices/199807/memorial-rees.pdf

Grems, Mandalay, and *Datamation* Staff. (1960, January/February). "EJCC Impressons." *Datamation,* 6(1): 23-25.

Gries, David and Dorothy Marsh. (1992, January). "The 1989-90 Taulbee Survey." *Communications of the ACM,* 35(1): 133-143.

Grosch, Herbert R. J. (1957). "The Computer Laboratory in Industry." In Preston C. Hammer (Ed.), *The Computing Laboratory in the University* (87-90). Madison, WI: The University of Wisconsin Press.

Grosch, Herbert R. J. (1961, July). "Software in Sickness and Health." *Datamation,* 7(7): 32-33.

Grosch, Herbert R. J. (1971). [Oral history interview conducted by Richard R. Mertz.] Retrieved October 26, 2006 from http://invention.smithsonian.org/downloads/fa_cohc_tr_gros710330.pdf

Guttag, John (Ed.). (2005). *The Electron and the Bit, Electrical Engineering and Computer Science at MIT, 1902-2002.* Cambridge, MA: MIT, Electrical Engineering and Computer Science Department.

Hamblen, John W. (1967, August). *Computers in Higher Education: Expenditures, Sources of Funds, and Utilization for Research and Instruction 1964-1965, with Projections for 1968-1969* (Report on a Survey Conducted with the National Science Foundation). Atlanta, GA: Southern Regional Education Board.

Hamming, Richard W. (1969, January). "One Man's View of Computer Science (1968 ACM Turing Lecture)." *Communications of the ACM,* 16(1): 3-12.

Harder, Edwin L. (1957a). "The Computing Revolution." *Electrical Engineering,* 76(6): 476-481.

Harder, Edwin L. (1957b). "The Computing Revolution." *Electrical Engineering,* 76(7): 586-590.

Hartman, Frank R. (1959, November). "The Demand for College Training in Digital Computing." *Computers and Automation,* 8(11): 14.

Hartmanis, Juris, and Herbert Lin. (Eds.). (1992). *Computing the Future: A Broader Agenda for Computer Science and Engineering*. Washington, DC: National Academy Press.

Harder, Edwin L. (1991). [Oral history interview conducted by William Aspray]. New Brunswick, NJ: IEEE History Center, Rutgers University. Retrieved October 23, 2006

from
http://www.ieee.org/portal/cms_docs_iportals/iportals/aboutus/history_center/oral_history/pdfs/Harder118.pdf

"Hardware." (1989). *Oxford English Dictionary Online* (Second Edition). Retrieved January 19, 2006 from http://dictionary.oed.com/

Hartman, Frank R. (1959, November). "The Demand for College Training in Digital Computing." *Computers and Automation,* 8(11): 11-14.

Hartree, Douglas R. (1947). *Calculating Machines.* Cambridge, UK: Cambridge University Press.

Harvard Computation Laboratory. (1946). *Manual of Operation for the Automatic Sequence Controlled Calculator*, The Annals of the Computation Laboratory of Harvard University, Volume 1. Cambride, MA: Harvard University Press.

Harvard Computation Laboratory. (1948). *Proceedings of a Symposium on Large-Scale Digital Calculating Machinery*, Harvard University, January 7-10, 1947. The Annals of the Computation Laboratory of Harvard University, Volume 16. Cambridge, MA: Harvard University Press.

Heising, William P. (1961, February). "EJCC Program Highlights." *Datamation,* 7(2): 36-38.

Hess, David. (1997). *Science Studies: An Advanced Introduction*. New York, NY: New York University.

Hettinger, Norman G. (1972, July/August). "On Merging with the ACM" [Letter to the Editor]. *Computer,* 5(4): 4.

Hoagland, Albert S. (1972a, January/February). "Some Thoughts on the JCCs, Professionalism and Society Goals." *Computer,* 5(1): 6.

Hoagland, Albert S. (1972b, May/June). "Should we merge with the ACM and leave the IEEE? An interview with Dr. Albert S. Hoagland, President of the IEEE Computer Society." *Computer,* 5(3): 1.

Hoagland, Albert S. (1973, February). "The IEEE Computer Society and the ACM." *Communications of the ACM,* 16(2): 67-68.

Hobbs, L. Charles. (1968a, January). "Chairman's Letter." *Computer Group News,* 2(1): 3A.

Hobbs, L. Charles. (1968b, November). "The Computer Group During 1968." *Computer Group News,* 2(6): 33.

Honeywell Electronic Data Processing. (1962b, January). “A Few Quick Facts on Software.” *Datamation*, 8(1): 46-47.

Honeywell Electronic Data Processing. (1962b, March). “More Facts About Honeywell Software.” *Datamation*, 8(3): 2-3.

Honeywell Electronic Data Processing. (1962c, April). “Engineers – Programmers.” *Datamation*, 8(4): 10-11.

Honeywell Electronic Data Processing. (1963, October). “The Dimensions of Logic Design at Honeywell.” *Computer Design*, 2(9): 23.

Honeywell Electronic Data Processing. (1965, April). “Job titles are clues, but they can be misleading.” *Computer Design*, 4(4): 61.

Honeywell Electronic Data Processing. (1966, February). “Circuit Design – Computers are realized in the mind of the Circuit Design Engineer.” *Datamation*, 12(2): 124.

Honeywell Electronic Data Processing. (1966, March). “System Design – Computers are conceived in the mind of the System Design Engineer.” *Datamation*, 12(3): 113.

Hopper, Grace M. (1953). “Compiling Routines.” *Computers and Automation*, 2(4): 1-5.

Hopper, Grace M. and John W. Mauchly. (1953, October). “Influence of Programming Techniques on the Design of Computers.” *Proceedings of the IRE*, 41(10), 1250-1254.

Householder, Alston S. (1954, January). “The End of an Epoch: The Joint Computer Conference, Washington, DC, December, 1953.” *Computers and Automation*, 3(1): 6-7.

Householder, Alston S. (1956a, January). “Presidential Address to the ACM, Philadelphia, September 14, 1955.” *Journal of the Association for Computing Machinery*, 3(1): 1-2.

Householder, Alston S. (1956b, May). “The Position of the University in the Field of High Speed Computation and Data Handling.” *Computers and Automation*, 5(5): 6-10.

Householder, Alston S. (1957, January). “Retiring Presidential Address.” *Journal of the Association for Computing Machinery*, 4(1): 1-4.

Huggins, William. (1969). “History and Activities of the COSINE Committee.” In William Viavant (Ed.), *Proceedings of the Park City Conference, Computers in Undergraduate Education, Volume I*, Park City, Utah, September 8-13, 1968 (51-67). Salt Lake City, UT: University of Utah.

Hughes Aircraft Company. (1958a, July). “The sky is no longer the limit.” *Electrical Engineering*, 77(7): 66A-67A.

Hughes Aircraft Company. (1958b, December). “The strange shape of defense.” *Electrical Engineering*, 77(12): 80A-81A.

Hughes Aircraft Company. (1960, May/June). “Digital Computer Engineers.” *Datamation*, 6(3): 71.

Hughes Research and Development Laboratories. (1954, May). “Digital Computer Techniques.” *Computers and Automation*, 3(5): 5.

Hughes, Joseph L. A., John Impagliazzo, Andrew McGettrick, Victor P. Nelson, David Soldan, Pradip K. Srimani, and Mitchell D. Theys. (2004, December). *Computer Curricula: Computer Engineering* (Final Report of the IEEE-CS/ACM Joint Task Force on Computing Curricula 2004). Retrieved from the World Wide Web on Feb. 10, 2005: http://www.eng.auburn.edu/ece/CCCE/CCCE-FinalReport-2004Dec12.pdf

Hunter, G. T. (1955). “Manpower Requirements by Computer Manufacturers.” In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (14-18). Detroit, MI: Wayne University Press.

Huskey, Harry D. (1955). “Status of University Educational Programs Relative to High Speed Computation.” In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (22-25). Detroit, MI: Wayne University Press.

Huskey, Harry D. (1960a, September). “Letter from the President of ACM.” *Communications of the ACM*, 3(9): 481.

Huskey, Harry D. (1960b, December). “Letter from the President of ACM.” *Communications of the ACM*, 3(12): 631.

Huskey, Harry D. (1961a, August). “A Perspective.” *Datamation*, 7(8): 18.

Huskey, Harry D. (1961b, December). “Letter from the President of ACM.” *Communications of the ACM*, 4(12): 530.

Huskey, Harry D. (1962ac, March). “Letter from the President of ACM.” *Communications of the ACM*, 5(3): 131.

Huskey, Harry D. (1962b, April). “Letter from the President of ACM.” *Communications of the ACM*, 5(4): 186.

Huskey, Harry D. (1991, July/September). “Harry D. Huskey: The Early Days (Memior).” *Annals of the History of Computing*, 13(3): 290-306.

Husson, Samir S. (1970). *Microprogramming: Principles and Practices*. Englewood Cliffs, NJ: Prentice-Hall, Inc.

IBM. (1954, November). "IBM has positions open in Development and Manufacturing for Electronic and Electro-mechanical Engineers." *Electrical Engineering*, 73(11): 72A.

IBM. (1955a, January). "IBM Has positions open for Engineers." *Electrical Engineering*, 74(1): 68A.

IBM. (1955b, May). "The Challenge of Creative Engineering." *Electrical Engineering*, 74(5): 77A.

IBM. (1955c, December). "The legacy of the scientist." *Electrical Engineering*, 74(12): 75A.

IBM. (1957, February). "Where do you belong in IBM Military Products?" *Electrical Engineering*, 76(2): 78A-79A.

IBM. (1961. October). "Immediate Opportunities with the IBM Advanced Systems Development Division." *Datamation*, 7(10): 94.

IBM. (1962, May). "At IBM, Programmers shape the future of a new technology." *Computers and Automation*, 11(5): 39.

IEEE Center for the History of Electrical Engineering. (1984). "The Origins of the IEEE." Retrieved April 20, 2006 from http://www.ieee.org/organizations/history_center/historical_articles/history_of_ieee.html

"IEEE Computer Group." (1965, February). *IEEE Transactions on Electronic Computers*, EC-14(1): Inside front cover.

"IEEE Computer Group Constitution and Bylaws." (1965, February). *IEEE Transactions on Electronic Computers*, EC-14(1): 1-6.

"IEEE Computer Society Constitution." (1977, July). *Computer*, 10(7): 108-109.

"IEEE Computer Society Technical Committees." (1976, May). *Computer*, 9(5): 26-27.

"IEEE-CS Membership Grows at Record Rate, Exceeds 62,000." (1982, March). *Computer*, 15(3): 105.

IEEE Educational Activities Board. (1978, February). "ECPD Accreditation Guidelines: Preliminary Computer Science and Engineering Programs." *Computer*, 11(2): 67-69.

"Information for Authors." (1961a, September). *IRE Transactions on Electronic Computers*, EC-10(3): Inside back cover.

"Information for Authors." (1961b, December). *IRE Transactions on Electronic Computers*, EC-10(4): Inside back cover.

"Institute Committees – 1954." (1954, October). *Proceedings of the IRE*, 42(10): 1580-1585.

"The Institute on the March – A New Professional Group System." (1948, May). *Proceedings of the IRE*, 36(5): 570.

"Integrated circuit." (2006, October 19). In *Wikipedia, The Free Encyclopedia*. Retrieved October 19, 2006 from http://en.wikipedia.org/wiki/Integrated_circuit

"Inventor Profile – George Stibitz." (2002). In *National Inventors Hall of Fame*. Retrieved November 7, 2006 from http://www.invent.org/hall_of_fame/140.html

"The IRE Professional Group System – A Status Report." (1948, December). *Proceedings of the IRE*, 36(12): 1507.

Irwin, David J. and C. V. Ramamoorthy. (1975, December). "Guest Editor's Introduction." *Computer*, 8(12): 26-27.

"Is it Overhaul or Trade-in Time?" (1959a, July/August). [Edited transcript]. *Datamation*, 5(4): 24-33.

"Is it Overhaul or Trade-in Time?" (1959b, September/October). [Edited transcript]. *Datamation*, 5(5): 17-26; 44-45.

"Is the Computer Field Staying Together or Separating?" (1958, June). *Computers and Automation*, 7(6): 6; 96.

Jacobson, Arvid W. (Ed.). (1955a). *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954. Detroit, MI: Wayne University Press.

Jacobson, Arvid W. (1955b). "Preface." In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (n.p.). Detroit, MI: Wayne University Press.

Jacobson, Arvid W. (1955c). "Opening Remarks by the Chairman." In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (3-4). Detroit, MI: Wayne University Press.

Jasanoff, Sheila (Ed.). (2004). *States of Knowledge: The Co-Production of Science and Social Order*. London and New York: Routledge.

Jensen, E. Douglas. (1973, November). "Hardware vs. Software: The Two Faces of Computers." *Computer*, 6(11): 14-15.

Jet Propulsion Laboratory, California Institute of Technology. (1954, June). “Caltech Laboratory Jet Propulsion.” *Proceedings of the IRE*, 42(6): 121A.

“Joint AIEE-IRE Computer Conference in Philadelphia Attracts Over 900.” (1952, February). *Electrical Engineering*, 71(2): 189.

“Joint AIEE-IRE Conference Committee.” (1952, February). In *Review of Electronic Digital Computers – Joint AIEE-IRE Computer Conference*, Philadelphia, PA, December 10-12, 1951 (3). New York, NY: American Institute of Electrical Engineers.

Joint Committee of the Association for Computing Machinery and the IEEE Computer Society. (1978). *A Library List on Undergraduate Computer Science, Computer Engineering, and Information Systems*. Long Beach, CA and New York, NY: IEEE Computer Society and ACM.

“Joint IRE/AIEE Computer Conference Slated.” (1951, October). *Proceedings of the IRE*, 39(10): 1343.

The Joint Task Force on Computing Curricula. (2001, December 15). *Computing Curricula 2001: Computer Science*. IEEE Computer Society and Association for Computing Machinery. Retrieved November 28, 2006 from www.computer.org/portal/cms_docs_ieeecs/ieeecs/education/cc2001/cc2001.pdf

Jones, Edwin C., Jr. and Michael C. Mulder. (1984, April). “Accreditation in the Computer Profession.” *Computer*, 17(4): 24-27.

Jones, Peter D. (1966, September). “Thirteen Programming Paradoxes.” *Datamation*, 12(9): 157.

Joseph, Earl C. (1969, March). “Evolving Digital Computer System Architectures.” *Computer Group News*, 2(8): 2-8.

Kagan, Claude A. R. (1961, March). “Computer Papers at the American Institute of Electrical Engineers Meeting in New York, Jan. 29-Feb. 3, 1961.” *Computers and Automation*, 10(3B): 6B.

Kapla, Gadi. (1999, June). “Charles Concordia: A Renowned Power Systems Guru Will Receive the 1999 IEEE Medal of Honor.” *IEEE Spectrum*, 36(6): 29-33.

Karp, Richard. (2004). “A Personal View of Computer Science at Berkeley.” In *U.C. Berkeley Computer Science 30th Anniversary Celebration*, Berkeley, CA, February 28, 2004. Retrieved February 16, 2006 from http://www.eecs.berkeley.edu/BEARS/CS_Anniversary/karp-talk.html

Katz, Adolph. (1960, June). “Do Computers Belong in the Engineering Curricula.” *Journal of Engineering Education*, 50(10): 835-838.

Katz, Donald, and Elliott I. Organick. (1960, December). “Use of Computers in Engineering Undergraduate Teaching.” *Journal of Engineering Education*, 51(3): 183-205.

Katz, Donald, and Elliott I. Organick, Silvio O. Navarro, and Brice Carnahan (1963). *The Use of Computers in Engineering Education: Final Report of Project.* Ann Arbor, MI: College of Engineering, The University of Michigan.

Kearfott Division, General Precision, Inc. (1960, November/December). “Digital Computer Engineers.” *Datamation*, 6(6): 80.

Keenan, Thomas A. (1964, April). “Computers and Education.” *Communications of the ACM*, 7(4): 205-209.

Kidder, Tracy. (1981). *The Soul of a New Machine*. Boston and Toronto: Little, Brown and Company.

King, Willis K. and Oscar N. Garcia. (1975, July). “Second Annual Symposium on Computer Architecture (Conference Report).” *Computer*, 8(7): 79-80.

Knuth, Donald E. (1968). *The Art of Computer Programming, Volume 1: Fundamental Algorithms* (First Edition). Reading, MA: Addison-Wesley.

Knuth, Donald E. (1972, August). “George Forsythe and the Development of Computer Science.” *Communications of the ACM*, 15(8): 721-726.

Kline, Ronald R. (2006, July). “The Emergence of ‘Information Technology’ as a Keyword, 1948–1985.” *Technology and Culture*, Vol. 47, No. 3: pp. 513-535.

Koffman, Elliot B., Philip L. Miller, and Caroline E. Wardle. (1984, October). “Recommended Curriculum for CS1, 1984.” *Communications of the ACM*, 27(10): 998-1001.

Koffman, Elliot B., David Stemple, and Caroline E. Wardle. (1985, August). “Recommended Curriculum for CS2, 1984: A Report of the ACM Curriculum Task Force for CS2.” *Communications of the ACM*, 28(8): 815-818.

Korfhage, Robert R. (1964, April). “Logic for the Computer Sciences.” *Communications of the ACM*, 7(4): 216-218.

Latour, Bruno. (1987). *Science in Action: How to Follow Scientists and Engineers Through Society.* Cambridge, MA: Harvard University Press.

“Large-Scale Computer Developments Discussed.” (1947, March). *Electrical Engineering*, 66(3): 289-290.

"The Last FJCC." (1987, December). *Computer*, 20(12): 100.

Layton, Edwin T. (1971). *The Revolt of the Engineers: Social Responsibility and the American Engineering Profession*. Cleveland, OH: Press of Case Western Reserve University.

Lee, J.A.N. (1995). *Computer Pioneers*. Los Alamitos, California: IEEE Computer Society Press.

Lee, J.A.N. (2001, January/March). "John Weber Carr III" (Obituaries). *IEEE Annals of the History of Computing*, 23(1): 67.

Lehmer, Derrick. (1952). [Summary] In *Proceedings of the Electronic Computer Symposium*, Los Angeles, CA, April 30-May 2, 1952 (Session XX:1-3). Los Angeles, CA: Los Angeles Chapter of the IRE Professional Group on Electronic Computers.

Lenoir, Timothy. (1997). *Instituting Science: The Cultural Production of Scientific Disciplines*. Stanford: Stanford University Press.

Lesser, Murray L. (1952). "An Approach to the Use of the IBM Card-Programmed Electronic Calculator in the Solution of Engineering Problems." In *Proceedings of the Electronic Computer Symposium*, Los Angeles, CA, April 30-May 2, 1952 (Session IX). Los Angeles, CA: Los Angeles Chapter of the IRE Professional Group on Electronic Computers.

Levine, Samuel. (1966, July). "Chairman's Letter (1966-1967)." *Computer Group News*, 1(1): n.p.

Levine, Samuel. (1967a, January). "Chairman's Letter." *Computer Group News*, 1(4): n.p.

Levine, Samuel. (1967b, March). "Chairman's Letter." *Computer Group News*, 1(5): 2.

Levine, Samuel. (1968, September). "The Computer Group Contemplates Its Next Move." *Computer Group News*, 2(5): 16-18.

Librascope. (1957, June). "I just have to tell you…" *Electrical Engineering*, 76(6): 88A.

Lindsay, Tom. (1969, March). "On the Track of New Members." *Computer Group News*, 2(8): 32.

Lindvall, F. C. (1955). "Computers Challenge Engineering Education." *Proceedings of the Western Joint Computer Conference*, Los Angeles, CA, March 1-3, 1955. New York, NY: Institute of Radio Engineers.

"Looking Back, Looking Ahead: A SIAM History." (2002). Philadelphia, PA: Society for Industrial and Applied Mathematics. Retrieved April 22, 2006 from http://www.siam.org/about/pdf/siam50.pdf

Lucena, Juan C. (2005). *Defending the Nation: U.S. Policymaking to Create Scientists and Engineers from Sputnik to the 'War against Terrorism.'* Lanham, MD: University Press of America, Inc.

Luebbert, William F. (1960, November). "Computers in Engineering Education" [Letter]. *Journal of Engineering Education*, 51(2): 134-137.

Mauchly, John W. (1948). "Preparation of Problems for EDVAC-Type Machines." In *Proceedings of a Symposium on Large-Scale Digital Calculating Machinery,* Harvard University, January 7-10, 1947 (203-207). Cambridge, MA: Harvard University Press.

Mauchly, Kahtleen R. (1984, April-June). "John Mauchly's Early Years." *Annals of the History of Computing*, 6(2): 116.

MacKenzie, Donald. (1990). *Inventing Accuracy: A Historical Sociology of Nuclear Missile Guidance*. Cambridge, MA: The MIT Press.

Macnaughton, Peter C. (1972, July/August). "On Merging with the ACM" [Letter to the Editor]." *Computer*, 5(4): 4-5.

MacWilliams, W. H., Jr. (1952). "Keynote Address." In *Review of Electronic Digital Computers – Joint AIEE-IRE Computer Conference*, Philadelphia, PA, December 10-12, 1951 (5-6). New York, NY: American Institute of Electrical Engineers.

Madden, J. Don. (1963, April). "Improving the Current Format." *Datamation*, 9(4): 45-46.

Magel, Kenneth I., Richard H. Austing, Alfs Berztiss, Gerald L. Engel, John W. Hamblen, A. A.J. Hoffman, and Robert Mathis. (1981, March). "Recommendations for Master's Level Programs in Computer Science: A Report of the ACM Curriculum Committee on Computer Science." *Communications of the ACM*, 24(3): 115-123.

Mahoney, Michael S. (1988, April). "The History of Computing in the History of Technology." *IEEE Annals of the History of Computing*, 10(2): 113-125.

Mahoney, Michael S. (1990). "The Roots of Software Engineering." *CWI Quarterly*, 3(4): 325-34.

Mahoney, Michael S. (1996). "Issues in the History of Computing." In Thomas J. Bergin and Rick G. Gibson (Eds.), *History of Programming Languages II* (772-781). New York: ACM Press.

Mahoney, Michael S. (2000). “Software as Science – Science as Software.” Retrieved September 14, 2006 from http://www.princeton.edu/~mike/softsci.htm

Mahoney, Michael S. (2004a, May). “The Histories of Computing(s).” Lecture delivered in the “Digital Scholarship, Digital Culture” Series, Centre for Computing in the Humanities, King's College, London, UK. Retrieved from the World Wide Web on May 10, 2004: http://www.princeton.edu/~mike/articles/histories/kingscch.htm

Mahoney, Michael S. (2004b, March). “Finding a History for Software Engineering.” *Annals of the History of Computing*, 26(1): 8-19.

“Manpower Needs and Educational Programs: Panel Discussion, A. C. Hall, Chairman.” (1955). In Arvid W. Jacobson (Ed.), *Proceedings of the First Conference on Training Personnel for the Computing Machine Field*, Detroit, MI, June 22-23, 1954 (32-34). Detroit, MI: Wayne University Press

Martin, William L. (1955). “Foreward.” *Proceedings of the Western Joint Computer Conference*, Los Angeles, CA, March 1-3, 1955 (p. 1). New York, NY: Institute of Radio Engineers.

Martin, William L. and S. R. Olson. (1957, March). “PGEC Membership Survey.” *IRE Transactions on Electronic Computers*, EC-6(1): 49-55.

Matula, David W. (1969, December). “Minutes of SIGCSE Meeting at FJCC (Las Vegas, Nov. 18, 1969).” *ACM SIGCSE Bulletin*, 1(4): 6.

McCluskey, Edward J. (1967). “The ACM-C^{3}S Curriculum.” In *Summary of Talks and Discussion Group Recommendations, Conference on Computer Sciences in Electrical Engineering Education*, Princeton University, March 28-29, 1967 (6-8). Washington, DC: National Academy of Engineering.

McCluskey, Edward. (1970, January/February). “Message from the Chairman: The Year of the Opening.” *Computer Group News*, 3(1): 2-3.

McCluskey, Edward. (1970, September, October). “On the Group’s Petition for a Change to Society Status.” *Computer*, 3(5): 1.

McCluskey, Edward J. (1976). “Logic Design.” In Anthony Ralston and Chester L. Meek (Eds.), *Encyclopedia of Computer Science, First Edition* (809-813). New York, NY: Van Nostrand Reinhold Company.

McCluskey, Edward. (2005). [Personal correspondence between Brent K. Jesiek and Edward J. McCluskey via telephone on September 27, 2005 and October 7, 2005.]

McCracken, Dan. (1979, March). “The Institute for Certification of Computer Professionals: A Call for ACM Action.” *Communications of the ACM*, 22(3); 145-146.

McCracken, Dan. (1980, February). “ACM President’s Letter: ACM Governance.” *Communications of the ACM*, 23(2): 65-66.

McMahon, A. Michal. (1984). *The Making of a Profession: A Century of Electrical Engineering in America*. New York, NY: IEEE Press.

McMahon, E. Lawrence, and Brice Carnahan, Donald L. Katz, and Warren D. Seider (1966). *Computers in Engineering Design Education: Volume IV – Electrical Engineering.* Ann Arbor, MI: College of Engineering, The University of Michigan.

McNeill, Daniel, and Paul Freiberger. (1993). *Fuzzy Logic*. New York, NY: Simon and Schuster.

Merkle, Luiz Ernesto. (2001). *Disciplinary and Semiotic Relations across Human-Computer Interaction.* Unpublished Dissertation. The University of Western Ontario.

Merkle, Luiz Ernesto, and Robert E. Mercer. (2002). “Variations in Computing Science's Disciplinary Diversity: The Case of Curricula Recommendations.” In Lillian N. Cassel and Ricardo Augusto da Luz Reis (Eds.), *Informatics Curricula and Teaching Methods* (87-96). IFIP TC3/WG3.2 Conference on Informatics Curricula, Teaching Methods, and Best Practice (ICTEM 2002), July 10-12, 2002, Florianópolis, SC, Brazil.

Mesa Scientific Corporation. (1964, November). “Mesa Men Now Come in Two Convenient Types: Software… and Hardware!” *Datamation*, 10(11): 64.

Messer-Davidow, David R. Shumway, and David J. Sylvan (Eds.). (1993). *Knowledges: Historical and Critical Studies in Disciplinarity*. Charlottesville and London: University Press of Virginia.

Miller, C. L. and W. W. Seifert. (1960, June). “The Faculty and the Computer – Some Problems and Goals.” *Journal of Engineering Education*, 50(10): 839-845.

MITRE Corporation. (1961, May). “Computer Engineers and Scientists.” *Computers and Automation*, 10(5): 23.

Model Program Committee of the IEEE Educational Activities Board. (1983). *The 1983 IEEE Computer Society Model Program in Computer Science and Engineering*. Silver Spring, MD: IEEE Computer Society Press.

Moon, Suzanne. (2004, July). “Tracy Kidder, *The Soul of a New Machine*” (Classics Revisited Book Review). *Technology and Culture*, 45(3): 597-602.

Moone, Tom. (2002, October). “Mac Van Valkenburg: ‘One of the very best engineering teachers in the world.’” *Ingenuity* (UIUC ECE Newsletter), 7(3). Retrieved October 24, 2006 from http://www.ece.uiuc.edu/ingenuity/1002/mac.html

Morse, Philip M. (Ed.). (1960, October). “Report on a Conference of University Computing Center Directors, June 2-4, 1960.” *Communications of the ACM*, 3(10): 519-521.

Mulder, Michael, George Davida, Oscar N. Garcia, Sakti P. Ghosh, and David Pessel. (1975). “Model Curricula for Computer Science and Engineering Programs.” In *COMPCON '75 Digest of Papers: Proceedings of the Spring ’75 COMPCON Conference*, February 25-27, 1975, San Francisco, CA (33-35). Long Beach, CA and New York, NY: IEEE Computer Society.

Mulder, Michael. (1975, December). “Model Curricula for Four-Year Computer Science and Engineering Programs: Bridging the Tar Pit.” *Computer*, 8(12): 28-33.

Mulder, Michael. (1977, December). “Computer Science and Engineering Education: Introduction and Overview.” *Computer*, 10(12): 70-71.

Mulder, Michael, and John Dalphin. (1984, April). “Computer Science Program Requirements and Accreditation.” *Computer*, 17(4): 30-35.

Muller, David E. (1964, April). “The Place of Logical Design and Switching Theory In the Computer Curriculum.” *Communications of the ACM*, 7(4): 222-224.

National Cash Register Company. (1956a, January). “Engineers for immediate placement.” *Electrical Engineering*, 75(1): 59A.

National Cash Register Company. (1956b, April). “Digital Computer Engineers.” *Computers and Automation*, 5(4): 41.

National Cash Register Company. (1960, December). “Digital Computer Engineers.” *Computers and Automation*, 9(12): 5.

Neumann, Peter G. (1976, May). “Letter from the Editor.” *ACM SICSOFT Software Engineering Notes*, 1(1): 2-3.

“New Attendance Record Set at AIEE Fall General Meeting.” (1956, December). *Electrical Engineering*, 75(12): 1108-1112.

“New Programming Committee Chairman.” (1967, September). *Computer Group News*, 1(8): 31.

Newell, Allen, Alan J. Perlis, and Herbert A. Simon. (1967, September 22). “Computer Science” [Letter to the Editor]. *Science*, 157: 1373-1374.

“News: Association for Computing Machinery.” (1948, April). *Mathematical Tables and Other Aides to Computation*, 3(22): 132-134.

"News: Association for Computing Machinery." (1949, January). *Mathematical Tables and Other Aides to Computation*, 3(25): 380.

"News." (1954, September). *IRE Transactions on Electronic Computers*, EC-3(3): 39.

Norberg, Arthur L. (2005). *Computers and Commerce: A Study of Technology and Management at Eckert-Mauchly Computer Company, Engineering Research Associates, and Remington Rand, 1946-1957*. Cambridge, MA and London England: The MIT Press.

Northrop Aircraft, Inc. (1956, January). "Computers." *Computers and Automation*, 5(1): 53.

North American Aviation, Inc. (1956, October). "New Developments in flutter, vibration, electronics, many other specialized fields: Exceptional Opportunities Now." *Electrical Engineering*, 75(10): 103A.

Oettinger, Anthony G. (1966a, April). "On ACM's Responsibility." *Communications of the ACM*, 9(4): 246.

Oettinger, Anthony G. (1966b, December). "President's Letter to the ACM Membership." *Communications of the ACM*, 9(12): 838-839.

Oettinger, Anthony G. (1967, October). "The Hardware-Software Complimentarity." *Communications of the ACM*, 10(10): 604-606.

Oettinger, Anthony G. (1968a). "Computers and Education." In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (27-38). New York and London: Academic Press.

Oettinger, Anthony G. (1968b, May). "President's Letter to the ACM." *Communications of the ACM*, 11(5): 293-294.

Opler, Ascher. (1967, January). "Fourth-Generation Software." *Datamation*, 13(1): 22-24.

"Organization of the National Joint Computer Committee." (1956). *Proceedings of the Eastern Joint Computer Conference*, New York, NY, December 10-12, 1956 (1-2). New York, NY: American Institute of Electrical Engineers.

"Panels Feature of First IEEE Computer Conference." (1967, October). *Datamation*, 13(10): 109-110.

Patrick, Robert L. (1961, May). "An Editorial Commentary on The Gap in Programming Support." *Datamation*, 7(5): 37.

Patrick, Robert L. (1966, June). "Not-So-Random Discs." *Datamation*, 12(6): 77-78.

"Pentium FDIV Bug." (2006, November 28). In *Wikipedia, The Free Encyclopedia*. Retrieved November 28, 2006 from http://en.wikipedia.org/wiki/Pentium_FDIV_bug

Perlis, Alan J. (1964, August). "Report of the Commission of Thoughtful Persons to the ACM Council, 24 April 1964." *Communications of the ACM*, 7(8): 507-508.

Perlis, Alan J. (1968). "Computer Science is Neither Mathematics nor Electrical Engineering." In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (69-81). New York and London: Academic Press.

Peterson, Harold A., and Charles Concordia. (1945, September). *General Electric Review*, 48: 29-37.

Philco Computer Division. (1960, April). "Computer Engineers." *Computers and Automation*, 9(4): 35.

Philco Computer Division. (1960b, November/December). "Computer Growth Opportunities." *Datamation*, 6(6): 81.

Phister, Montgomery. (1958, tenth printing 1967). *Logical Design of Digital Computers*. New York, NY: John Wiley and Sons.

Phister, Montgomery. (2005). [Personal correspondence between Brent K. Jesiek and Montgomery Phister via telephone on October 14, 2005.]

"Post Conference Feedback." (1958, May/June). *Datamation*, 4(3): 20-21.

"Program, AIEE Summer General Meeting." (1947, June). *Electrical Engineering*, 66(6): 592-594.

"Progress of Institute Technical Groups Program." (1961, May). *Electrical Engineering*, 80(5): 372-373.

"Professional Group Notes." (1951, November). *Proceedings of the IRE*, 39(11): 1466.

"Provisional IEEE Computer Society Constitution and Bylaws." (1970, September/October). *Computer*, 3(5): 33-37.

"Q – Science." (Library of Congress Classification Outline). (n.d.). Library of Congress. Retrieved October 9, 2006 from http://www.loc.gov/catdir/cpso/lcco/lcco_q.pdf

"Radio Progress During 1951." (1952, April). *Proceedings of the IRE*, 40(4): 388-439.

Ralston, Anthony. (1973a, January). “The Computer Society and ACM.” *Computer*, 6(1): 1-2.

Ralston, Anthony. (1973b, December). “Computer Science Resarch – Storm Clouds in Washington (ACM President’s Letter).” *Communications of the ACM*, 16(12): 725-726.

Ralston, Anthony, and Chester L. Meek (Eds.). (1976). *Encyclopedia of Computer Science (First Edition).* New York, NY: Van Nostrand Reinhold Company.

Ralston, Anthony. (1980). “Preface.” In Anthony Ralston (Ed.), *Taxonomy of Computer Science and Engineering* (v). Arlington, VA: The American Federation of Information Processing Societies, Inc.

Ralston, Anthony, and Edwin D. Reilly, Jr. (1983). *Encyclopedia of Computer Science and Engineering (Second Edition).* New York, NY: Van Nostrand Reinhold Company.

Ralston, Anthony. (2004, January-March). “Four Editions and Eight Publishers: A History of the *Encyclopedia of Computer Science*.” *IEEE Annals of the History of Computing*, 26(1): 42-52.

Ramamoorthy, C. V. (1976, December). “Computer Science and Engineering Education.” *IEEE Transactions on Computers*, C-25(12): 1200-1206.

Ramo, Simon. (1960, January). “Intellectronics.” *Computers and Automation*, 9(1): 6, 23.

Randell, Brian. (1982). *The Origins of Digital Computers: Selected Papers* (3rd ed.). Berlin and New York: Springer-Verlag.

Randell, Brian. (2002). “The New Electronic Technology.” In Atsushi Akera and Frederik Nebeker (Eds.), *From 0 to 1: An Authoritative History of Modern Computing* (41-50). Oxford, England and New York, NY: Oxford University Press.

“Record Attendance at Computer Conference.” (1953, February). *Proceedings of the IRE*, 41(2): 299.

“Record Growth: IEEE Tops 200,000; Computer Society over 44,000.” (1980, March). *Computer*, 13(3): 96.

Rector, Robert W. (1986, July). “Personal Recollections on the First Quarter-Century of AFIPS.” *Annals of the History of Computing*, 8(3): 261-269.

Redmond, Kent C., and Thomas M. Smith (1980). *Project Whirlwind: The History of a Pioneer Computer*. Bedford, MA: Digital Press.

“Reflections on a Quarter-Century: AFIPS Founders.” (1986, July). *Annals of the History of Computing*, 8(3): 225-256.

"Report on the Chicago Gathering." (1967, November). *Computer Group News*, 1(9): 24.

"Report of the Board of Directors." (1954, August). *Electrical Engineering*, 73(8): 755-782.

"Report of the Board of Directors." (1955, August). *Electrical Engineering*, 74(8): 709-740.

"Report of the Board of Directors." (1956, August). *Electrical Engineering*, 75(8): 729-761.

"Report of the Board of Directors." (1957, August). *Electrical Engineering*, 76(8): 711-748.

Republic Aviation. (1955, March). "Creative Engineering Opportunities with Republic." *Computers and Automation*, 4(3): 31.

"Retirements: Samuel Byron Williams." (1946, June). *Bell Laboratories Record*, 24(6): 252-253.

Reynolds, Terry S. (1986, October). "Defining Professional Boundaries: Chemical Engineering in the Early 20th Century." Technology and Culture, 27(4): 694-716.

Rhodes, Ida. (1952). "The Human Computer's Dreams of the Future." In *Proceedings of the Electronic Computer Symposium*, Los Angeles, CA, April 30-May 2, 1952 (Session XII:1-5). Los Angeles, CA: Los Angeles Chapter of the IRE Professional Group on Electronic Computers.

Rice, Rex. (1973, September). "COMPCON: Establishing a Conference Identity." *Computer*, 6(9): 15-16.

Rideout, Vincent C. (1957). "Curriculum Needs in the Computing Field." In Preston C. Hammer (Ed.), *The Computing Laboratory in the University* (153-159). Madison, WI: The University of Wisconsin Press.

Rine, David C., S. P. Ghosh, C. A. Harlow, and M. Tsuchiya. (1976). "Regional HELP for Computer Education." In *COMPCON '76 Digest of Papers: Proceedings of the Spring '76 COMPCON Conference*, February 24-26, 1976, San Francisco, CA (208-211). Long Beach, CA and New York, NY: IEEE Computer Society.

Rine, David C. and Ralph E. Lee. (1978). "Introductory Remarks." In *Proceedings of College Curriculum in Computer Science, Engineering, and Data Processing*, February 2-3, 1978, Orlando, FL (front matter). Long Beach CA: IEEE Computer Society.

Rine, David C. (1979, September). "Special Message – From the Education Committee Chairman." *Computer*, 12(9): 3-5.

Rojas, Raúl, and Ulf Hashagen (Eds.). (2000). *The First Computers: History and Architecture*. Cambridge, MA and London: The MIT Press.

Rose, C. W. and M. Albarran. (1975). “Modeling and Design Description of Hierarchical Hardware/Software Systems.” In *Proceedings of the 12th Conference on Design Automation* (421-430). Piscataway, NJ: IEEE Press.

Rossmann, George E., C. Gordon Bell, Michael J. Flynn, Frederick P. Brooks, Jr., Samuel H. Fuller, and Herbert Hellerman. (1975, December). “A Course of Study in Computer Hardware Architecture.” *Computer*, 8(12): 44-57.

Rubinoff, Morris. (1971). [Oral history interview conducted by Richard R. Mertz on May 17, 1971.] Retrieved November 7, 2006 from http://invention.smithsonian.org/downloads/fa_cohc_tr_rubi710517.pdf

Russo, Roy L. (1983, November). “From the Vice-President for Technical Activities.” *Computer*, x(11): 6.

Ryder, John D., and Donald G. Fink. (1984). *Engineers and Electronics: A Century of Electrical Progress.* New York, NY: IEEE Press.

Sacks, S. Henry. (1963, June). “Joint Computer Conferences (Editorial Notes).” *Computer Design*, 2(6): 1.

Salisbury, A. B., J. N. Snyder, and E. J. Smith. (1975). “A Report of the Subcommittee on Coordination, IEEE Computer Society, Education Committee.” In *COMPCON '75 Digest of Papers: Proceedings of the Spring ’75 COMPCON Conference*, February 25-27, 1975, San Francisco, CA (41). Long Beach, CA and New York, NY: IEEE Computer Society.

Sammet, Jean E. (1976, May). “What Has Been Accomplished?” *Communications of the ACM*, 19(5): 227-228.

Samuel, Arthur L. (1953, October). “Computer Bit by Bit or Digital Computers Made Easy.” *Proceedings of the IRE*, 41(10): 1223-1230.

Saunders, Robert M. (1965, June-September). “Electrical Engineering Education in 1975.” *IEEE Transactions on Education*, E-8(2-3): 33-37.

Schweppe, Earl J. (1964). “A Proposed Academic Program in the Computer Sciences.” In *Proceedings of the 1964 19th ACM National Conference* (L1.1-1-L1.1-2). New York, NY: ACM Press.

Scott, Norman R. (1961, September). “Editorial.” *IRE Transactions on Electronic Computers*, EC-10(3): front matter.

Seely, Bruce E. (1999, July). “The Other Re-engineering of Engineering Education, 1900-1965.” *Journal of Engineering Education*, 88(3): 285-294.

Seising, Rudolf. (2005). “1965 – ‘Fuzzy Sets’ appear – A Contribution to the 40th Anniversary.” In *Proceedings of the 2005 IEEE International Conference on Fuzzy Systems*, Reno, NV, May 22-25, 2005 (5-10). IEEE.

Serra, Micaela., and William B. Gardner. (1998). “A First Course in Hardware/Software Codesign.” In *Proceedings of the Third Western Canadian Conference on Computing Education (WCCCE '98)*, Vancouver, BC, May, 1998 (57-66). Retrieved on May 6, 2004 from: http://www.uoguelph.ca/~gardnerw/pubs/WCCCE98.pdf

Shapiro, Fred R. (2000, April/June). “Origin of the term software: Evidence from the JSTOR electronic journal archive.” *IEEE Annals of the History of Computing*, 22(2): 69-70.

Shapiro, Stuart. (1994). “Boundaries and Quandaries: Establishing a Professional Context for IT.” *Information Technology and People*, 7(1): 47-68.

Shaw, Christopher J. (1962, September). “Programming Schisms.” *Datamation*, 8(9): 32.

Shaw, Mary (Ed.). (1985). *The Carnegie-Mellon Curriculum for Undergraduate Computer Science*. New York, NY: Springer-Verlag.

Simmons, Dick B. (1980, November). “From the Division V Director…” *Computer*, 13(11): 7.

Simmons, Dick B. (1982, September). “Membership Growth and Information Activities.” *Computer*, 15(9): 6-7.

Simon, Herbert A. (1969). *The Sciences of the Artificial*. Cambridge, MA: The MIT Press.

“SJCC Society Gleanings.” (1964, May). *Datamation*, 10(5): 19.

Slamecka, Vladimir. (1968). “The Science and Engineering of Information.” In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (81-92). New York and London: Academic Press.

Sloan, Martha. (1973). *The Impact of the COSINE Committee on the Undergraduate Electrical Engineering Curriculum*. Unpublished Dissertation. Stanford University.

Sloan, Martha. (1974, November). “The Impact of the COSINE Committee on the Undergraduate Electrical Engineering Curriculum.” *IEEE Transactions on Education*, E-17(4): 179-189.

Sloan, Martha. (1975, December). “Survey of Electrical Engineering and Computer Science Departments in the U.S.” *Computer*, 8(12): 35-42.

Sloan, Martha. (1985, December). “Two Years of Transition.” *Computer*, 18(12): 6-7.

Sloan, Martha. (2005). [Personal correspondence between Brent K. Jesiek and Martha Sloan via telephone on June 29, 2005 and July 27, 2005].

Sloan, Martha E., Clarence L. Coates, and Edward J. McCluskey. (1973, June). "COSINE Survey of Electrical Engineering Departments, Fall 1973." *Computer*, 6(6): 30-39.

Smith, Merlin G. (1977a, January). "From the President." *Computer*, 10(1): 2.

Smith, Merlin G. (1977b, December). "Special Message – From the President." *Computer*, 10(12): 3.

Smith, Merlin G. (1978, October). "New Transactions Launched; Computer Society Growth Continues." *Computer*, 11(10): 4.

Smith, Merlin G. (1991, September). "IEEE Computer Society: Four Decades of Service, 1951-1991." *Computer*, 24(9): 6-12.

Smotherman, Mark. (1999, March). "A Brief History of Microprogramming." Retrieved June 8, 2006 from http://www.cs.clemson.edu/~mark/uprog.html

"Software Engineering Prospectus Proposed." (1975, November). *Computer*, 8(11): 2.

Spaanenburg, Ben. (1982, January). "Mermelade or jam?" *Computer*, 15(1): 146.

"Standards on Electronic Computers: Definitions of Terms, 1950." (1951, March). *Proceedings of the IRE*, 39(3): 271-277.

Star, Susan L., and James R. Griesemer. (1989). "Institutional Ecology, `Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39." *Social Studies of Science*, 19(3): 387-420.

Stephan, Karl D. (2002, Fall). "All This and Engineering Too: A History of Accreditation Requirements." *IEEE Technology and Society Magazine*, 21(3): 8-15.

Stern, Nancy. (1980, April-June). "John William Mauchly: 1907-1980." *Annals of the History of Computing*, 2(2): 100-103.

"Symposia on Modern Calculating Machinery and Numerical Methods." (1949, January). *Mathematical Tables and Other Aids to Computation*, 3(25): 381-388.

"Symposium on the Impact of Computers on Science and Society." (1956, September). *IRE Transactions on Electronic Computers*, EC-5(3): 142-158.

System Development Corporation. (1959, November). "Computer Programmers: Seen any new horizons lately?" *Computers and Automation*, 8(11): 5.

"T – Technology (Library of Congress Classification Outline)." (n.d.). Library of Congress. Retrieved October 9, 2006 from http://www.loc.gov/catdir/cpso/lcco/lcco_t.pdf

Tanenbaum, Andrew S. (1976). *Structured Computer Organization.* Englewood Cliffs, NJ: Prentice-Hall, Inc.

Tartar, John (Ed.), Bruce Arden, Taylor Booth, Peter Denning, Ray Miller, and Andries van Dam. (1985, May). "1984 Snowbird Report: Future Issues in Computer Science." *Computer*, 18(5): 101-104.

"Technical Committee Notes." (1949, January). *Proceedings of the IRE*, 37(1): 62-63.

"Technical Committees – May 1, 1948-May 1, 1949." (1948, June). *Proceedings of the IRE*, 36(6): 761-762.

"Technical Committees, May 1, 1949-May 1, 1950." (1949, June). *Proceedings of the IRE*, 37(6): 668-669.

"Technical Committees, May 1, 1951-April 30, 1952." (1951, June). *Proceedings of the IRE*, 39(6): 721-722.

"Technical Committee Notes." (1949, December). *Proceedings of the IRE*, 37(12): 1448.

"Technical Interest Councils and Technical Committees." (1979, September). *Computer*, 12(9): 113.

Temco Aircraft Corporation. (1957, June). "In Engineering, the Best Opportunities are in Aviation. In Aviation, the Best Opportunities are at Temco." *Electrical Engineering*, 76(6): 75-79.

"Tentative Program, AIEE Winter Meeting." (1947, January). *Electrical Engineering*, 66(1):

"Tentative Program, Conference on Electron Tubes." (1948, March). *Electrical Engineering*, 67(3): 267.

Titus, James P. (1968, August). "The New NAS Board as a Government Advisor." *Communications of the ACM*, 11(8): 580-581.

Tompkins, Howard E. (1963). "Computer Education." In Franz L. Alt and Morris Rubinoff (Eds.), *Advances in Computers*, Volume 4 (135-168). New York and London: Academic Press.

Tucker, Allen B. (Ed.), Bruce H. Barnes, Robert M. Aiken, Keith Barker, Kim B. Bruce, J. Thomas Cain, Susan E. Conry, Gerald L. Engel, Richard G. Epstein, Doris K. Lidtke, Michael C. Mulder, Jean B. Rogers, Eugene H. Spafford, and A. Joe Turner. (1991).

Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force. New York, NY and Los Alamitos, CA: ACM Press and IEEE Computer Society Press.

Tucker, Allen B. (Ed.). (1991, June). "Computing Curricula 1991." *Communications of the ACM*, 34(6): 69-84.

Tucker, Allen B. and Bruce H. Barnes. (1991, November). "Flexible Design: A Summary of Computing Curricula 1991." *Computer*, 24(11): 56-66.

Tumbleson, Robert C. (1948, January). "Calculating Machines." *Electrical Engineering*, 67(1): 6-12.

Uncapher, Keith W. (1959, March). "1958 PGEC Membership Survey Report." *IRE Transactions on Electronic Computers*, EC-8(1): 61-67.

Uncapher, Keith W. (1961, March). "1960 PGEC Membership Report." *IRE Transactions on Electronic Computers*, EC-10(1): 81-91.

Uncapher, Keith W. (1964a, June). "Message from the New Chairman." *IEEE Transactions on Electronic Computers*, EC-13(3): 184.

Uncapher, Keith W. (1964b, December). "Chairman's Newsletter." *IEEE Transactions on Electronic Computers*, EC-13(6): 792.

Uncapher, Kieth W., Malcolm Davis, James Babcock, and Shirley Marks. (1959, January/February). "Computer Conferences: Some Observations, Some Suggestions." *Datamation*, 5(1): 47.

Van Atta, L. C. (1950, October). "The Role of Professional Groups in the IRE." *Proceedings of the IRE*, 38(10): 1124-1126.

Van der Spiegel, Jan, and James F. Tau, Titiimaea F. Ala'ilima, and Lin Ping Ang. (2000). "The ENIAC: History, Operation, and Reconstruction in VLSI." In Raúl Rojas and Ulf Hashagen (Eds.), *The First Computers: History and Architectures* (121-178). Cambridge, MA and London, England: The MIT Press.

Van Valkenburg, Mac E. (1967). "Objectives of the COSINE Committee." In *Summary of Talks and Discussion Group Recommendations, Conference on Computer Sciences in Electrical Engineering Education*, Princeton University, March 28-29, 1967 (3). Washington, DC: National Academy of Engineering.

Van Valkenburg, Mac E. (1971). "Foreward." *Proceedings of the IEEE*, 59(6): 854.

Van Valkenburg, Mac E. (1972, November). "Electrical Engineering Education in the U.S." *IEEE Transactions on Education*, E-15(4): 240-246.

Varga, Richard S. (1964). "Computer Technology at Case." In *Proceedings of the 1964 19th ACM National Conference* (L1.3-1-L1.3-2). New York, NY: ACM Press.

Vemuri, V. Rao. (1993, February). "Computer Science and Engineering Curricula." *IEEE Transactions on Education,* 36(1): 108-110.

Viavant, William (Ed.). (1968). *Proceedings of the Park City Conference, Computers in Undergraduate Education, Volume I,* Park City, Utah, September 8-13, 1968. Salt Lake City, UT: University of Utah.

Viehman, M. J. (1973, January). (Letter to the Editor). *Computer,* 6(1): 8.

Vincenti, Walter G. (1990). *What Engineers Know and How they Know It: Analytical Studies from Aeronautical History.* Baltimore and London: The Johns Hopkins University Press.

Ware, Willis H. (1959, June). "The Chairman's Column." *IRE Transactions on Electronic Computers,* EC-8(2): 90-91.

Ware, Willis H. (1963, April). "Perspective on AFIPS." *Datamation,* 9(4): 42-43.

Ware, Willis H. (1986, July). "AFIPS in Retrospect." *Annals of the History of Computing,* 8(3): 303-310.

Ware, Willis H. (2005). [Personal correspondence between Brent K. Jesiek and Willis H. Ware via telephone on November 3, 2005].

Wasserman, Anthony I. (1977a, January). "Letter from the Chairman." *ACM SICSOFT Software Engineering Notes,* 2(1): 1.

Wasserman, Anthony I. (1977b, July). "Chairman's Message." *ACM SIGSOFT Software Engineering Notes,* 2(4): 2.

Weiss, Eric A. (1968). "Industry's View of Computing Science." In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967* (105-116). New York and London: Academic Press.

Weiss, Eric A. (1988, January-March). "John Grist Brainerd: Obituary." *Annals of the History of Computing,* 10(1): 78-79.

Weiss, Eric A. (1992). "Saul Gorn (Obituary)." *IEEE Annals of the History of Computing,* 14(3): 76-77.

Westinghouse-Baltimore, Westinghouse Electric Corporation. (1957, February). "Are You Looking For A Job… Or A Career?" *Electrical Engineering,* 76(2): 89A.

"Where Should I Send My Manuscript?" (1955, September). *IRE Transactions on Electronic Computers*, 4(3): 87.

"Which Institute Technical Groups Do Members of AIEE Want to Join?" (1961, September). *Electrical Engineering*, 80(9): 704-705.

Whitby, Oliver. (1956). "Foreward." *Proceedings of the Western Joint Computer Conference*, San Francisco, CA, February 7-9, 1956. New York, NY: American Institute of Electrical Engineers.

Wiener, Norbert. (1948). *Cybernetics; or, Control and Communication in the Animal and the Machine*. New York, NY: John Wiley and Sons, Inc.

Wiesner, Jerome B. (1958, October). "Communication Sciences in a University Environment." *IBM Journal of Research and Development*, 2(4): 268-275.

Wildes, Karl L., and Nilo A. Lindgren. (1985). *A Century of Electrical Engineering and Computer Science at MIT, 1882-1982.* Cambridge, MA and London, England: The MIT Press.

Wilkes, Maurice V. (1989). "The Best Way to Design an Automatic Calculating Machine." In M. R. Williams and Martin Campbell-Kelly (Eds.), *The Early British Computer Conferences* (182-184). Cambridge, MA and London England: The MIT Press and Los Angeles/San Francisco: Tomash Publishers. (Original work published 1951)

Wilkes, Maurice V. (1969, September). "The Growth of Interest in Microprogramming: A Literature Survey." *Computing Surveys*, 1(3): 139-145.

Wilkes, Maurice V. (1992). "EDSAC 2." *IEEE Annals of the History of Computing*, 14(4): 49-56.

Wilkes, Maurice V. (2004). "The Origins and Growth of Electronic Engineering – A Personal View." Paper presented at the 2004 IEEE Conference on the History of Electronics, Bletchley Park, UK, June 28-30, 2004. Retrieved October 19, 2006 from http://www.ieee.org/portal/cms_docs_iportals/iportals/aboutus/history_center/conferences/che2004/Wilkes.pdf

Williams, Kathleen Broome. (1999, Summer). "Scientists in Uniform: The Harvard Computation Laboratory in World War II." *Naval War College Review*, LII(3). Retrieved October 19, 2006 from http://www.nwc.navy.mil/PRESS/Review/1999/summer/art4-su9.htm

Williams, Michael. (2002). "Computing before the Computer." In Atsushi Akera and Frederik Nebeker (Eds.), *From 0 to 1: An Authoritative History of Modern Computing* (11-24). Oxford, England and New York, NY: Oxford University Press.

Williams, Samuel B. (1953, January). “What Computers Do.” *The Computing Machinery Field*, 2(1): 21.

Williams, Samual B. (1954, January). “The Association for Computing Machinery.” *Journal of the Association for Computing Machinery*, 1(1): 1-3.

Winegrad, Dilys. (1996, Spring). “Celebrating the Birth of Modern Computing: The Fiftieth Anniversary of a Discovery at the Moore School of Engineering of the University of Pennsylvania.” *IEEE Annals of the History of Computing*, 18(1): 5-9.

Yau, Stephen S. (1974a, June). “From the President.” *Computer*, 7(6): 2.

Yau, Stephen S. (1974b, October). “From the President.” *Computer*, 7(10): 2-3.

Yau, Stephen S. (1975, January). “From the President.” *Computer*, 8(1): 2-3.

Yau, Stephen S. (1976, January). “Finished and Unfinished Business: A Message from the Outgoing President.” *Computer*, 9(1): 3-4.

Yau, Stephen S. (1981, January). “Proposed Bylaws Changes.” *Computer*, 14(1): 108.

Yau, Stephen S. (Ed.), Robert W. Ritchie, Warren Semon, Joseph F. Traub, Andries van Dam, and Stanley Winkler. (1983, December). “Meeting the Crisis in Computer Science.” *Computer*, 16(12): 83-87.

Yeargan, Jerry R. (2002, May). “The Integration of ABET and CSAB.” *IEEE Transactions on Education*, 45(2): 111-117.

Zadeh, Lotfi A. (1950, January). “Thinking Machines: A New Field in Electrical Engineering.” *Columbia Engineering Quarterly*, 3: 12-13, 30-31.

Zadeh, Lotfi A. (1965a). “Fuzzy Sets.” *Information and Control*, 8: 338-353.

Zadeh, Lotfi A. (1965b). “Electrical Engineering at the Crossroads.” *1965 IEEE International Convention Record*, 12(13): 47-50.

Zadeh, Lotfi A. (1965c, June-September). “Electrical Engineering at the Crossroads.” *IEEE Transactions on Education*, E-8(2-3): 30-33.

Zadeh, Lotfi A. (1967). “Curricula for Computer Science.” In *Summary of Talks and Discussion Group Recommendations, Conference on Computer Sciences in Electrical Engineering Education*, Princeton University, March 28-29, 1967 (9-10). Washington, DC: COSINE Committee of the Commission on Engineering Education.

Zadeh, Lotfi A. (1968a). “The Dilemma of Computer Sciences.” In Aaron Finerman (Ed.), *University Education in Computing Science, Proceedings of a conference on graduate*

academic and related research programs in computing science, held at the State University of New York at Stony Brook, June 1967 (61-68). New York and London: Academic Press.

Zadeh, Lotfi A. (1968b). "Computer Science as a Discipline." *Journal of Engineering Education*, 58(8): 913-916.

Zadeh, Lotfi A. (1971, November). "Impact of Computers on the Orientation of Electrical Engineering Curricula." *IEEE Transactions on Education*, E-14(4): 153-157.

Zadeh, Lotfi A. (1998). [Tribute to Mac Van Valkenburg.] In Tamer Basar (Ed.), *Mac Van Valkenburg Memorial Volume: Proceedings and Related Documents of the Mac Van Valkenburg Memorial Symposium, November 15, 1997* (134). UIUC ECE Publications Office.

Zadeh, Lotfi A. (2001). [Interview with Christian Freksa, Rudolf Kruse, Ramon López de Mántaras.] Retrieved October 24, 2006 from *KI Zeitschrift* [*AI Magazine*]: http://www.kuenstliche-intelligenz.de/index.php?id=%3ANO-1687&tx_ki_pi1[showUid]=254&cHash=1f5e15ef23

Zaphyr, P. A. (1959, January). [Letter to the Editor]. *Communications of the ACM*, 2(1): 4.