Considering Computing Education in Undergraduate Computer Science Programmes

Quintin Cutts*
University of Glasgow
Glasgow, Scotland
quintin.cutts@glasgow.ac.uk

Maria Kallia†
University of Glasgow
Glasgow, Scotland
maria.kallia@glasgow.ac.uk

Ruth Anderson
University of Washington
Seattle, USA
rea@cs.washington.edu

Tom Crick
Swansea University
Swansea, Wales
thomas.crick@swansea.ac.uk

Marie Devlin
Newcastle University
Newcastle, England
marie.devlin@newcastle.ac.uk

Mohammed Farghally
Virginia Tech
Blacksburg, USA
mfseddik@vt.edu

Claudio Mirolo
University of Udine
Udine, Italy
claudio.mirolo@uniud.it

Ragnhild Kobro Runde
University of Oslo
Oslo, Norway
ragnhilk@ifi.uio.no

Otto Seppälä
Aalto University
Espoo, Finland
otto.seppala@aalto.fi

Jaime Urquiza-Fuentes
Universidad Rey Juan Carlos
Móstoles, Spain
jaime.urquiza@urjc.es

Jan Vahrenhold
University of Münster
Münster, Germany
jan.vahrenhold@uni-muenster.de

ABSTRACT

This working group concerns the adoption of computing education (CE) in undergraduate computer science (CS) programmes. Such adoption requires both arguments sufficient to persuade our departmental colleagues and our education committees, and also curricular outlines to assist our colleagues in delivery. The goal of the group is to develop examples of both arguments and curricular outlines, drawing on any prior experience available.

CCS CONCEPTS

• **Social and professional topics → Model curricula**.

KEYWORDS

undergraduate; computing education; argument; curriculum outline

ACM Reference Format:

Quintin Cutts, Maria Kallia, Ruth Anderson, Tom Crick, Marie Devlin, Mohammed Farghally, Claudio Mirolo, Ragnhild Kobro Runde, Otto Seppälä, Jaime Urquiza-Fuentes, and Jan Vahrenhold. 2023. Considering Computing Education in Undergraduate Computer Science Programmes. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 2 (ITiCSE 2023), July 8–12, 2023, Turku, Finland.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3587103.3594210

*Co-leader

†Co-leader

ITiCSE 2023, July 8–12, 2023, Turku, Finland

ACM ISBN 979-8-4007-0139-9/23/07.
https://doi.org/10.1145/3587103.3594210

1 INTRODUCTION AND MOTIVATION

This working group considers the adoption of computing education (CE) in undergraduate computer science (CS) programmes. It does so as one approach to addressing a perceived shortage of people who have an understanding of both education and CS, and their merging in the field of CE; such people are needed to successfully drive the movement of CS education down into schools and also to help the population at large to come to terms with the transformative effects of digital technology, data and computation in our societies. The working groups also considers other motivations for the adoption of these courses. All of these are summarised below.

1.1 Direct issues caused by personnel shortages

The most obvious direct issue for CS education is the availability of suitably qualified teachers for the burgeoning school CS provision. While this is not a problem in all countries, it certainly affects many. Both the UK and the US for example, have significant shortages and recruitment is not easy. Most CS graduates go into industry careers; CS teacher training in both countries is being delivered often to student teachers who have no significant prior CS education experience. Instead, they receive a crash course in programming and other CS material that may last just a few weeks; or else they may have studied a little computing in the past. While it is commendable that such students are attempting to fill the breach, it is hard to imagine an established subject, like mathematics or physics, accepting student teachers with such limited disciplinary backgrounds. Disciplinary expertise comes with years of practice

A national school system will need leadership in discipline-specific education at many levels. All of the following roles will be more effective staffed by those with expertise in CE: government level staff leading curriculum planning, design and oversight processes; national teacher profession development agencies; the officers of sub-national administrative areas, such as local authorities or local school boards, to coordinate, lead and strengthen local teacher communities; and finally, the staff of companies producing CE resources for the education sector. Staff in all these positions, for long-standing school subjects, will have had school experience and hence have at least some understanding of the subject. The challenge for CS is that few have any experience of CE at all.

Even in academia, the established seat of CS education practice, there can be difficulty in recruiting CS education expertise – that is, staff familiar and practised with, and extending, the CS education research corpus. There are plenty of staff whose focus is teaching and exploring their own practice, but that is not the same as staff to take our understanding of CS education forward for all.

1.2 Industry needs

While the obvious need for those knowledgeable and interested in both CS and education is in the school sector, students destined for the software industry will also benefit from some understanding of education in the discipline, as follows.

Software engineers are always learning. Whether picking up new languages and systems, or working in new problem domains, fast learning of complex topics is required. Highly successful industry professionals move jobs on a regular basis. This requires highly-developed learning skills, to accommodate new problem or computing domains – but this is not made explicit. Undergraduate programmes vary in how well they prepare students for this: for example, the more languages they are exposed to, the better; the more new topic areas they must develop software in, the faster they will pick up and be productive in a new problem domain. A successful programme teaches students how to learn, however much of this is part of the so-called *hidden curriculum*. The working group will surface this essential skill in its course outlines and arguments.

Many of our students will progress in industry to become team leaders, where they will need to coach and mentor their team members in order to get the most out of them. Team members themselves will often be asked to support and work with new staff. Coaching and mentoring skills can be part of a CE course.

Both students and engineers learn a lot using informal learning resources. In addition to taking courses they browse documentation, blogs, tutorials, videos, etc. Some of the materials are pedagogically great, others bad. The lack of a great tutorial can drive someone to select a different technology just because they cannot get started. Adoption of new technologies is about making them understandable and easy to adopt: this justifies why creators of new technologies should know the basics of computing education and pedagogy.

1.3 Developing a full eco-system

Computing education as a crucial topic within the wider computing world is *coming of age*. When we educated a relatively tiny proportion of engineers and scientists who self-selected into university-level programmes, the lack of status for the topic was perhaps understandable; now that computing education is becoming part of mandatory school curricula worldwide, the study of how we teach the subject is of national importance. With international conferences and respected journals, and an increasing capacity for training PhD students in the field, and some research funding, we are piecing together the kind of ecosystem that other CS topic areas already have. Crucially, however, we do not have a presence in the undergraduate CS curriculum, and so CS students are not aware of CE as a topic of interest. Furthermore, the relatively few who do enter Masters and PhD programmes have to start their learning about CE, and education, from scratch. Finally, those who graduate and take up roles in academia may be seen as teaching faculty or generalists, rather than specialists with a strong research area, both to teach and to continue investigating.

Periodically, new topics appear and there are challenging times as they are recognised as valid subjects of study within the wider discipline. HCI has travelled this rocky road and is now largely accepted, although there are purists who would argue against it. We anticipate the same challenges for CE, and so will need to develop strong arguments for its inclusion.

2 OBJECTIVES

The working group has the following three key objectives.

- *Literature review and other related work.* While we are not aware of CE courses in typical undergraduate curricula in any institutions, we do know of tutor/TA training courses with significant CE research underpinnings, as well as Masters and PhD level preparation, and also of teacher training programmes with strong CE input. We will conduct a literature review and also for more informal approaches in order to be able to draw on existing expertise.
- *Arguments for undergraduate CE courses.* We know that the undergraduate curriculum is crowded, and that departmental battles rage about what to include. We will work both within and beyond the working group to develop and test arguments to persuade departmental committees that CE should have at least some footing within the curriculum. Our multi-national team will be able to consider the highly situated nature of education: an argument fit for German academia may not work in Italian institutions, for example.
- *Curriculum outlines.* There are many issues and alternatives to work through in considering curriculum outlines: does the curriculum principally concern best CE practices, or research frontiers, or both; what are the key topics; is there an appropriate developmental sequence; what material is relevant in industry; is this compulsory or optional material; and is it embedded in existing modules, or delivered standalone? In order to develop a range of possible outlines, we will run a Delphi-like process with a number of cycles to surface a range of potential topic areas and coalesce onto the ones that are most popular.

We expect to produce a collection of argument/outline pairs, each designed for a particular CE niche. For example, a strong software engineering department might accept a CE course with a clear industry training element. Ultimately, we hope that an open international community of practice will form around this work.