A VISUAL SIMULATION LIFE-CYCLE OF THE QUESTON PHYSICIAN NETWORK

Jong (Brian) H. Jun

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

Master of Science

in

Industrial and Systems Engineering

_____

Dr. Sheldon H. Jacobson, Chair

_____

Dr. Osman Balci

_____

Dr. John E. Kobza

April 27, 1999

Blacksburg, Virginia

Keywords: Simulation, Health Care, Object-oriented Modeling

A Visual Simulation Life-cycle of the Queston Physician Network

Jong (Brian) H. Jun

Committee Chairman: Dr. Sheldon H. Jacobson

Industrial and Systems Engineering

(ABSTRACT)

This research develops a discrete-event simulation model of the Queston Physician Network using the Visual Simulation Environment (VSE), an object-oriented simulation software program. The Queston Physician Network, a subsidiary of Biological & Popular Culture, Inc., attempts to centralize the administrative, financial, and telecommunication needs of a network of primary care physicians located throughout the United States. The VSE, running on the NeXTSTEP operating system, is a discrete event simulation software package with the capability to tackle the complexities associated with such a network design. The advantages of VSE over other simulation languages include its visualization of objects, domain independence, and object-oriented design and modeling.

The objective of the Queston simulation model is to addresses the performance capabilities of the physician network operationally centralized in the Queston Information Center. Additionally, the model could be used to analyze a physician-patient encounter of a generic clinic to identify recommended staffing and scheduling schemes.

Object-oriented programming allows the objects in the model to be instantiated at the time of execution. This feature permits the creation of one flexible generic clinic that can be reused to produce several identical clinics at program execution. In this model, one generic, family practice clinic and the Queston Information Center are created. Input data provided by both medical experts and a time study are used. Verification and validation techniques are applied in all phases of the modeling effort. Results using different configurations are presented and recommendations for future research are discussed.

ACKNOWLEDGEMENTS

I would like to express thanks to my former colleague, James R. Swisher, director of analysis and design at Biological & Popular Culture, Inc. (Biopop).. Without his technical and personal support, I would have been unable to complete this thesis.

I would like to express my gratitude to the chairman of my advisory committee, Dr. Sheldon H. Jacobson, for his guidance, encouragement, and support of my graduate studies as well as my professional duties at Biopop.

I would like to thank Dr. Osman Balci for serving as my thesis committee member. His invaluable support and teaching capabilities have enhanced my understanding of object-oriented programming and modeling. I acknowledge his company, ORCA Inc., and its employees for their technical support of the Visual Simulation Environment.

I would like to thank Dr. John E. Kobza for serving as a member of my thesis committee member and his advice in this research endeavor.

Additionally, I wish to thank Randall J. Kirk, president and chief executive officer, Biopop, Lee Talbot, vice-president of information systems, Biopop, and Dr. Edward Heller, president of Heller Consulting.

I would like to thank my parents for their enduring support and encouragement. Finally, I thank Kelly A. Sullivan for her technical, editorial, and loving support throughout my graduate studies and to whom I will always be indebted.

TABLE OF CONTENT

CHAPTER 1 INTRODUCTION 1

1.1 THE QUESTON PHYSICIAN NETWORK 1

1.2 PURPOSE OF THE MODEL 2

1.3 SUMMARY OF THE CHAPTERS 3

CHAPTER 2 LITERATURE REVIEW 5

2.1 PATIENT FLOW 8

2.1.1 Patient Scheduling and Admissions 8

2.1.2 Patient Routing and Flow Schemes 14

2.1.3 Scheduling and Availability of Adequate Resources 16

2.2 ALLOCATION OF RESOURCES 18

2.2.1 Bed Sizing and Planning 19

2.2.2 Room Sizing and Planning 22

2.2.3 Staff Sizing and Planning 24

2.3 DISCUSSIONS FOR FUTURE DIRECTIONS 28

2.3.1 Complex Integrated Multi-facility Systems 29

2.3.2 Combining Simulation and Optimization Techniques 32

2.3.3 Advances in Simulation Software 34

2.3.4 Implementation Issues 36

2.4 CONCLUSIONS 37

CHAPTER 3 SIMULATION LIFE-CYCLE: PLANNING 39

3.1 PROBLEM IDENTIFICATION 41

3.2 SOLUTION TOOLS AND TECHNIQUES 42

3.2.1 Modeling and Simulation 43

3.2.2 Object Oriented Programming (OOP) 44

3.2.3 The Visual Simulation Environment (VSE) *47*

3.3 PROBLEM BOUNDARY AND RESOURCES 48

3.3.1 Problem Boundary *48*

3.3.2 Key players, roles, and functions *52*

3.3.3 Preliminary Project Plan *52*

CHAPTER 4 SIMULATION LIFE-CYCLE: MODELING 54

4.1 SYSTEM DEFINITION AND MODEL FORMULATION 54

4.1.1 Assumptions *54*

4.2 DEFINE MODEL OBJECTS 57

4.2.1 Model hierarchy *67*

4.2.2 Clinic layout *68*

4.3 DYNAMIC OBJECT FLOW 71

4.3.1 Phone Call Dynamic Object Flow *72*

4.3.2 Patient Dynamic Object Flow *72*

4.3.3 Medical Staff Member Dynamic Object Flow *77*

4.3.4 Companion Dynamic Object Flow *78*

4.4 INPUT DATA REQUIREMENTS 79

4.4.1 Queston Information Center *80*

4.4.2 Patient Input data *88*

4.4.3 Random Variate Stream *104*

4.5 DATA COLLECTION 106

4.6 TRANSIENT PERIOD ANALYSIS 113

4.5 OUTPUT STRUCTURE 115

CHAPTER 5 QUESTON SIMULATION MODEL DESCRIPTION 118

5.1 OBJECT CLASSES AND METHODS 118

5.2 DESCRIPTION OF CLASSES, INSTANCE METHODS, AND CLASS METHODS 122

5.2.1 Formating *122*

5.2.2 Queston Simulation User-Defined Classes *124*

CHAPTER 6 VERIFICATION, VALIDATION, AND TESTING 164

6.1 VERIFICATION TECHNIQUES 166

6.1.1 Assertion Checking *167*

6.1.2 Debugging *168*

6.1.3 Desk Checking *171*

6.1.4 Execution Monitoring *172*

6.1.5 Execution Tracing *173*

6.1.6 Special Input Testing *177*

6.1.7 Visualization/Animation *179*

6.2 VALIDATION TECHNIQUES 180

6.2.1 Face Validation *180*

6.2.2 Turing Test *181*

6.2.3 Extreme condition test *182*

6.2.4 Animation/Operational Graphics *183*

6.3 SUMMARY 183

CHAPTER 7 PRELIMINARY EXPERIMENTATION AND DEVELOPMENT 185

7.1 EXPERIMENTATION 185

7.2 PRELIMINARY RESULTS FROM INITIAL BASELINE MODEL 185

7.3 RESULTS FROM THE SECOND BASELINE MODEL 194

7.4 SUMMARY 197

CHAPTER 8 SUMMARY AND RECOMMENDATIONS 199

8.1 SUMMARY 199

8.2 RECOMMENDATIONS 200

REFERENCES **203**

APPENDIX A: PATIENT INPUT DATA FROM DOMAIN EXPERTS **215**

APPENDIX A.1 INITIAL PATIENT CATEGORY INPUT DATA 215

APPENDIX A.2 REVISED PATIENT CATEGORY INPUT DATA 216

APPENDIX B: QUESTON SIMULATION OUTPUT FILES **218**

APPENDIX C: TIME STUDY DATA **225**

APPENDIX C.1 CHRISTIANSBURG, VIRGINIA CLINIC PATIENT VISIT DATA FROM DAY ONE 225

APPENDIX C.2 CHRISTIANSBURG, VIRGINIA CLINIC PATIENT VISIT DATA FROM DAY TWO 227

APPENDIX D: EXPERIMENTATION RESULTS **228**

VITA **233**

LIST OF FIGURES

FIGURE 2.1: USE OF DISCRETE-EVENT SIMULATION OVER THE PAST TWENTY-FIVE YEARS 29

FIGURE 3.1: SIMULATION LIFE-CYCLE DIAGRAM 40

FIGURE 3.2: MESSAGE PASSING 45

FIGURE 3.3: PROTOTYPE OF THE QUESTON SIMULATION MODEL CLINIC 49

FIGURE 3.4: QPN SIMULATION LIFE-CYCLE TIMELINE 52

FIGURE 4.1: VSE PALETTE WINDOW 58

FIGURE 4.2: VSE CONSTANTS PANEL 59

FIGURE 4.3: QUESTONMEDICALPRACTICE OBJECT 60

FIGURE 4.4: MULTIPLE VIEWS OF THE QUESTON SIMULATION MODEL 61

FIGURE 4.5: QUESTON SIMULATION MODEL CLINIC OBJECT 62

FIGURE 4.6: ROOMSAREA OBJECT 63

FIGURE 4.7: HIERARCHY OF THE DEEP STATIC OBJECT IN THE QUESTON SIMULATION MODEL 64

FIGURE 4.8: MEDICAL STAFF OFFICE AREA 65

FIGURE 4.9: DEEP OBJECT HIERARCHY 71

FIGURE 4.10: PATIENT FLOW 74

FIGURE 4.11: TIME SLOT NUMBER AND APPOINTMENT TIMES 82

FIGURE 4.12: CALL INTERARRIVAL RATE INTO THE QUESTON INFORMATION CENTER 85

FIGURE 4.13A: NORMAL VISIT PATIENT CATEGORY COLLECTION FORM 91

FIGURE 4.13B: PRE-VISIT PATIENT CATEGORY COLLECTION FORM 92

FIGURE 4.14: PATIENT SERVICE FLOW 93

FIGURE 4.15: LUNCH BREAK WALK-IN PATIENT POLICIES 97

FIGURE 4.16: APPOINTMENT DAY SEARCH ALGORITHMS 98

FIGURE 4.17: NUMBER OF PATIENTS ADMITTED INTO THE CLINIC PER DAY 103

FIGURE 4.18: PATIENT VISIT DATA COLLECTION FORM 107

FIGURE 4.19: ARRIVAL DEVIATION FROM SCHEDULED APPOINTMENT TIME 112

FIGURE 5.1: QUESTON SIMULATION MODEL OBJECT CLASSES 119

FIGURE 6.1: VSE TRANSCRIPT WINDOW 174

FIGURE 6.2: VSE INSPECTOR 175

FIGURE 6.3: VSE METHOD TRACE WINDOW 176

LIST OF TABLES

TABLE 4.1: ASSUMPTIONS AND MOTIVATION 56

TABLE 4.2: PATIENT'S PROBABILITY OF HAVING COMPANIONS 79

TABLE 4.3: THINNING PROBABILITIES FOR EACH TIME PERIOD 87

TABLE 4.4: DAY OF THE WEEK TABULATION 88

TABLE 4.5: AMERICAN MEDICAL ASSOCIATION'S CATEGORIZATION OF PATIENTS 89

TABLE 4.6: QUESTON SIMULATION MODEL PATIENT CATEGORY DEFINITION 90

TABLE 4.7: TIME SLOT SCHEDULING RULES 100

TABLE 4.8: PROCESSING TIME OF THE PATIENTS 108

TABLE 4.9: GENERAL INFORMATION ON THE CLINIC 109

TABLE 4.10: CHANGE TO THE PATIENT'S ALOS USING THE REVISED PATIENT CATEGORY INPUT DATA 111

TABLE 4.11: WARM UP VALIDATION DATA POINTS 114

TABLE 7.1: EXPERIMENTS 186

TABLE 7.2: BASELINE CONFIGURATION 187

TABLE 7.3: STAFF COMPOSITION FOR PHYSICIAN PRACTICES 191

LIST OF EQUATIONS

EQUATION 4.1: ACCEPTANCE/REJECTION CONDITION 96

EQUATION 4.2: EQUATIONS FOR SAMPLE STATISTICS 109

Chapter 1 Introduction

1.1 The Queston Physician Network

In today's highly competitive healthcare market, physicians are being driven to both cut costs and provide quality healthcare. The advantages associated with these objectives can be realized if primary care physicians ally themselves into a centralized administrative, financial, telecommunication, and information network. The challenge of integrating and coordinating these functions for a new healthcare paradigm is being met by the *Queston Physician Network* (QPN), a subsidiary of Biological and Popular Culture, Inc. (Biopop). The QPN is Biopop's unique system of centralized management and revenue enhancement for networks of primary care physicians, employing the latest in office automation, information systems, and medical technology. The QPN attempts to improve operating margins by reducing network operational inefficiencies and overhead through an increased reliance on centralized information technology and economies of scale, as well as improving its market position by creating brand value and awareness. Physician clinics in the QPN receive centralized call routing and scheduling, patient billing, clerical staffing, automatic order generation,

marketing, and all other essential administrative and financial services necessary to operate their clinic.

The QPN is a comprehensive marketing and operations management system retained by physician clinics located throughout the United States. The objective of joining the QPN is to improve a physician's competitive advantage in their geographical area of operations. The QPN does not retain ownership of the physician practices. The physician network is managed by the Queston Information Center located in Radford, Virginia. This center conducts all non-medical affairs that are typically handled by on-site clerks, such as patient scheduling via an 800 number, patient billing, and insurance claims. This allows physicians to devote their time to providing the best quality medical care to patients, while increasing patient satisfaction and throughput.

1.2 Purpose of the Model

The purpose of this research is to create a discrete-event simulation model of the QPN that will assist Biopop in evaluating the operating procedures and the resource utilization of the participating clinics and the Queston Information Center. More specifically, the research will attempt to model the detailed operations of a physician practice, hence provide a tool to design the optimal patient-physician encounter (i.e., maximize patient throughput and minimize costs per patient), without sacrificing patient satisfaction and quality of care.

Clinics partnered through the QPN can benefit from the results of this analysis, for determining staffing requirements and identifying operating efficiencies.

The model is built using a discrete event, object-oriented, domain-independent, visual simulation software, the Visual Simulation Environment (VSE). Developed by Orca Computer, Inc., VSE is a powerful software tool for creating highly complex discrete event simulation models. The object-oriented paradigm, the picture-based format, and the English-like scripting features of VSE contribute to the ease of codification and verification of the Queston model.

1.3 Summary of the Chapters

A brief introduction of the QPN, as well as the purpose and objectives of the research are presented in this chapter. Chapter 2 presents a literature on the application of simulation in the area of health care, with a particular focus on single and multi-facility health care clinics. Problem identification, solution tools, solution techniques, and the initial planning for the simulation model are described in Chapter 3. In Chapter 4, the modeling and the identification of the input data for the simulation model are discussed. The individual objects of the simulation model are described and a brief summary of the model's code are given in Chapter 5. Chapter 6 describes the procedures and techniques used for the verification of the code and the validation of the results. Preliminary experimentation and results are presented

in Chapter 7. Conclusions and recommendations for future research are discussed in Chapter 8.

Chapter 2 Literature Review

Over the past thirty years, the dramatic increase in the cost of health care has compelled researchers and health care professionals to examine ways to improve operational efficiency and reduce costs. Discrete-event simulation is one tool available to health care decision makers that can assist in this endeavor. Discrete-event simulation is an operations research technique that allows the end user (i.e., hospital administrator, clinic manager) to assess the efficiency of existing health care delivery systems, to ask "what if?" questions, and to design new systems. Discrete-event simulation can also be used to forecast the impact of changes in patient flow, to examine resource needs (either in staffing or in physical capacity), and to investigate the complex relationships among the different model variables (e.g., rate of arrivals, rate of service). This information allows managers to select management alternatives that can be used to reconfigure existing systems, to improve system performance, or to design and plan new systems, without altering the present system.

In recent years, the application of discrete-event simulation in health care has become increasingly more wide spread. This may be attributed to the numerous successful studies reported using computer simulation to address health care system problems and the ever-increasing sophistication of simulation software packages. To gain a better appreciation for

how extensively discrete-event simulation has been used to address health care clinic problems, a survey of the literature was conducted.

Though numerous health care simulation models have been developed during the past thirty years, this chapter focuses on articles that analyze single or multi-facility health care clinics (e.g. outpatient clinics, emergency departments, surgical centers, orthopedics departments, and pharmacies). This chapter provides an extensive taxonomy of the literature over the past twenty years that is presented by Jun et al. [1999]. Simulation studies on wide area or regional health community planning, ambulance location service, gurney transportation, disease control planning, and studies that do not address patient flow are not discussed. For a review of the literature in health care prior to the mid-seventies, see England and Roberts [1978] and Valinksy [1975].

Several review papers [England and Roberts 1978, Klein et al. 1993, Smith-Daniels et al. 1988, Valinksy 1975] and tutorials [Banks and Carson 1987, Kanon 1974, Lowery 1996, Mahachek 1992] have been written about conducting a simulation project in health care clinics. England and Roberts [1978] give an excellent comprehensive survey on the application of simulation in twenty-one health care areas, from laboratory studies to emergency services to the national health care system. They cite ninety-two simulation models out of twelve hundred models reviewed, including all published models developed through 1978. Klein et al. [1993] present a bibliography that includes operational decision making, medical decision making, and system dynamics planning models. Smith-Daniels et al. [1988] review the literature pertaining to acquisition decisions (e.g., facility location,

aggregate capacity, and sizing of facility) and allocation decisions (e.g., inpatient admissions scheduling, surgical facility scheduling, and ambulatory care scheduling). Their review includes several methodologies, such as heuristics, Markov chains, linear programming, and queueing theory, as well as simulation.

Banks and Carson [1987] and Mahachek [1992] provide tutorials on the steps required when conducting a health care system simulation study. Mahachek also provides details of a simulation study on hospital patient flow. Kanon [1974] shows the relative ease with which one could use sample data to build a simulation model of a simplified problem in a hospital setting. Lowery [1996] discusses some of the issues facing an analyst when using simulation to study a health care system, such as degree of model complexity, definition of input distributions, model validation, and interpretation of findings. All these articles provide useful information for practitioners interested in using discrete-event simulation to study health care systems and issues. Moreover, the articles focus on a common theme, namely the unique factors inherent in health care systems, and how simulation can be used to address such systems.

The chapter is organized as follows: Chapter 2.1 discusses the impact of patient scheduling and admissions, patient flow schemes, and staff scheduling on patient flow and work flow. Chapter 2.2 examines the allocation of resources when sizing and planning beds, rooms, and staff personnel. Chapter 2.3 discusses the lack of complex integrated multi-facility systems, advances in visual simulation, use of both simulation and optimization techniques, and issues

in implementation. Finally, a summary of simulation modeling in health care clinics is presented.

2.1 Patient Flow

Hospitals and clinics are facing increasing competition for their services. To attract new patients and retain their patronage, hospitals and clinics must be able to provide fast and efficient health care. Effective and efficient patient flow is indicated by high patient throughput, low patient waiting times, short length of stay at the clinic, and low average daily clinic overtime, while maintaining adequate staff utilization rates and low physician idle times. Three areas that impact patient in clinics are:

i) patient scheduling and admissions,

ii) patient routing and flow schemes,

iii) scheduling and availability of resources.

Each of these three issues will now be looked at in greater depth.

2.1.1 Patient Scheduling and Admissions

Patient scheduling and admissions focus on procedures that determine how patient appointments (with a medical staff member) are scheduled, both in terms of when and how they are set in a given day, and their length of time. More specifically, this involves

scheduling rules that determine when appointments can be made (e.g., morning versus afternoon) and the length (spacing) of time between appointments. This may also be extended to include designating the specific types of medical staff member who will be responsible for treating patients and the clinic space that will be required to deliver the necessary treatments. These issues can have a significant impact on how resources (e.g., physicians, staff, facilities) can be optimally utilized so as to maximize patient flow (hence increase profitability) without incurring additional costs or excessive patient waiting.

Most simulation studies that focus on patient scheduling and admissions are for outpatient clinics. Fetter and Thompson [1965] present results of one of the earliest simulation studies conducted in the area of individual clinical facility operations for outpatient clinics. They analyze the physician utilization rate with respect to patient waiting time by using different input variables (such as patient load, patient early or late arrival patterns, no show rates, walk-in patient rates, appointment scheduling intervals, physician service times, interruptions, and physician lunch and coffee breaks). They determine that if the physician appointments increase from 60% to 90% (capacity), the physician idle time decreases by 160 hours and patient waiting time increases by 1600 hours (over a fifty day period). If this capacity increase were to be implemented, the simulation study suggests that the physician's time would have to be worth ten times the patient's time to justify such a shift in patient scheduling and admission policies.

Evenly distributing patient demand has been used to improve patient throughput and patient waiting times in outpatient clinics. Smith and Warner [1971] compare the case when patients

arrive according to a uniform arrival pattern versus patient arrival patterns that are highly variable. They show that the uniform arrival pattern can decrease the average length of stay at a clinic by over 40% (from 40.6 minutes to 24 minutes). Similarly, Rising et al. [1973] increase the number of appointments slots in an outpatient clinics on those days that had the least number of walk-in patients, thereby smoothing demand on the physician. Their results show a 13.4% increase in patient throughput and less average daily clinic overtime. Kho and Johnson [1976] and Kachhal et al. [1981] show that in a radiology department and in an ear, nose, and throat clinic, respectively, performance can be improved when demand for outpatient services is evenly distributed.

In contrast to uniform scheduling, alternative scheduling rules have also been investigated. Bailey [1952] develops an outpatient clinic scheduling rule that yields acceptable results for both patients (in terms of waiting times) and staff (in terms of utilization), assuming that all patients have the same service time distributions and that all patients arrive punctually (i.e., at their designated times). The rule schedules two patients at the beginning of every session (morning or afternoon), with all other patients scheduled at equal intervals. In a similar study, Smith et al. [1979] use a modified-wave scheduling scheme for an outpatient clinic to find the maximum number of patients a physician could see while minimizing patient waiting time. This scheme schedules more patients at the beginning of each hour and less towards the end of the hour, thus allowing the physician to absorb unexpected delays and return back to schedule at the end of each hour. They show this schedule to be superior to the uniform scheduling scheme, in terms of patient flow and patient waiting times. Williams et al. [1967] study the relationship between physician utilization and patient waiting time in an outpatient

clinic using a staggered block scheduling system (eight patients arriving every half hour) versus the single block scheduling system (sixteen patients arriving simultaneously). The single block system emphasizes the physician's idle time, whereas, the staggered block system emphasizes the patient's waiting time, hence a substantial decrease in the patient waiting times (with no decrement in the utilization of the physician) occurs with the staggered system.

Surgical (operating room) center scheduling has also been studied using simulation. Murphy and Sigal [1985] examine surgical block scheduling, where a block of operating room time is reserved for an individual surgeon or a group of surgeons. Fitzpatrick et al. [1993] study the use of first-come-first-serve, fixed, variable, and mixed block scheduling for hospital operating rooms. Fixed block scheduling is defined as scheduling the same block of time in the same time slot each day of the week. Similarly, variable block scheduling is scheduling under the influence of seasonal fluctuation in demand. Mixed block scheduling means using a fixed block for those procedures that are time consuming or require specialized set-ups, while using a variable block for all other procedures. They find that variable block scheduling is superior to the other block scheduling rules, in terms of facility utilization, patient throughput, average patient waiting time, and patient queue length. Magerlein and Martin [1976] review the literature (prior to 1980) on using simulation for scheduling surgical centers.

Klassen and Rohleder [1996] use simulation to study the best time to schedule patients with large patient service time differences and variances. They analyze several rules and arrive at

the best result (which is to schedule such patients towards the end of the appointment session) that minimizes the patient's waiting time and the physician's idle time. Additionally, they analyze the best position for unscheduled appointment slots for potentially urgent calls and found no conclusive scheduling rule. Likewise, Swisher et al. [1997] experiment on scheduling more patients with larger service distribution in the morning session, rather than the afternoon, in an outpatient clinic. They find that staff overtime decreases sharply but the amount of time the physician has for lunch also decreases. Another study concerning overtime is conducted by Steward and Standridge [1996] in a simulation model of a veterinary practice. The veterinary domain is very similar to the larger human medical systems domain, since both involve the issues of flow and service rates, as well as resource utilization, staffing, demand, and scheduling. In this study, the output measure of interest is the average time interval between the closing time of the clinic and the time the last client is discharged after clinic hours. This serves as an indicator of overhead cost and client satisfaction. The results from the study indicate that performance can improve if the clinic disallows the scheduling of appointments less than 90 or 120 minutes prior to closing, rather than 60 minutes, as was the current practice.

Hancock and Walter [1979] attempt to use simulation to reduce variance in occupancy levels in a hospital inpatient facility, with the goal of increasing patient throughput and maximizing average occupancies. Unfortunately, they were unsuccessful in achieving their stated objective, since the staff was accustomed to admitting patients on the date of the requests 90% of the time (and they refused to schedule over four weeks in advance). In another paper, Hancock and Walter [1984] attempt to smooth the daily patient loads of nineteen hospital

departments by varying the admission days of urgent inpatient and outpatient loads. The variation in average load for each of the departments led them to conclude that no one single admission policy could provide a stable workload for all departments, since each department had its own unique patient arrival patterns and treatment requirements, including different inpatient and outpatient requirements.

Lim et al. [1975] apply two admission policies (quickcall and maximum queue lengths) to a simulation model of an inpatient orthopedics ward. Quickcall is defined as a patient willing to enter the hospital on very short notice; whereas, maximum queue lengths is a concept in which the physicians are required to maintain a maximum number of patient requests on a waiting list. Both systems improved system performance, in terms of patient waiting times and staff utilization.

Using several different appointment schemes, Walter [1973] describes several aspects of a queueing system in a radiology department. By segregating patients into inpatient and outpatient sessions with a similar examination time distribution, he found that a substantial staff time savings was possible. He also found that the practice of multiple bookings for a given appointment time (i.e., overbooking) yields a small increase in staff utilization while substantially increasing the patient waiting time. Additionally, efficiency always improves when the proportion of patients with appointments increases, resulting in a smoothing of the arrival rate. Goitein [1990] supports these conclusions using Monte Carlo simulation to examine factors, for example physician idle time versus patient waiting time. He found that if the physician overbooked the schedule (even slightly), patients would experience very long

waiting times. His model provides insights into how delays build up as a result of statistical fluctuations.

In conclusion, patient scheduling and admission rules along with patient appointment timing can significantly impact physician utilization and patient waiting. In general, the studies using simulation discussed here suggest that rules and policies can be employed that balance the tradeoff between physician utilization rates and patient waiting times, though unique features of each clinic environment need to be taken into account to determine the exact extent of these tradeoffs.

2.1.2 Patient Routing and Flow Schemes

One advantage of using discrete-event simulation over other mathematical modeling tools (such as linear programming and Markov chain analysis) when modeling a health care clinic is the capacity of simulation to model complex patient flows through health care clinics, and then to play "what if" games by changing the patient flow rules and policies. Such flows are typical in emergency room settings, where patients arrive (without appointments), and require treatment over a large and varied set of ailments. These ailments can range from the benign (e.g., mild sports injuries) to the fatal (e.g., heart attacks, gunshot wounds). Though the arrival of patients is highly unpredictable, the sequence by which patients can be treated (i.e., routed) can be controlled by medical staff member. By altering patient routing and flow, it may be possible to minimize patient waiting times and increase staff utilization rates.

Garcia et al. [1995] analyze the effects of using a fast track lane to reduce waiting times of low priority patients in an emergency room. Emergency rooms are prioritized according to the level of patient sickness, hence low priority patients regularly wait for excessively long periods of time. A fast track lane is a special queue dedicated to serving a particular level of patient (in this case, non-urgent patients). They found that a fast track lane that uses a minimal amount of resources could greatly reduce patient waiting times. In a simulation model of the emergency department at the University of Louisville Hospital, Kraitsik and Bossmeyer [1993] suggest using a fast track lane as well as using a "stat" lab for processing high volume tests to improve patient throughput. Kirtland et al. [1995] examine eleven alternatives to improve patient flow in an emergency department and select three alternatives, that when combined, can save each patient an average of thirty eight minutes. The three selected alternatives were:

i) using a fast track lane in minor care,
ii) placing patients in the treatment area instead of sending them back to the waiting room,
iii) using point-of-care lab testing.

McGuire [1994] uses MedModel to determine how to reduce the length of stay for patients in an emergency service department in a SunHealth Alliance hospital. From the simulation study results, several alternatives were recommended, including adding an additional clerk during peak hours, adding a holding area for waiting patients, extending the hours of the fast track lane, and using physicians instead of residents in the fast track area. Blake et al. [1996] also analyze an emergency department at the Children's Hospital of Eastern Ontario. Based

on their simulation results, a fast track lane for treating patients with minor injuries was implemented.

Ritondo and Freedman [1993] show that changing a procedural policy (of ordering tests while in triage) results in a decrease in patient waiting times in the emergency room and an increase in patient throughput. Edwards et al. [1994] compare the results of simulation studies in two medical clinics that use different queueing systems: serial processing, where patients wait in a single queue, and quasi-parallel processing, where patients are directed to the shortest queue to maintain flow. They show that patient waiting times could be reduced by up to 30% using quasi-parallel processing. Pardue and Cognetta [1995] use a SLAM based simulation model of a dermatology clinic to estimate benefits of a real-time (simultaneous) approach to the treatment of skin cancer. By conducting three out of four steps of the treatment process in a single visit, the setup cost and the inconveniences of multiple visits were decreased and overall patient satisfaction increased.

2.1.3 Scheduling and Availability of Adequate Resources

The majority of scheduling simulations for health care clinics are directed at patient scheduling (so as to distribute patient demand for the physician and the staff). A number of studies, however, have addressed the problem from the opposite side (i.e., staff can be scheduled to meet patient demand, while the patient arrivals can be left unchanged). In fact, some clinics, such as walk-in patient clinics, are unable to change the arrival rate of patients and must schedule their staff accordingly. Furthermore, concurrent scheduling of staffing, as

well as scheduling for the patients, could be used to better meet demand and better allocate resources. Incorporating this idea, Alessandra et al. [1978] study both the staffing levels and patient arrival rates to ease bottlenecks and to improve patient throughput. Eight alternatives that involve varying the staffing pattern and the patient scheduling scheme were analyzed. It was found that the best alternative was to keep the staffing and arrival rate the same, but to distribute the current morning appointment patients to the afternoon shift. Mukherjee [1991] identifies a staffing mix that reduces patient waiting time and increases patient throughput, while controlling resource costs in a pharmacy.

A number of simulation models of scheduling nursing staff in emergency departments have been developed, due to the high volume of emergency visits, as well as the urgency of the care required. Furthermore, hospitals are faced with pressures to maintain high quality health care while reducing or minimizing costs. Draeger [1992] simulates nurse workload in an emergency room and their effect on the average number of patients, average time in system, average number of patients waiting, and average patient waiting time. Comparing the current schedule's performance to those of two alternative staffing schedules, they found an alternative that could reduce both the average patient time in system (by 23%) and the average patient waiting time (by 57%), without increasing costs. Similarly, Evans et al. [1996] reduce a patient's length of stay by finding the optimal number of nurses and technicians that should be on duty during four shift periods in an emergency room. Kumar and Kapur [1989] examine ten nurse scheduling policy alternatives, selecting and implementing the policy yielding the highest nurse utilization rate.

Lambo [1983] applies a recursive linear programming and simulation methodology to examine staffing problems in a health care center in Nigeria. In the study, the clinic was observed to be at 50% capacity due to the misallocation of (rather than the inadequacy of) personnel. After making changes to the staffing patterns and other policy changes, capacity increased by 60% and patient waiting times were reduced by 45 minutes.

These studies suggest that when patient flow patterns cannot be controlled, staffing strategies can be employed to mitigate some of the unavoidable variability in the systems. This can result in improved patient throughput, while keeping staff utilization rates at acceptable levels.

2.2 Allocation of Resources

With the rise in the cost of providing quality health care, hospital and clinic administrators have approached cost containment by minimizing resources for health care provisions while still striving to provide quality health care for patients. This predicament is becoming more prevalent in the health care community as indicated by the large body of literature that analyzes the allocation of scarce health care resources. Simulation modeling is attractive since it can be used to estimate the operational characteristics of a system as well as to observe the consequences of changes in planning or policies prior to when decisions are actually implemented, hence reducing the financial risks. The allocation of resources can be

divided into three general areas: i) bed sizing and planning, ii) room sizing and planning, and iii) staff sizing and planning.

2.2.1 Bed Sizing and Planning

The demand for hospital or clinic beds can be classified as either routine (i.e., scheduled) or emergency (i.e., unscheduled) admissions. Both these types of admissions impact how many beds are needed to meet demand, while maintaining reasonable bed utilization rates. In the literature, most bed planning simulation models attempt to overcome bed shortages or policies that lead to patient misplacement, bumping, or rejection. Hospitals are faced with the tradeoff between having available beds to service patient demand versus high bed occupancy (utilization) rates.

Butler et al. [1992a] use simulation to study patient misplacements, where patients are scheduled and assigned to an alternative unit within a hospital due to a shortage of beds in the preferred hospital area. They examine the sensitivity of patient misplacement to various modifications in their bed allocation policy, including patient transfers, bed scheduling, and assignments. It was found that reducing a patient's length of stay and reallocating rooms among the different services within a hospital could substantially decrease patient misplacement. Furthermore, the smoothing of routine patient arrivals may only marginally reduce patient misplacement. In another study designed to reduce patient misplacement, Butler et al. [1992b] use a two-phase approach involving a quadratic integer programming model and a simulation model to evaluate bed

configurations and to determine optimal bed allocations across a number of hospital service areas.

Lowery [1992, 1993] and Lowery and Martin [1992] study the use of simulation in a hospital's critical care areas (e.g., operating rooms, recovery units, intensive care units, and intermediate care units) and to determine critical care bed requirements. Their literature review reveals that most models do not fully consider the interrelationships between different hospital units, and few models have been validated using actual hospital output measures. Focusing on these deficiencies, they demonstrate improvements in their methodologies over previous models. Dumas [1984, 1985] also focuses on the interrelationships between several units within a hospital by comparing two bed planning rules (vacancy basing and home basing) for locating a bed within different hospital units when a patient cannot be allocated a bed at the preferred unit. Vacancy basing rules utilize a ranked list of alternative misplacement possibilities, while home basing prohibits off-service misplacements, hence is more restrictive with respect to patient placement. They show that home basing policies result in better overall performance but less patient days, hence less revenue. Cohen et al. [1980] present a bed planning model of a progressive patient care hospital, where patients are moved between units within a hospital as their condition changes. They demonstrate that the probability of inappropriate patient placement (i.e., placing patients into the wrong unit) is a function of the capacities of all the units, as well as the policies for handling priority patients and bumped patients.

Looking at individual units within a hospital, Zilm et al. [1983] use simulation to model a surgical intensive care unit for various bed levels and future demand. They observe that most of

the unit's volume consist of weekday cases (routine admissions) and consequently any attempt at a high overall occupancy level would not be possible without straining the system. Romanin-Jacur and Facchin [1987] use simulation to study the facility dimensioning problem and the sizing of the assistance team in a pediatric semi-intensive care unit. They compare several different priority-based models by using peak admission conditions to find the optimal number of beds and the best choice of the nurse's turn of duty (i.e., number of nurses on duty at different periods of time). Other bed sizing models include Hancock et al. [1978], Wright [1987], and Harris [1985]. Harris [1985] compares the difference in the number of beds needed in a surgical center for three physicians under two operating timetable scenarios. Under the first (and current) scenario, each physician scheduled their patients independent of the other two physicians. In the second scenario, the physicians pooled their resources to schedule their patients and consequently reduced the number of beds required by over 20% (from 62 to 49).

Gabaeff and Lennon [1991] use an extensive time-motion study to collect data on the mix of patient types, patient characteristics (such as x-ray requirements), and staffing mix for emergency admissions in an emergency department feasibility study at Stanford University Hospital. Using simulation models, they highlight deficiencies in a number of key areas, such as maximum bed utilization exceeding current bed availability, which would cause displacement of minor care patients. Vassilacopoulos [1985] develops a simulation model to determine the number of beds with the following constraints: high occupancy rates, immediate (emergency admission) patients, and low length of waiting lists. He shows that by using a waiting list and smoothing the patient demand, high occupancy rates could be achieved. Altinel and Ulas [1996] simulate emergency department bed planning at the Istanbul University School of Medicine; Freedman [1994] at St.

Joseph Hospital and Washington Adventist Hospital in Maryland, USA; Lennon [1992] at the Stanford University Hospital; and Williams [1983] at the University of Pennsylvania Hospital.

In conclusion, simulation provides a valuable "what if" tool for hospital planners when deciding how many beds are needed to meet demand and maintain profitability. Moreover, simulation allows hospital administrators to experiment with different bed allocation rules to help optimally utilize hospital facilities and improve bed occupancy rates.

2.2.2 Room Sizing and Planning

The continuing trend towards the development of freestanding surgicenters, as well as the movement to deliver health care services away from inpatients facilities and more towards outpatient facilities, has put increased pressure upon hospital management to expand their outpatient services and/or to build new facilities to handle these additional patients. The use of computer simulation for the planning of future expansion, integration, and/or construction of new facilities and departments has greatly enhanced the hospital administration decision maker's ability to find the most cost effective and efficient solutions to remain competitive.

The number of and utilization of operating rooms is often an important resource in maintaining hospital profitability and patient services. Currie et al. [1984] study operating room utilization, vertical transportation needs, radiology staffing, and emergency medical system operations at the West Virginia University Hospital. They arrive at an estimate for the number of operating rooms and recovery beds needed to handle a 20% increase in future demand. Kwak et al. [1975] use simulation to determine the capacity of a recovery room for

an operating room expansion. Similarly, Kuzdrall et al. [1981] use a simulation model of an operating and recovery room facility to determine and assess the facility utilization levels and facility needs under different scheduling policies. Olson and Dux [1994] apply simulation modeling to study the decision to expand the Waukesha Memorial surgicenter from seven to eight operating rooms. The study reveals that an eighth operating room would only serve to meet the hospital's needs for one to two years, at a cost of $500,000. However, an analysis of the cross-departmental and administrative needs reveal that an ambulatory surgery center that separates the inpatients and outpatient procedures would better serve the hospital's future health care delivery needs. Likewise, Amladi [1984] uses simulation to assist in the sizing and planning of a new outpatient surgical facility, by considering patient wait time (quality) and facility size (resource).

Meier et al. [1985] consider eleven scenarios in varying the number of examination rooms and demand shifts of both a hospital ambulatory center and a freestanding surgicenter. They found that existing room capacity is adequate to handle demands forecasted over the next five years. Iskander and Carter [1991] also found that current facilities were sufficient for future growth in a study of a same day (outpatient) health care unit in an ambulatory care center. However, they suggest a threefold increase in the size of the waiting room.

Kletke and Dooley [1984] examine the effects on service level and utilization rates in a maternity ward to determine if the current number of labor rooms, delivery rooms, post-partum rooms, nursery, and nurses are able to meet future demands. Their simulation study

recommends increasing the number of labor rooms and the number of post-partum rooms, while maintaining four full-time nurses.

Levy et al. [1989] analyze the operational characteristics of an outpatient service center at Anderson Memorial Hospital to determine whether to merge this service with an offsite outpatient diagnostic center. They collected data on the utilization of the servers, the total number of patients in the center, the maximum and average times spent in the center, the maximum and average times spent in each service queue, and the total number of patients in each queue. This information was used to specify staffing and facility sizing requirements. In another facility integration plan, Mahachek and Knabe [1984] simulate a proposal to combine an obstetrics clinic and a gynecology clinic into one single facility in order to cut costs. The analysis using simulation found that this proposal would not be successful due to the shortage of examination rooms.

In conclusion, simulation provides a valuable tool in determining how to set the size of key hospital facilities (such as operating rooms). As the health care industry continues to move more towards outpatient delivery systems, and away from traditional inpatient health care facilities, simulation will also be useful in assessing how to best undertake this transition.

2.2.3 Staff Sizing and Planning

Several simulation studies have been conducted to determine the staff size or the number of physicians for emergency departments [Carter et al. 1993, Badri and Hollingsworth 1993,

Klafehn and Owens 1987, Klafehn et al. 1989, Liyanage and Gale 1995]. Badri and Hollingsworth [1993] analyze the impact of different scenarios on scheduling, limited staffing, and changing the patient demand patterns in an emergency room of the Rashid Hospital in the United Arab Emirates. Several different scenarios were investigated:

i) using a priority rule based on severity of ailment,

ii) not serving a category of patient that does not belong in the emergency room,

iii) eliminating one or more doctors on each shift,

iv) a combination of ii) and iii).

The results from scenario iv) were accepted and implemented. Klafehn and Owens [1987] and Klafehn et al. [1989] address the linkage between patient flow and the number of staff available in an emergency department. They conclude that moving one nurse from the regular emergency area to a triage position reduces patient waiting lines and patient waiting times. Furthermore, the addition of a second orthopedic team in the emergency department increases patient flow, though utilization levels were lower and the average length of stay remains virtually the same (since the number of patients flowing through the orthopedic area was relatively small). Liyange and Gale [1995] develop an M/M/n queueing model of the Campbelltown Hospital emergency facility. They use the model to estimate and develop patient arrival time distributions, patient waiting times, and patient service times. These parameters were then used in a simulation model to estimate the expected waiting time of the patients, the expected idle time of the doctors, and the optimal number of doctors. Gonzalez et al. [1994] evaluate eight alternative scenarios in an emergency department simulation, by varying the number of staff, the arrival rates, and the service times of diagnostic equipment

(alterable by purchasing better equipment). They recommend that the arrival rate should not exceed twelve patients per hour. Moreover, they recommend that investments in human resources would be more effective than in newer (better) equipment. In contrast, Bodtker et al. [1992] and Godolphin et al. [1992] determine that a reduction in staff by at least one staff member could be achieved if better equipment were purchased.

O'Kane [1981], Klafehn [1987], and Coffin et al. [1993] analyze staff allocations to improve patient flow in a radiology laboratory. Klafehn and Connolly [1993] model an outpatient hematology laboratory using Proof Animation from Wolverine Software. They compare a number of staffing configurations and found that if the staff is cross-trained (hence can be more fully utilized), then patient waiting times can be reduced. Vemuri [1984] and Ishimoto et al. [1990] explore the operations of a pharmacy unit in a hospital. Using simulation, they find the optimal medical staff size and mix that reduces patient waiting times.

Hashimoto and Bell [1996] conduct a time-motion study to collect data for a simulation model of an outpatient (general practice) clinic. They show that increasing the number of physicians, and consequently the number of patients, without increasing the support staff, would significantly increase the length of stay for the patients. By limiting the number of physicians to four and increasing the number of dischargers to two, they were able decrease the patient's average total time in the system by almost 25% (from 75.4 minutes to 57.1 minutes). Wilt and Goddin [1989] evaluate patient waiting times to determine appropriate staffing levels in an outpatient clinic. McHugh [1989] examines the staffing adequacy of various nurse-staffing levels and their effects on cost, understaffing, and overstaffing in a

hospital. Her analysis shows that 55% of the maximum workload produces a good balance between the three measures. Swisher et al. [1997] discuss a simulation model of the Queston Physician Practice Network where individual family outpatient clinics are modeled and integrated into a network of clinics, with a central appointment scheduling center located in Virginia, USA. Output measures such as patient throughput, patient waiting time, staff utilization, and average daily clinic overtime are analyzed for various numbers of examination rooms and staff sizes. In certain cases, adding additional support personnel had negligible effects on the output measures.

Stafford [1989] and Aggarwal and Stafford [1976] develop a multi-facility simulation model of a university health center, incorporating fourteen separate stations (e.g., receptionist area, injections, dentist, gynecology, physical therapy, radiology, and pharmacy). Using student population figures and seven output measures, they were able to estimate the level of demand for services in the clinic. They also show that patient inter-arrival times are distributed negative exponential with the mean changing according to the time of day, and patient service times are distributed Erlang-K. Using this data, they investigate the effects of adding another pharmacist to the pharmacy. A multi-factor experimental design was developed to examine the relationships between the controllable system variables and the system output variables. They show that different calling population sizes and different levels of staffing can impact the output measures at each station. Additionally, the aggregation of two or more similar facilities can cause an increase in the average number of patients waiting at each of the remaining facilities and the average waiting times of the patients. However, these increases were offset by a significant decrease in the staff idle times and staff costs.

In conclusion, staffing levels and staff distributions have a significant impact on patient throughput and patient waiting times. Similar to facility sizing and planning, simulation can be used as an effective tool to study various staffing strategies for a wide variety of health care facilities and systems.

2.3 Discussions for Future Directions

There is a significant amount of literature on using simulation to study health care clinics. Figure 2.1 depicts the trend in the number of publications based on such studies. Figure 2.1 shows this number increasing from eight in 1973-1977 to twenty-eight in 1993-1997. Moreover, this positive trend can be attributed to the increasing demand for cost cutting in health care, coupled with an increase in the ease-of-use and power of simulation software packages (especially over the past five years). Furthermore, a growing number of these studies attempt to combine optimization techniques with discrete-event simulation. Future modelers are moving towards realizing the advantages of simultaneously applying simulation and optimization techniques. Despite the upward trend of health care simulation studies and the integration of discrete-event simulation and optimization techniques, there is still a void in the literature focusing on complex integrated systems. This void may be due to the associated complexity issues and resource requirements. Moreover, no matter how complex the modeled system is or what techniques are applied, future modelers will continue to face difficulties implementing their results. However, recent advances in simulation software may

alleviate some of these difficulties. The following subsections are devoted to a discussion of these topics and issues.

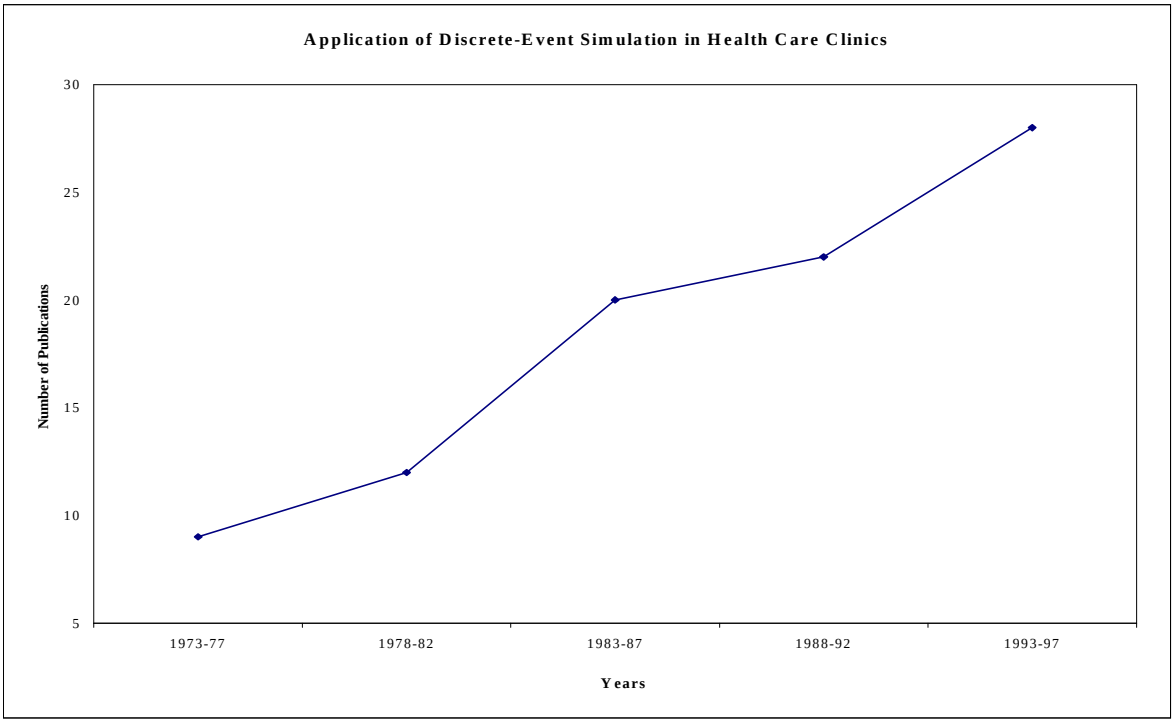


Figure 2.1: Use of discrete-event simulation over the past twenty-five years

2.3.1 Complex Integrated Multi-facility Systems

A limited number of simulation models have been developed that analyze complex multi-facility health care delivery systems. Most simulation models, such as the ones described previously, report on individual units within multi-facility clinics or hospitals. Using a macroscopic analysis of multi-facility systems, simulation can be used to estimate patient demand load (directly related to arrival rates), utilization of staff, and overall costs. The estimation of these output measures may not be possible in a microscopic, single level model, due to the duplication of and overlapping of facilities and services. Simulation models that

depict the interaction of major service departments and support services in a hospital, and the information that can be gained from analyzing the system as a whole, can be invaluable to hospital planners and administrators.

To remain competitive in today's market, the health care industry is being forced to integrate hospitals and clinics, especially the ever-growing number of ambulatory care facilities, into health maintenance organizations (HMO), multi-hospital, or multi-clinic organizations. This presents a challenging application for simulation: to operate these networks of clinics or departments efficiently and cost effectively. Studies in multi-facility simulation models have been conducted by Rising et al. [1973], Aggarwal and Stafford [1976], Hancock and Walters [1984], Swisher et al. [1997], and Lowery and Martin [1993].

One benefit from simulating integrated systems is the more realistic representation of the system under study, hence greater confidence in the results. Though this may not be significant when analyzing a small system, the consequences of invalid results or the lack of a thorough study may lead to a costly decision for large multi-million dollar organizations. With this potential benefit, the question that has to be asked is why is there a lack of literature in this area? The answer may lie in one or both of two issues:

i) the level of complexity and resulting data requirements of the simulation model,
ii) the resource requirements, including time and cost.

A widely recognized guideline in simulation modeling is to keep the model as simple as possible while capturing the necessary measures of interest. This is reiterated by Dearie et al.

[1976] who stress the importance of capturing only relevant output variables when creating a simple, though not necessarily the most complete model. They suggest that it is best to depict the various subsystems at the lowest level of complexity such that the model is accurate while providing information that is easily interpreted. Additionally, Lowery [1996] suggests using simple analytical models if they can provide the level of detail required. However, when analyzing integrated systems, most studies require a level of detail that will far exceed the complexity and demands of analytical techniques. Therefore, care must be taken when determining the required level of detail (since more detail typically means that more data must be collected). Lehaney and Paul [1994, 1995] and Lehaney and Hlupic [1996] suggests using soft system methodology (SSM) to aid in determining the level of detail, identifying system boundaries, and ascertaining system activities, particularly in complex models. Through increased participation of the users/customers, SSM encourages acceptability of the model, its results, and eventually the model's implementation.

Resource requirements, such as the length of time, the cost, and the skills necessary to complete the project, must be fully considered before commencing any (large scale) simulation projects. Today's health care environment is rapidly changing and if the process of developing and searching for a solution requires a large investment in time, the system may be outdated before the simulation becomes useful. Consequently, an adequate amount of resources must be dedicated to the project to insure completion of the study in a reasonable length of time. For example, the cost of collecting the required data (in terms of time and money), the cost of purchasing a simulation software package that would ease the

development of complex models, and the cost of skilled consultants or in-house engineers may all be prohibitively high.

2.3.2 Combining Simulation and Optimization Techniques

Unlike analytical methods, simulation is not an optimization tool. Simulation can only provide an analyst with estimates of output measures for various system alternatives. Moreover, simulation models typically have several output measures upon which to optimize, hence creating a multi-criteria objective function environment.

There are several advantages and disadvantages of using either simulation methodology or optimization techniques to model complex systems. Davies and Davies [1994] and Stafford [1978] compare simulation modeling to several techniques, such as Markov chain analysis, semi-Markov chain analysis, input-output analysis, and queueing analysis of an outpatient clinic. They find that simulation is particularly suitable for modeling health care clinics due to the complexity of such systems. This is because many optimization techniques, such as linear programming, have a limited capacity for characterizing the complexities of medical systems. An optimization technique may require too many unrealistic assumptions about the process, hence rendering the solution invalid. For example, optimization models cannot be used to study the dynamic behavior of a medical clinic, such as appointment scheduling, service routing, and service priorities, which can be easily handled by simulation. On the other hand, many optimization models require only one experimental run to produce optimal or near optimal solutions, though the complexity of the model may result in an intractable

solution; whereas, simulation may require a large amount of effort in time, cost, and data needs. For all of these reasons, operations researchers have attempted to combine simulation with deterministic operations research techniques, such as linear programming, to capitalize on the advantages of using both techniques simultaneously.

Several health care analysts have successfully combined these techniques to find the best staffing allocations and facility sizes [Butler et al. 1992b, Butler et al. 1992c, Carlson et al. 1979, Kropp et al. 1978, Kropp and Hershey 1979]. A common technique when applying an optimization methodology to simulation models of health care clinic is a recursive method employed by Carlson et al. [1979], Kropp et al. [1978], and Kropp and Hershey [1979]. First, an optimization technique is used to analyze and reduce the number of alternatives of the system at an aggregate level (the total system level). These results are then used in a more complex and detailed simulation model of the same system, that identifies additional information and acceptability of the results. Finally, these additional constraints are passed back into the optimization model and this process is repeated. Similarly, Butler et al. [1992b, 1992c] employ a two phase approach by first using quadratic integer programming for facility layout and capacity allocation questions, and then a simulation model to capture the complexities of alternative scheduling and bed assignment problems.

All of the above studies use a variety of optimization techniques to arrive at parameters for the simulation model. Generally, recursive simulation optimization techniques can be very difficult, and therefore, costly to implement in the health care sector. However, in recent years, a number of simulation software packages have appeared that provide an optimization

add-on to the software [Carson and Maria 1997]. Instead of an exhaustive, time-consuming, and indiscriminate search for an optimal alternative, simulation software companies are now starting to provide special search algorithms to guide a simulation model to an optimal or near-optimal solution. Examples of these include an add-on to MicroSaint 2.0 called OptQuest, that uses a scatter search technique (based on tabu search) to find the best value for one or multiple objective functions [Glover et al. 1996]. Other optimization simulation software includes ProModel's SimRunner Optimization [Benson 1997] and AutoStat for AutoMod [Carson 1996].

2.3.3 Advances in Simulation Software

In recent years, simulation software has undergone a series of technological leaps. First, the introduction of visually oriented graphical outputs greatly aids in the verification and validation of models and results [Gipps 1986, Sargent 1992], though this does not necessarily guarantee model correctness [Paul 1989]. Moreover, animation in simulation is primarily used to present (to decision-makers) the actual operation of the model and system and, in essence, to sell the insights of the system. Second, the wide use of the object-oriented paradigm (OOP) in simulation software design enables analysts to model a system without writing a single line of code [Banks 1997]. Numerous companies are developing general-purpose software packages incorporating the latest technologies [Banks 1996]. Some such packages are specifically aimed at the health care industry, such as MedModel [Carroll 1996, Keller 1996] and ARENA with a health care template [Drevna and Kasales 1994].

Jones and Hirst [1986] present one of the early articles on modeling with visual simulation, using the simulation software package named See-Why. The visualization of different policies in the visual simulation of a surgical unit and surrounding resources plays an integral part in assisting managers to identify the best solutions. Paul and Kuljis [1995] use a generic simulation package called CLINSIM to illustrate how clinic appointments and operating policies can influence patient waiting time. Evans et al. [1996] use ARENA to model an emergency department using thirteen patient categories. They reduce the patient's length of stay using alternative scheduling rules for the number of nurses, technicians, and doctors on duty during each particular hour of the simulation run. Besides these studies, several other visual simulation modeling project of interest have been conducted [McGuire 1994, Ritondo and Freedman 1993].

The number of health care organizations and government agencies using these advanced simulation packages has grown, with much of their work and the results of their efforts not available in the literature. Considering the number of easy to use simulation software packages available today, it seems unusual to find few articles describing visually oriented simulation models of health care clinics. This may be attributed to the fact that since simulation models have become easier to develop with these new simulation software packages, the type of users have changed. Although the number of operations research analysts and management engineers conducting simulation projects are unknown, numerous non-technical, and therefore non-publishing, users have emerged. It is no longer necessary to have an advanced degree in operations research or simulation to use simulation software packages because it is virtually a drag-and-drop operation. However, this development does

not diminish the importance of the contributions from operations research professionals to the health care field. They will still be required to provide expertise when conducting or managing critical or large-scale simulation projects.

2.3.4 Implementation Issues

Simulation modeling of health care clinics has been extensively used to assist decision makers as well as to improve efficiencies. For simulation to reach its greatest potential as the key tool for analyzing health care clinics, simulation results must be implemented. Unfortunately, in a survey of two hundred papers of simulation in health care, Wilson [1981] found only sixteen projects reported successful implementations. A number of recommendations were given to increase the chances of success in implementation. These recommendations include:

i) the system studied is in need of a decision,
ii) the project must be completed before a deadline,
iii) data must be available,
iv) the organizer or the decision maker must participate in the project.

Lowery [1994, 1996] addresses some additional barriers, as well as solutions to help overcome the resistance to implementation. Some of the suggestions include animating the simulation to easily communicate the problem and the solution to the decision maker, making sure management stays involved throughout the project, and avoiding too many assumptions or making the model too complex. Additionally, they suggest management engineers must

simplify the simulation process and improve their sales skills. Marsh [1979] lists three key elements necessary for the successful implementation of simulation results:

i) total commitment and support from the user,
ii) credibility of the model,
iii) the analyst must work with the real operations on hand instead of any esoteric studies.

Despite the lack of implementation observed in the literature, other benefits can still be gained from conducting a simulation study. The procedure and methodology of applying simulation requires decision makers and managers to work closely with the simulation analyst to provide details of the system, often for the first time. As a result, the manager is likely to gain a new perspective on the relationships between the available resources and the quality of health care offered by the system. Rakich et al. [1991] study the effects of simulation in management development. They conclude that simulation not only develops a manager's decision making skills, but also forces them to recognize the implications of system changes. Also, in agreement with Wilson [1981], in the cases where managers developed their own simulation models, implementation occurred very easily. Finally, Lowery [1996] realizes that there are benefits, such as identifying unexpected problems unrelated to the original problem, that arise even if implementation fails

2.4 Conclusions

This chapter presents a survey of the literature (focusing primarily on the past twenty five years) on applying discrete-event simulation to understand the operation of health care

facilities. This chapter identifies a large body of research conducted in the area of patient flow as well as resource allocation. The multiple output measures associated with health care systems makes simulation particular well-suited to tackle problems in this domain. The numerous simulation studies reported in the literature have the common theme that they each attempt to understand the relationship that may exist between various inputs into a health care system (e.g., patient scheduling and admission rules, patient routing and flow schemes, facility and staff resources) and various output measures from the system (e.g., patient throughput, patient waiting times, physician utilization, staff and facility utilization). The breadth and scope of units within hospitals and clinics makes it impossible to undertake one single comprehensive study that addresses all these issues simultaneously.

These observations, together with the void of literature in the area of complex integrated multi-facility systems, suggest the need to develop a comprehensive simulation modeling framework for determining clinical output measures and interdepartmental resource relationships. Furthermore, several observations were made on the continuing trends in simulation software, such as the development of optimization add-ons, increased visualization, and the shift to an object-oriented paradigm. These powerful features will have the greatest impact when educating decision makers on what changes need to be made and lessening resistance to implementation. The outlook for simulation in health care looks promising. The further development of more powerful high speed processing, distributed simulations [Baezner et al. 1990], and object-oriented simulation, will facilitate the creation of complex, but tractable, models of large integrated systems, with the results implemented more easily and frequently.

Chapter 3 Simulation Life-cycle: Planning

Operations research analysts, like other scientists or engineers, study and solve problems in methodical time-tested steps described as a *problem life-cycle* [Glasow et al. 1996]. A problem life-cycle can be implemented in many different ways, including the waterfall, spiral, iterative, evolutionary, fountain, and rapid application development methods [Glasow et al. 1996]. A *simulation life-cycle* is a specific type of problem life-cycle where the analyst employs simulation as the tool to study the problem [Glasow et al. 1996]. A typical simulation life-cycle from identification to implementation is depicted in Figure 3.1. This figure is the result of combining the recommendations from several authors (e.g., Banks, et al. [1996], Balci [1994], Glasow et al. [1996], and Law and Kelton [1991]) in guiding an analyst through a simulation study

There are four common phases to a simulation life-cycle: 1) planning, 2) modeling, 3) verification, validation, and testing (VV&T), and 4) experimentation and analysis. This research focuses on the modeling and simulation of the QPN using this simulation life-cycle, from planning to VV&T; this research will conduct preliminary experimentation to develop the most accurate model, but the experimentation and analysis phases will be conducted in future research.

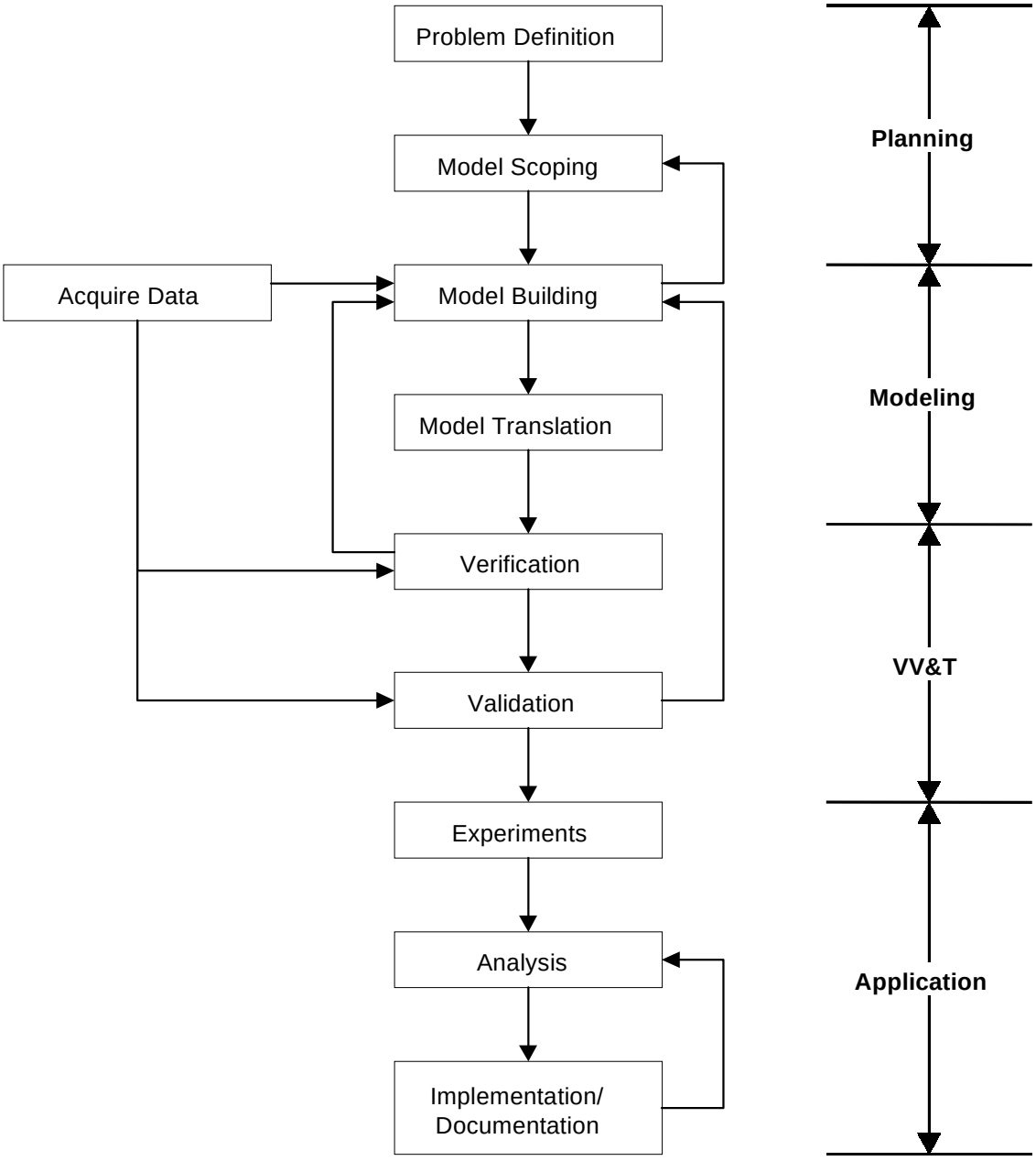


Figure 3.1: Simulation Life-cycle Diagram

This chapter presents the problem identification and formulation for the planning phase of the QPN simulation life-cycle. The planning phase includes selection of the simulation modeling tool, formation of the team that will include the analyst and the domain experts, key assumptions, scope of the problem, identifying input data parameters, and specifying output measures. The following chapters will describe the modeling phase (Chapter 4), the VV&T phases (Chapter 6), and some preliminary experimentation (Chapter 7).

3.1 Problem Identification

The QPN, a subsidiary of Biopop, attempts to centralize the administrative, financial, and telecommunication needs of a network of primary care physicians located throughout the United States. The purpose of this research is to create a discrete-event simulation model of the QPN that will assist Biopop in evaluating the operating procedures and the resource utilization of the participating clinics and the Queston Information Center. Specifically, the research will attempt to model the detailed operations of a *single* physician practice clinic, allowing future studies to design the final Queston simulation model of a multi-facility health care system. The Queston simulation model can also be used to analyze a physician-patient encounter to identify recommended staffing and scheduling schemes for a generic clinic. This research will present some preliminary results and experimentation to address these issues.

3.2 Solution Tools and Techniques

Discrete-event simulation can be used to model and evaluate the QPN. In the past 25 years, simulation has earned a proven record of success for all sizes of applications, but the strength of simulation lies in the modeling of large scale, complex systems [Banks et al. 1996, Stafford 1976]. While other analytical tools, such as queueing theory, Markov chain analysis, and linear programming can be applied to extremely complex systems, only simulation can handle the modeling details in complex systems with ease. The complexity and the level of detail in a simulation model is limited only by the size and speed of the computer hardware as well as the amount of resources allocated to the project. However, there are drawbacks to using simulation, such as the increase in cost due to the time spent and the need for additional input data to model greater system details.

The potential for the Queston simulation model to grow in complexity was realized early and simulation was selected as the technique to model the QPN. Furthermore, this potential to grow in complexity also required a particular type of discrete-event simulation software package. The simulation software package would need to possess certain features, such as object-oriented programming and visual graphics, which would allow the programmer to create a rich, complex model (module by module) and also allow future extensions to the model with minimal effort. Additionally, visual graphics would allow the user to convey the results to upper management and clients, such as marketing the QPN to potential clients of Biopop. With these criteria in mind, the software package, Visual Simulation Environment (VSE), developed by ORCA Inc, was selected as the simulation package to model the QPN.

VSE offers cutting edge technology in simulation modeling as well as a local support capability. Further details of the VSE are described in Chapter 3.2.3.

3.2.1 Modeling and Simulation

A model is a representation of a real or a proposed system. Models can be classified into two categories: mathematical and physical [Law and Kelton 1991]. A *physical model* is a physical replica of the system it is representing, such as a scaled model of a plane or a car. *Mathematical models*, such as simulation models, linear programming models, or queueing models, represent a system in terms of logical and quantitative relationships that can be manipulated to measure how the model reacts and, thus, how the real or proposed system reacts [Law and Kelton 1991].

A mathematical model can be further classified as *descriptive* or *prescriptive* [Law and Kelton 1991]. A descriptive model only describes the behavior of the system it is representing and provides solutions, but not necessarily the desired or acceptable solutions. All simulation models are descriptive. The output from a simulation model must be interpreted by the simulation analyst to judge its quality and validity. In contrast, prescriptive or analytical model provides exact analytical solutions for the system. Examples of prescriptive or analytical models are queueing models, linear programming models, and Markov chain models. Although closed form solutions would be desirable in every instance, this may not be possible if the system of interest is extraordinarily complex, as many real life

large systems can be. In this case, as in the Queston simulation model, simulation modeling is the most desirable technique to apply.

States are defined to be the collection of variables that describe a system at a particular time [Law and Kelton 1991]. Simulation models can be further categorized as either *continuous* or *discrete-event*. Continuous simulation modeling describes a system with continuously changing states, while discrete-event simulation modeling describes a system where the state variables change instantaneously at discrete intervals in time [Law and Kelton 1991]. The Queston simulation model is modeled as a discrete-event simulation model.

3.2.2 Object Oriented Programming (OOP)

Object-oriented programming (OOP) is centered around *objects* (also called *instances* or *components*). An object can be viewed as a module that contains code which receives and sends messages to other objects (see Figure 3.2) [Khoshafian and Abnous 1995]. The data contained within the objects cannot be directly modified or tampered with by other objects. The only access to this data is via a message from one object to another. This data or information hiding, also referred to as *encapsulation*, allows these modules to remain independent of each other [Khoshafian and Abnous 1995]. Consequently, during software or simulation modification, a change in one module will not effect other modules and cause either an incorrect result or a program failure.

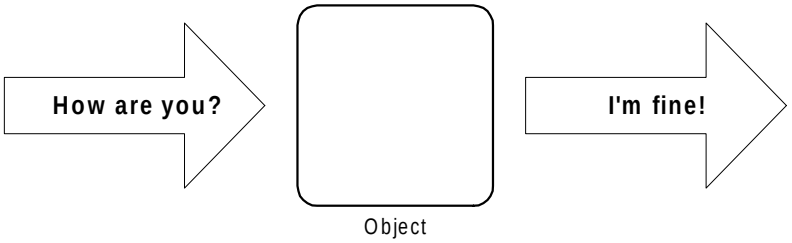


Figure 3.2: Message passing

An object is defined by its *class*, which determines what the object can and cannot do. Objects are individual instances of a class. It is possible to create numerous objects from a class. For example, objects called Patient1, Patient2, and Patient3 can all be created from a Patient class. The Patient class defines what it is to be a Patient object, and all the messages the Patient object can respond to. Everything an object can do, such as respond to a move command, reply to a query, or even to send another object a message, is represented by its message interface or *methods* [Booch 1994]. There are two types of methods: *class methods* and *instance methods*. Class methods are used to provide services specific to a class [Balci 1996]. For example, a class can provide the class method **new** which creates an instance of the class. Instance methods, given in a class, are used to specify the services provided and behavior exhibited for each object instantiated from that class [Balci 1996]. Each object or instance created as belonging to a class inherits the services and behaviors specified in each instance method of that class. The services an object can provide and the behaviors it can exhibit are specified in its instance methods.

One of the advantages of programming in the OOP paradigm is that it allows a predetermined number of objects of a class, such as Room class, Chair class, Patient class, to be instantiated at runtime. This allows parameters to be set prior to model execution. This powerful feature

permits the creation of one flexible generic clinic object that can be reused in future studies to create several identical clinics that form a network of clinics at model execution. However, constructing reusable simulation objects to create libraries of simulation modules is only one of several advantages in OOP. Ease in maintainability, the ability to modify an existing model without programming from the very beginning, and the opportunity to model the real world as close to a user's perspective as possible are other advantages of OOP.

The disadvantage of using OOP is that the program depends on message passing between objects for all data communications; the variables of an object are hidden, and hence, more lines of code will be required to access and pass the same information. With these additional lines of program, the computational speed will be slower. However, with the cost of processors decreasing, the cost of offsetting the difference in speed may be negligible.

When an object is defined as a member of a class, it inherits the characteristics and behaviors of that class as well as the class's *superclass*, and any other ancestral classes, all the way back to the root class - the class that all objects inherits from [Booch 1994]. A class that inherits from its superclass are called its *subclass* of that superclass. This feature is called *inheritance* and allows reusability of earlier developed classes [Joines and Roberts 1998]. New objects can be assigned to a subclass of the original class and inherit all the existing methods (both class and instance), and therefore, all the behavior of the original class without rewriting the code. For example, after creating the class Patient, a new class called Companion could be required, which defines some companion specific instance methods. The Companion class could be a subclass of Patient and acquire certain instance methods that Patient class already

possess such as *lookForAvailableSeat*. However, it may make more sense to define a common class called Person, that both Companion and Patient are subclasses, depending upon the behaviors required.

3.2.3 The Visual Simulation Environment (VSE)

The software package chosen to model the QPN system was the Visual Simulation Environment (VSE). The VSE, running first on the NEXTSTEP operating system and then on Microsoft NT 4.0 and OPENSTEP, is a well-suited discrete-event simulation software package for tackling the complexity of this network design. The advantages of VSE over some of the other simulation languages are its visualization of objects, domain independence, and object-oriented design and modeling. Furthermore, the close proximity of Biopop to Orca Computer, Inc. allows easier access and support from the software.

Development of the VSE started in August 1992 at Virginia Tech with funding from the U.S. Navy [Balci et al. 1998]. A fully functional VSE research prototype was created by using the NEXTSTEP object-oriented software engineering environment in 1995 at Virginia Tech [Balci et al. 1998]. After a technology transfer to Orca Computer, Inc., the first commercial version of VSE was released in November 1996 [Balci et al. 1998]. Subsequently, VSE was released for OPENSTEP Mach Unix, Windows NT 4.0, Windows 95, and Windows 98 [Balci et al. 1998]. Some of VSE's major features and advantages are:

- general purpose and domain independent

- object-oriented characteristics; facilitate the reuse of model objects
- can be used to visually simulate a system at any desired level of detail
- high level English-like scripting language for programming
- build models graphically and hierarchically
- easy to use debugging tools; traces execution errors to the source point
- controls the start, stop, and speed of the animation
- pipes output measures to files
- creates, modifies, and destroys dynamic and static model objects during model execution
- reads specifications from a constants panel and instantiates objects at run-time
- decomposes an object into a graphical hierarchy of objects
- allows dynamic objects to enter and move inside another object
- constructs confidence intervals using the VSE Output Analyzer
- provides 27 random variate generators for 27 probability distributions

A more complete description and details of VSE's capabilities and features can be found in the VSE User's Guide [Balci et al. 1996].

3.3 Problem Boundary and Resources

3.3.1 Problem Boundary

This research modeled the operations of the Question Information Center and one generic family practice clinic. The clinic includes registration/check-in booths, examination rooms,

check-in rooms, specialty rooms, a waiting room, waiting room chairs, a waiting room buffer, physician offices, a medical staff office area, and an interior waiting area. The Queston Information Center includes operators and a call queue. Dynamic objects, or objects that physically or logically move, consist of physicians, physician assistants/nurse practitioners, nurses, medical assistants, patients, companions, and phone calls.

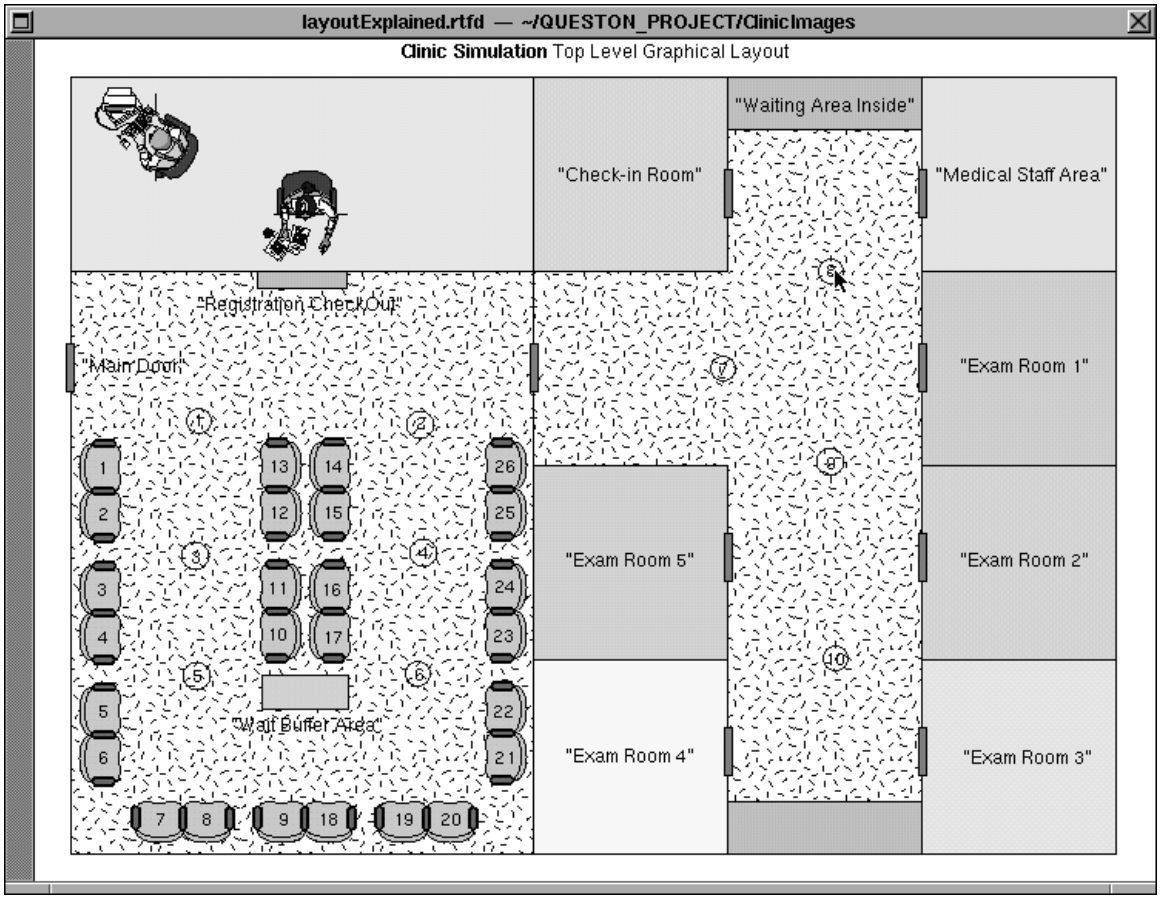


Figure 3.3: Prototype of the Queston simulation model clinic

Figure 3.3 depicts a prototype of a Queston simulation model clinic developed in the early stages of model formulation. Starting with a simple model and building towards greater complexity is a sound strategy when building simulation models [Law and Kelton 1991]. This simple prototype model helped determine some of the requirements and needs in

developing the final model, such as recognizing the need for passing parameters to instantiate the number of clinic objects (e.g., examination rooms, physicians, registration booths) at model execution. Furthermore, instead of one medical staff office area holding all the medical staff members including the physicians, separate rooms are required for the physicians to capture the output measures, since at the individual level the functionality of the physician and the other medical staff members are distinct.

At startup, the Queston simulation model is initialized with zero patients scheduled in the system (i.e., empty and idle). This (unrealistic) initial condition will bias the steady-state (long run) behavior of the output measures collected from the Queston simulation model. This phenomena, termed *initial transient*, can be reduced in a number of ways, including

1) Initialize the simulation model to be representative of the steady-state conditions [Banks et al. 1996].
2) Accept the initial transient (and the resulting initialization bias) and overwhelm the bias by running extremely long simulation runs [Yücesan 1993].
3) Divide the simulation run into two periods: a warm-up period where output measures are not being collected and a period of time when output measure are being collected [Banks et al. 1996].

The third approach was used in the Queston simulation model. The first period, called the *warm up* or the *initial transient period,* is a period of simulated time where the model truncates (deletes) all observations from the beginning of the simulation run from time zero to time T_0 (called the *truncation point*). In the Queston simulation model, T_0 was set to three months. A three month warm up period was deemed adequate to allow the simulation model to generate a sufficiently large number of appointments for patients who may schedule their

appointment weeks in advance. This assumption is validated in Chapter 4.6. Moreover, Swisher [1999] determines that the truncation point for the Queston simulation model could be as short as seven days.

Following the warm up period, daily output measures are collected for one simulated year, generating 253 discrete data points, the number of workdays in a year minus the eight holidays. Although the Queston Information Center is modeled to be open 24 hours a day, 365 days a year, the Queston clinics are modeled to be open only during the weekdays and taking appointment patients and walk-in patients between 9 AM and 4:15 AM The actual closing time each day varies depending on the work load for that day. *Walk-in patients* are patients seeking medical assistance without first checking for an appointment. Chapter 4.4.2.2 discusses the walk-in patients in further detail.

The Queston simulation model is capable of multiple replications; however, multiple replications were deemed unnecessary for the purpose of building the Queston simulation model. Since each day is virtually a beginning and an ending (opening and closing of the clinic) of a simulation run, each day was considered to be a single observation. A simulation model that exhibits this behavior is termed a *terminating simulation model* – a simulation model that runs for some duration of time until a specific event (such as the clinic closing) stops the model [Banks et al. 1996]. Even though the Queston simulation model does not terminate until 15 months have been simulated, it can be treated in this manner. This produces output measures for 253 simulated days (after warm-up

3.3.2 Key players, roles, and functions

A cross-functional team consisting of members from different units within Biopop as well as external consultants was assembled to support this research. Specifically, the team consisted of two operations research analysts, two registered nurses, and three external consultants. The nurses and the consultants will be referred to in this study as the *domain experts*. The team met periodically to map out strategy, discuss input data requirements, review progress, and move the project forward. Additionally, the domain experts provided input data as well as validation of the results of the model.

3.3.3 Preliminary Project Plan

A preliminary plan was formulated to address the expectations of the upper management of Biopop and to create a strategic and tactical plan for the Queston simulation model. Figure 3.4 depicts the timeline of the Queston project.

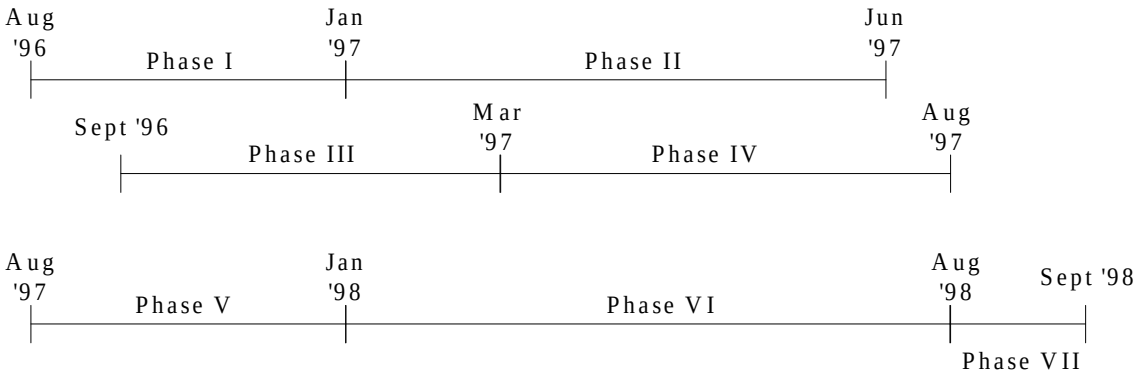


Figure 3.4: QPN Simulation Life-Cycle Timeline

The different phases identified were:

Phase I: Model Building and Testing
- Construction of model details necessary to describe the QPN
- Model Testing to assess the logic of the Queston simulation model

Phase II: Testing with Hypothetical Input Data
- Model validation and verification using hypothetical input data

Phase III: Compilation
- Input data collection

Phase IV: Input Data Fitting
- Distribution fitting
- Interactive refinement of input data collection

Phase V: Verification, Validation, and Testing (VV&T)
- VV&T of the Queston simulation model involving numerous testing and analysis procedures

Phase VI: Output and Sensitivity Analysis
- Experimental design to collect output measures from the Queston simulation model
- Sensitivity analysis of output runs with the Queston simulation model
- Extensive interaction with Biopop personnel to test and forecast utility associated with various Queston designs
- Formulation of metrics to assess the value of various Queston simulation model designs

Phase VII: Wrap-up
- Final assessment and evaluation of viable Queston simulation model designs

Chapter 4 Simulation Life-cycle: Modeling

In the second phase of the QPN simulation life-cycle, the QPN was translated and developed into the Queston simulation model. The assumptions used in the abstraction of the QPN were identified, as were the model objects and variables. The system and object flow diagrams were then developed and constructed. Since obtaining sufficient input data can be very challenging, input data requirements were identified early in the study. Lastly, output measures were specified and the simulation model programmed to collect these output measures.

4.1 System Definition and Model Formulation

4.1.1 Assumptions

Modeling a system to its most minute detail is extremely difficult and in most cases economically infeasible. Therefore, the scope and the assumptions of the Queston simulation model were first studied to determine the required objects and the level of detail necessary to capture the important features of the QPN. The inclusion of objects for the QPN, and the level of detail for each object, were determined using the following guidelines:

- Will the inclusion or exclusion of the object affect the accuracy of the output results?
- Is there accurate and available data for the object?
- Are there additional programming complexities resulting from the inclusion of the object?
- What are the additional cost and time associated with including this object?
- What is the model sacrificing if the object is not included?

For example, consider the waiting room chair object. The question is whether to instantiate these chairs at model execution or prespecify and fix the number of chairs. The conclusion to fix the number of chairs resulted in a decrease in the complexity due to instantiating chairs and movement paths (paths that patients would need to follow to get to a particular chair). The maximum number of chairs used for each physician results in the chairs being numbered consecutively and forcing patients and companions to take the lowest numbered chair available. If all the chairs are occupied, people wait in the waiting room buffer, which also records a maximum number. Note that chairs are unnecessary to capture the required output measure; however, the opportunity of observing the animation of patients and companions occupying the chairs would be lost without these objects.

All the assumptions during this early stage, as well as a description of the motivation for these assumptions, are contained in Table 4.1. Assumptions made later in the development of the model are discussed in later sections.

Table 4.1: Assumptions and Motivation

Assumptions	Motivation
The initial model includes one physician clinic modeled around a family practice.	The objective of this research is to model the operations of a single clinic and then instantiate additional clinics to form a network in future studies. Moreover, the input data and the output measures for a family practice clinic is better estimated and more available for collection than other types of practices.
Patients are scheduled between the hours of 9:00AM and 4:30PM, excluding 11:30AM – 1:00PM.	These are the typical hours of most family practice clinics. The hours excluded are typical lunch break hours.
Two appointments with a physician are scheduled for every 15 minute time slot, with the exception that only one appointment is scheduled every fourth time slot. One additional clinic appointment can be scheduled in every 15-minute slot.	According to the domain experts, each physician is found to schedule a maximum of 43 patients per day plus any additional clinic appointments. *Clinic appointments* are scheduled appointments that a medical staff member can handle but will not require the assistance of a physician.
Walk-in patients are included.	To study the effects of walk-in patients on the operations of the clinic as well as to determine the quality of service provided to the walk-in patients.
Companions are included.	Companions utilize chairs that will be needed to analyze the facility size.
During lunch and after clinic hours, all patients still in the clinic will be processed and treated.	By processing these patients, output measures, such as the length of physician lunch break and the overtime of the clinic, will be collected.
Physicians will be allowed to take their lunch between 11:30 and 1:00.	The physician lunch will start at 11:30, as long as the physician is not required to service any patients still in the clinic.
No scheduled breaks (including lunch) for non-medical and medical staff members.	Depending on the clinic, this could be a realistic policy.
A patient's actual arrival time for their appointment is distributed normally around their scheduled appointment time.	The normal distribution is the best fit from the small amount of data collected at a Christiansburg, Virginia clinic.
No preemptive servicing.	Although excluding this would be unrealistic, the number of occurrences in a family practice clinic is small and are represented by walk-in patients.
No staff or physician absenteeism; staff reports to work on-time every.	Modeling this detail would exceed the model complexity desired. In addition, it would be difficult to collect data on this statistic.

4.2 Define Model Objects

The abstraction of a real system into a logical representation of a simulation model requires a modeler to determine which elements of the real system are relevant to the objective of the simulation study. In the Queston simulation model, some of these determinations were provided during the assumption making stage; however, to complete the conceptual model, the exact details of the model, such as the patient flow, must be identified, specified, and planned. By consulting with the domain experts, all the objects were identified, the model hierarchy outlined, the clinic layout detailed, and the flow of patients, phone calls, and medical staff member determined. The objects deemed necessary for the Queston simulation model consisted of:

- One clinic object
- One Queston Information Center object
- Queston Information Center's phone operator
- Queston Information Center's phone call queue
- Clinic facility consisting of examination rooms, check-in rooms, specialty rooms, medical staff office area, physician office, registration booth, interior waiting area, waiting room buffer, and waiting room chairs
- Clinic medical staff members comprised of technicians, medical assistants, nurse, physician assistant/nurse practitioner, and the physician
- Clinic receptionist
- Patients
- Companions
- Phone calls

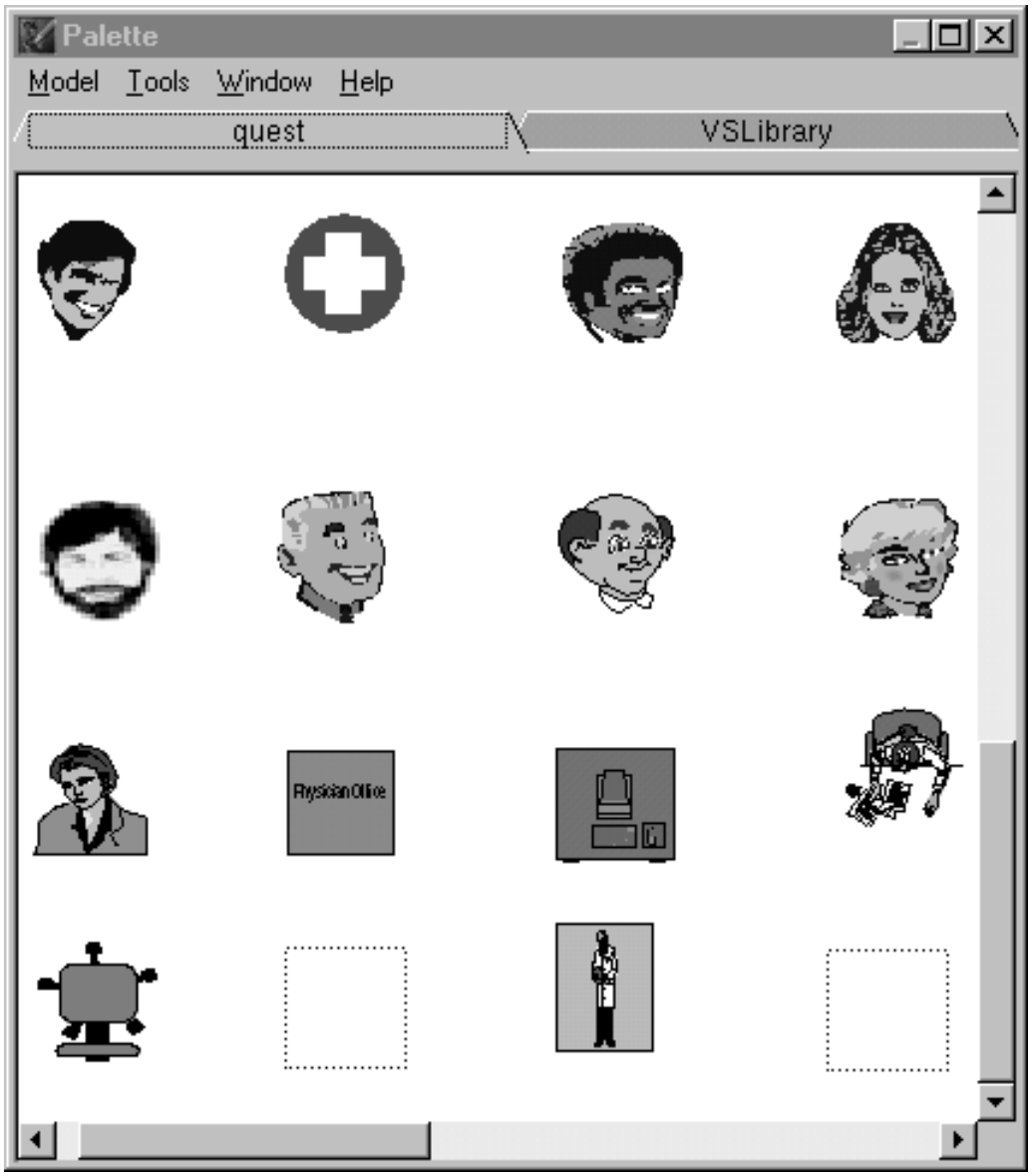


Figure 4.1: VSE Palette Window

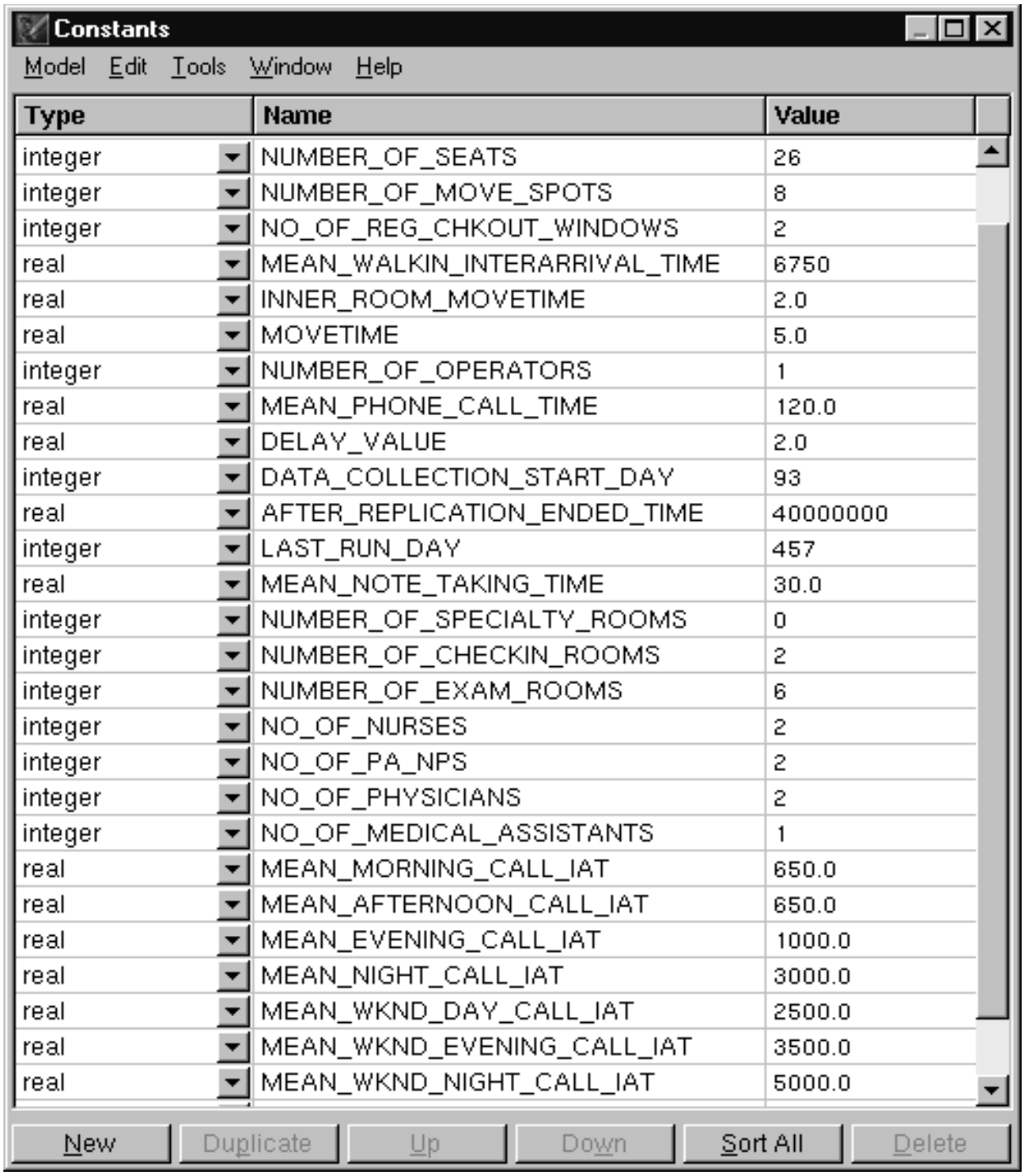


Figure 4.2: VSE Constants Panel

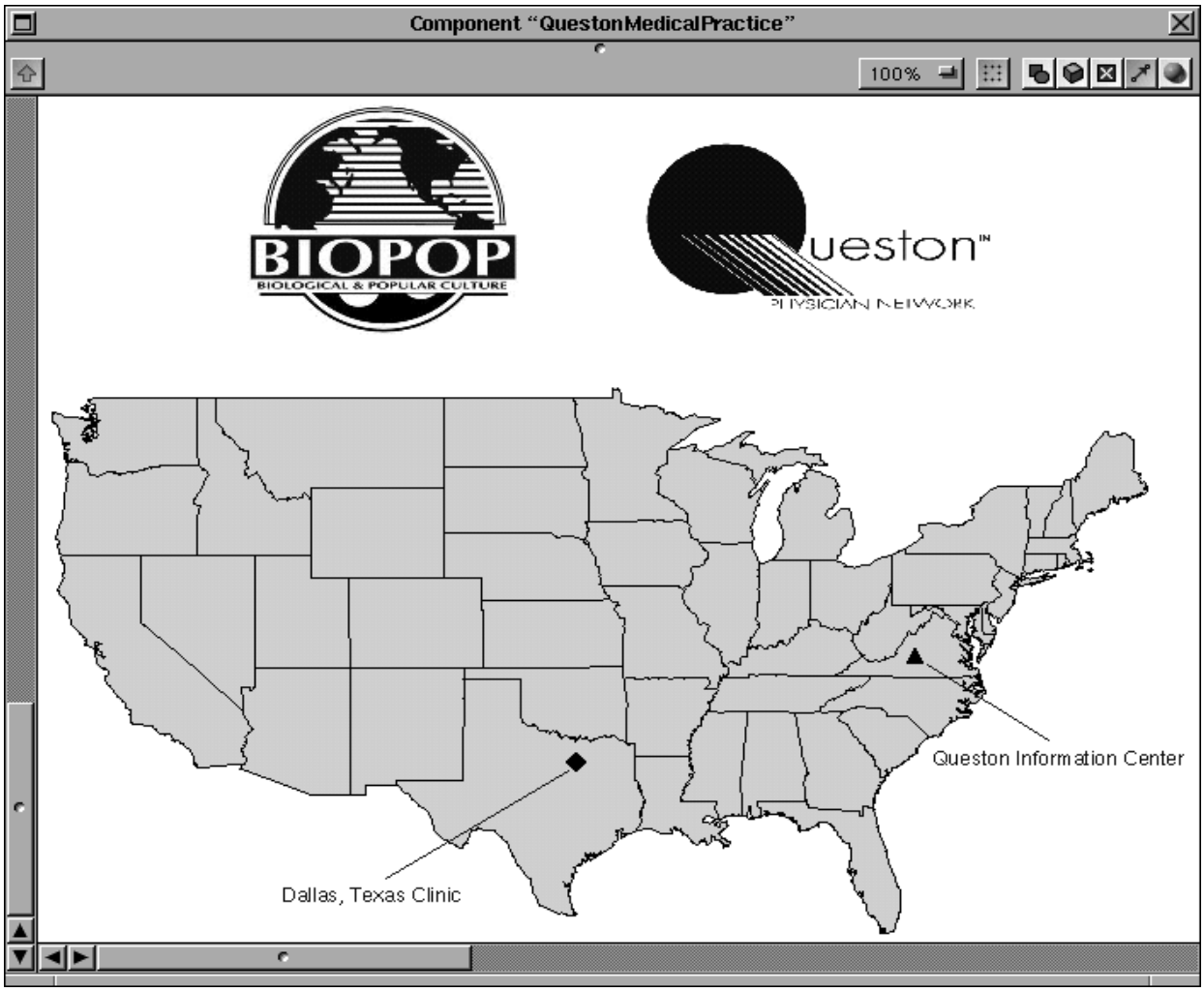


Figure 4.3: QuestonMedicalPractice Object

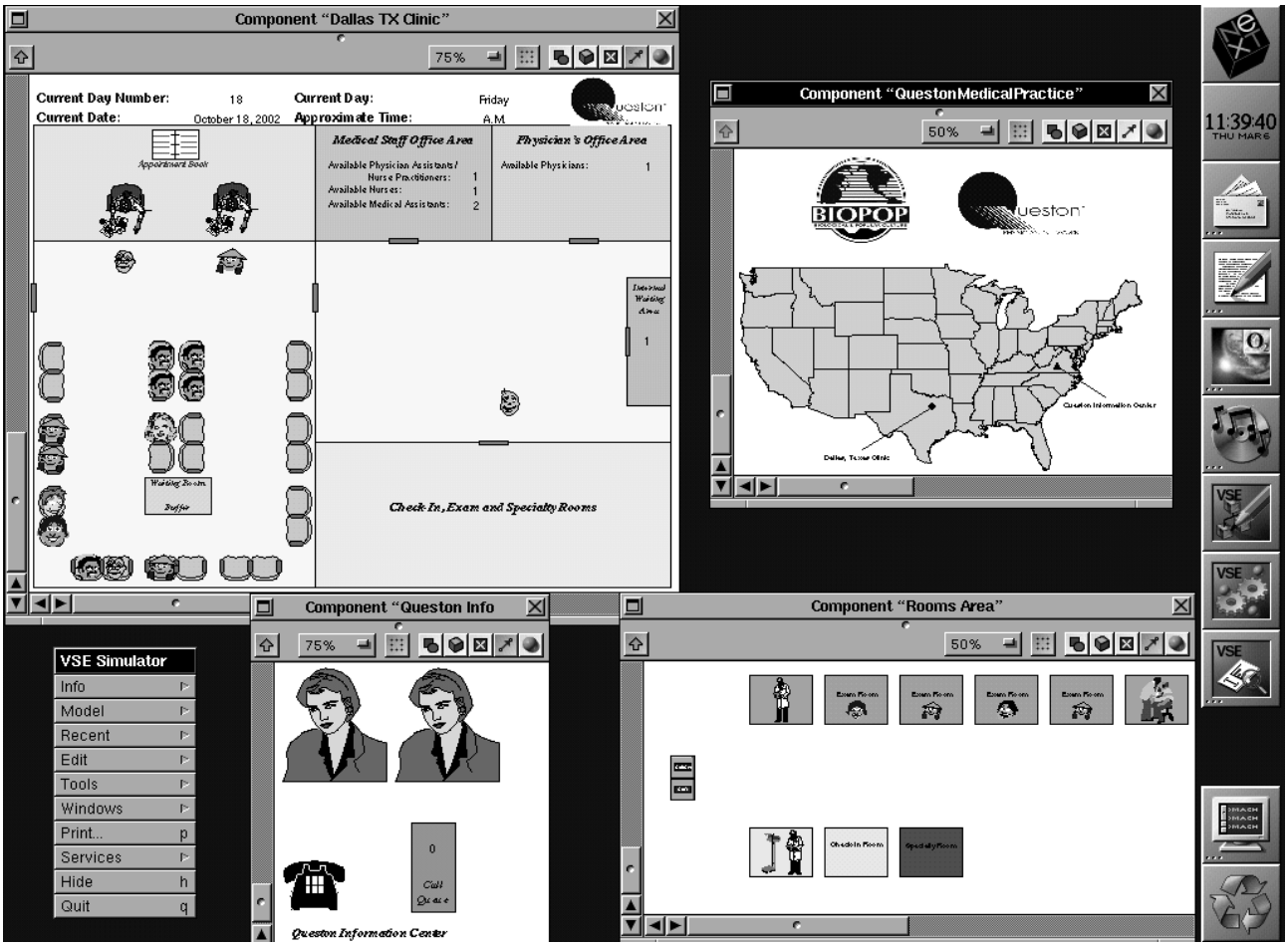


Figure 4.4: Multiple Views of the Queston Simulation Model

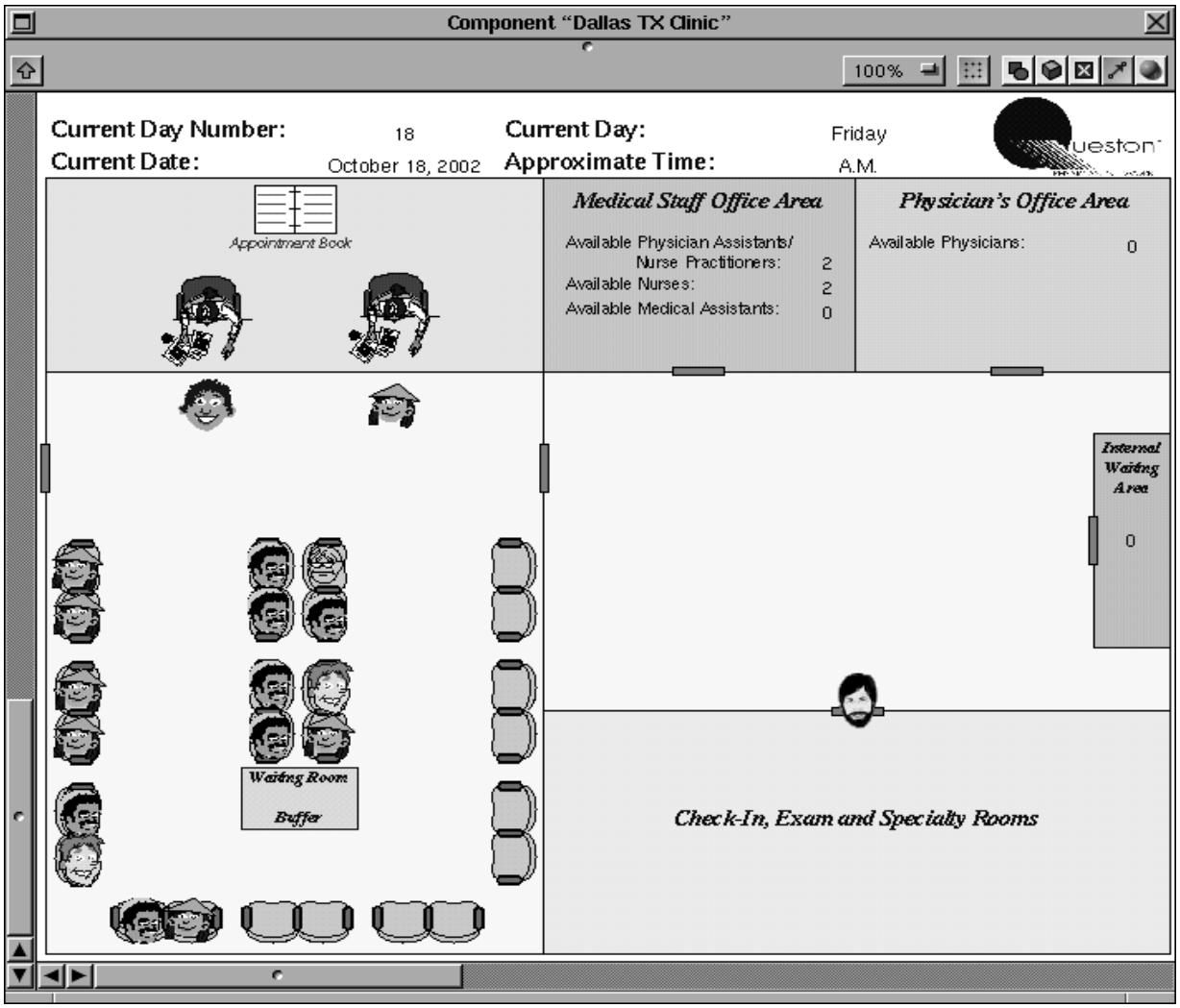


Figure 4.5: Queston Simulation Model Clinic Object

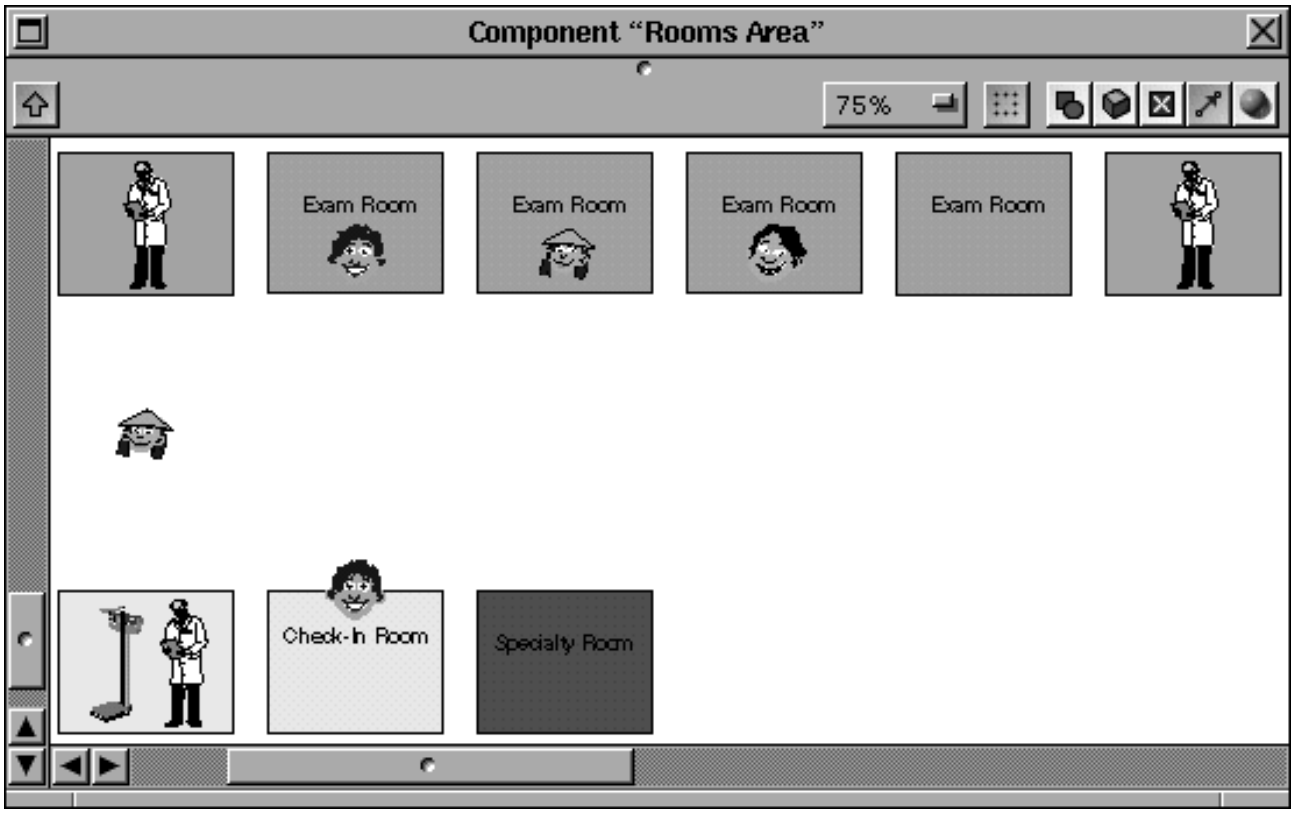


Figure 4.6: RoomsArea Object

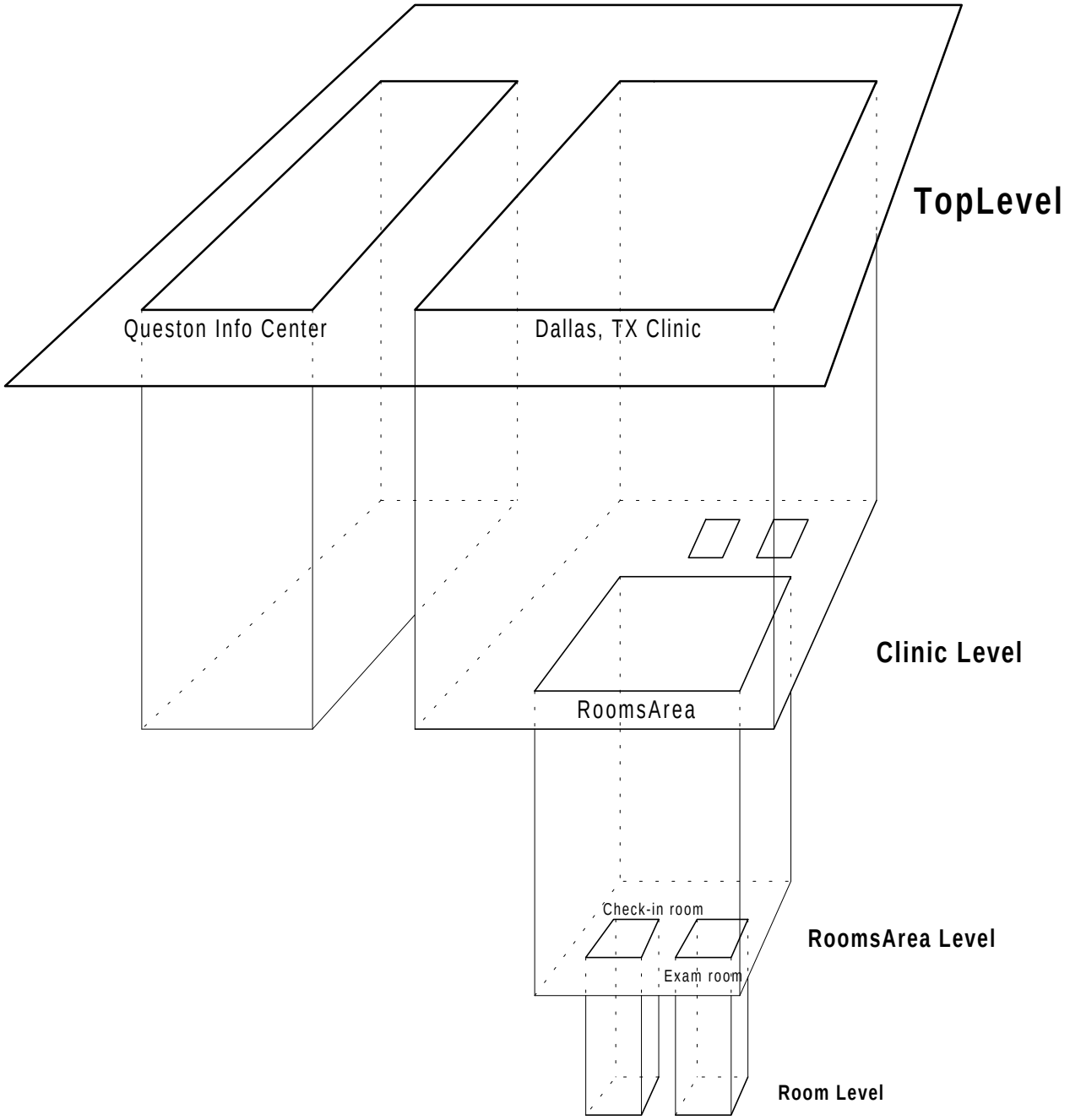


Figure 4.7: Hierarchy of the Deep Static Object in the Queston Simulation Model

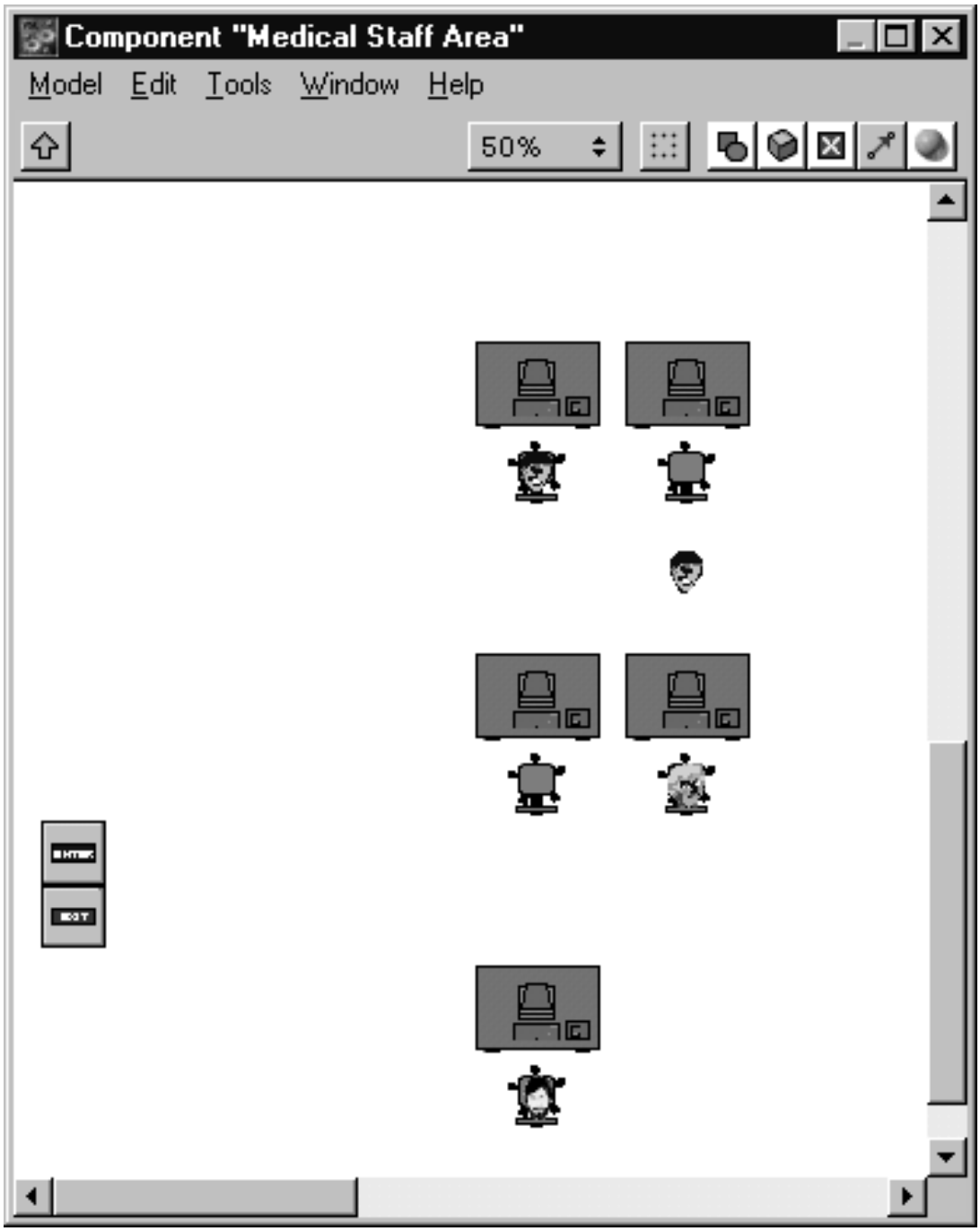


Figure 4.8: Medical Staff Office Area

Note that the nurse practitioner and the physician assistant were combined into a single job function. The domain experts felt that most physician practices employed either one or the other, but not both. Since the job functionality of the two are closely related, the two job functions were combined into one for model simplicity, hence, the simulation model used physician assistants instead of nurse practitioners.

To provide maximum flexibility in the Queston simulation model and subsequent analysis, a user specifies the number of each of the above objects. Therefore, the medical staff members, examination rooms, check-in rooms, specialty rooms, receptionist booth, and the operator, can all be instantiated prior to model execution. This is accomplished by first creating and defining an object, and then placing this object in the VSE palette. The purpose of the VSE palette is to allow users to use objects created earlier at design time or model execution time. It contains an exact copy, physically and logically, of all of the objects that can be reused. Figure 4.1 depicts the palette window containing a number of reusable model objects for the Queston simulation model. Once an object is in the palette, the modeler only has to specify the number of each object to instantiate in the VSE constants panel (Figure 4.2). The constants panel is an extremely important tool in creating a scaleable Queston clinic, as well as for designing future experiments and performing output analysis. The key parameters specified in the constants panel are:

- Number of examination rooms, check-in rooms, and specialty rooms
- Number of medical assistants, nurses, nurse practitioners/physician assistants, and physicians
- Number of registration windows/booths

- Call arrival rates during the morning, afternoon, evening, and night time periods for both weekdays and weekends
- Number of operators
- Mean length of a phone call
- Mean note-taking time
- Number of seats
- Number of days simulated
- Length of warm up period

4.2.1 Model hierarchy

The VSE allows the modeler to develop and create models in a hierarchical manner. The VSE model architecture consists of *deep static* and *deep dynamic* objects, objects that can be further decomposed into as many levels as required by the modeler, and *shallow static* and *shallow dynamic* objects, objects with no decompositions [Balci et al. 1996]. Dynamic objects are either objects created at model execution, such as physicians, or objects created and destroyed during model execution, such as patients and phone calls. This type of object can be manipulated by physically or logically moving from one object in the model to another object in the model. Like deep static objects, deep dynamic objects are objects that can be hierarchically decomposed into multiple layers. Note that deep dynamic objects were not used in the Queston simulation model.

In the Queston simulation model, the *top level* (or highest) object (Figure 4.3) consists of a deep static object depicted as a top level map of the continental United States (called the "QuestonMedicalPractice" object), an object representing a Queston clinic (called the "Dallas, TX Clinic" object), and an object representing the Queston Information Center

(called the "Queston Info Center" object). The *Queston clinic* object and the *Queston Information Center* object are both deep static objects that can be further decomposed to reveal additional layers. Figure 4.4 depicts the top level, the clinic, and the Queston Information Center objects. VSE allows the user to open as many viewers (or windows) at one time as needed, without regards to the hierarchical level. By clicking on the Queston Information Center object in the top level object, the VSE decomposes the Queston Information Center object down to its next level, showing the objects that define the Queston Information Center, including the operator object, a call queue object, and a call entrance object (Figure 4.4). Similarly, the Queston clinic object can be decomposed to reveal a clinic with several additional objects (Figure 4.5). Clicking on any one of the deep objects, such as the *room area* object (called the "Rooms Area" object), reveals a third level in the hierarchy, with a view of the medical area that consists of examination rooms, check-in rooms, and a specialty room (Figure 4.6). The structure of the hierarchy is depicted in Figure 4.7.

4.2.2 Clinic layout

Figure 4.5 depicts the layout of the Queston clinic. Shallow objects (i.e., objects that do not have additional layers) visible at the Queston clinic level are the registration windows and the waiting room chairs. The object representing the *registration window* (called the "RegistrationWindow" object) is both an area where the patient registers with a non-medical staff upon entering the clinic to fill out forms and an area where the patient checks out by paying the bill, rescheduling the next appointment, etc., before exiting the clinic. Upon entering the clinic, the patient moves to an available registration window and is serviced. If

the registration windows are all busy, the patient waits on one of the waiting room chairs, if available, and is assisted on a first-come-first-serve basis. The *waiting room chair* objects, labeled “Seat1”, “Seat2”, “Seat3”, etc., can accommodate one patient each. The number of such objects are fixed at twenty-six. If all the chairs are occupied, then the patient is sent to the waiting room buffer until a chair becomes available.

Deep objects of the Queston simulation model at the clinic level are the entrance object (called the “Main Door” object), waiting room buffer object (called the “Waiting Area Buffer” object), internal waiting area object (called the “Internal Waiting Area” object), medical staff office area object (called the “Medical Staff Area” object), physician office area object (called the “Physician Offices” object), and the rooms area object that contains the check-in, examination, and specialty rooms. Patients enter and depart the clinic through the *entrance* object; this is also where patient objects are destroyed in the model. If the patient or companion is required to wait and no chairs are available, then the patient or companion enters the waiting room buffer. The *waiting room buffer* object is an area where patients are collected and counted if there are no available waiting room chairs available. A counter on top of the waiting room buffer object displays the number of patients currently within the buffer. This buffer serves to capture the number of patients that would be standing due to the lack of waiting room chair resources. Similar to the waiting room buffer, the *internal waiting area* object is an area that patients enter if there are no available examination rooms or specialty rooms. In reality, the internal waiting area may represent a room, hallway, or a designated area that patients wait until an examination room becomes available. Like the

waiting room buffer, a counter displays the number of patients currently residing inside the internal waiting area object.

The *medical staff office area* object (Figure 4.8) and the *physician office area* object provide medical staff members and the physicians, respectively, an area where they may wait while being idle or taking notes and recording information about a patient. The number of desks and chairs in the medical staff office area, as well as the number of physician offices in the physician office area can be specified prior to model execution on the constants panel.

The *rooms area* object can be decomposed to show examination room objects, check-in room objects, and specialty room objects. A *check-in room* object is a room or an area designated for the purpose of gathering preliminary medical information on a patient, such as weight, blood pressure, and temperature. In some clinics, this room may be represented only by a chair and a weighing instrument in a hallway. An *examination room* object is where the examination, pre-examination, post-examination, and the exit interview are conducted (these will be defined later in this chapter). A *specialty room* object is a room or an area where special equipment, such as a x-ray machine or a blood centrifuge is placed. For each specialty room, a technician is created to serve patients using the room for pre-examination or post-examination procedures. Unlike other medical staff members, once a patient is serviced, the technician remains in the specialty room, and becomes idle until the next patient arrives.

Figure 4.9 depicts a hierarchical diagram of deep static objects found in the Queston simulation model. Note that each room is also a deep object, hence can be further

decomposed; however, decomposing the rooms provides no additional information, since the only objects at this level are the entrance and exit objects.

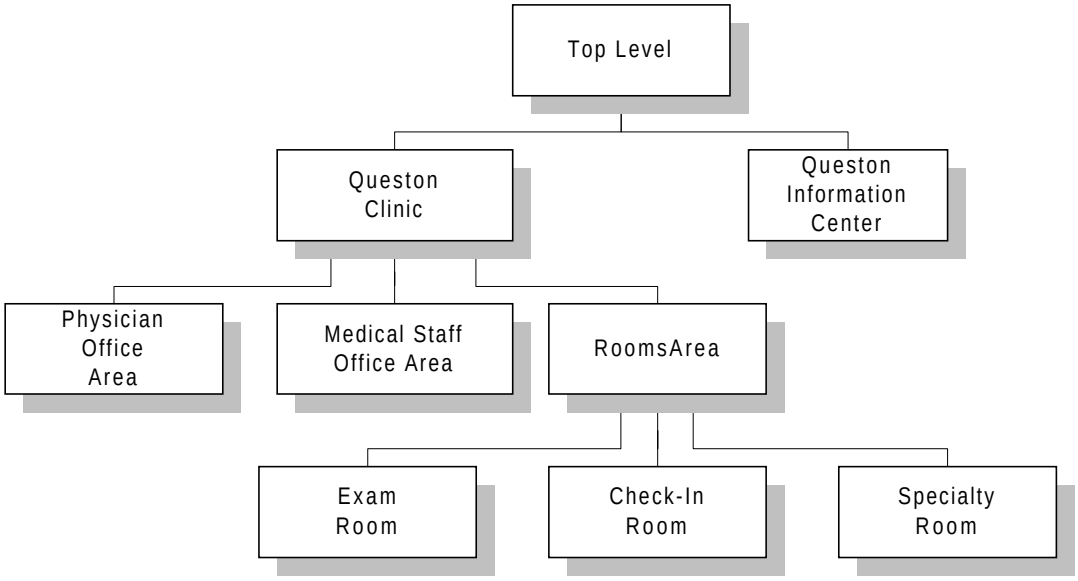


Figure 4.9: Deep Object Hierarchy

4.3 Dynamic Object Flow

Dynamic objects in the Queston simulation model can be classified into four principle types: phone calls, patients, medical staff members, and companions. These objects are programmed to move physically within the Queston simulation model, some through multiple levels within deep static objects. For example, medical staff members may start from the medical staff office area, move up to the clinic object, down into the rooms area object, and further down into an examination room object to assist a patient, as depicted in Figure 4.9. Dynamic objects interact with static objects, such as when a patient moves to a registration window, as well as with other dynamic objects, such as when a medical staff

member provides treatment to a patient. The flow and logic of the four types of dynamic objects, phone calls, patients, medical staff members, and companions are explained below.

4.3.1 Phone Call Dynamic Object Flow

A phone call object is created in the Dallas clinic area and routed to the Queston Information Center, entering the Queston Information Center through the icon of a phone that represents the telephone switching equipment. The phone call is then either routed to an available operator or is queued in the call queue object until an operator becomes available; the call queue operates on a first-come-first-serve policy. When the call is (eventually) routed to the operator, the picture of the operator icon changes from a face, representing an idle operator, to a face holding a telephone, representing a busy operator. Once the call has been completed, the phone call is destroyed.

4.3.2 Patient Dynamic Object Flow

A patient is created seventy-five percent of the time a call is routed to the Queston Information Center, since not all calls received by the Queston Information Center are requests to schedule an appointment. For this reason, the domain experts agreed that twenty-five percent of all calls should not result in an appointment. These calls may represent calls asking for directions, questions for a medication, confirming an appointment, etc. When a call results in the scheduling of a patient, a patient still may not appear at the clinic; this is due to the no-show rate of each patient. No show patients are created at the time of its

appointment and occupies either a physician time slot (for appointments that the physician is required to assist the patient) or a clinic time slot (for scheduled appointments that the medical staff member can handle and does not require the assistance of a physician), but is destroyed before entering the clinic.

Other than no shows, once a patient is created, it enters the clinic through the entrance and proceeds to a registration window. Figure 4.10 depicts the flow of patients through their entire lifecycle in the Queston simulation model. Within the clinic, the patient may undergo up to seven different medical and clerical procedures: registration, check-in, pre-examination, examination, post-examination, exit interview, and check-out. Furthermore, the procedures the patient will undergo as well as the type of medical staff member required to assist the patient in each of these procedures are determined when the patient is created.

All patients entering the clinic visit registration. This is the process of providing clerical information required for the visit. In the Queston simulation model, this time is spent by the patients interacting with the registration clerk. Once registration is complete, the patient is required to check-in. This requires the patient to physically move from the registration window to one of the available check-in rooms in rooms area. If all the check-in rooms are in use, the patient moves to either an unoccupied waiting room chair or to the waiting room buffer (if no chair is available). Patients waiting for the registration window, the check-in room, or a waiting room chair queue are served using a first-come-first-serve service rule; in the Queston simulation model, all queues are serviced in this manner.

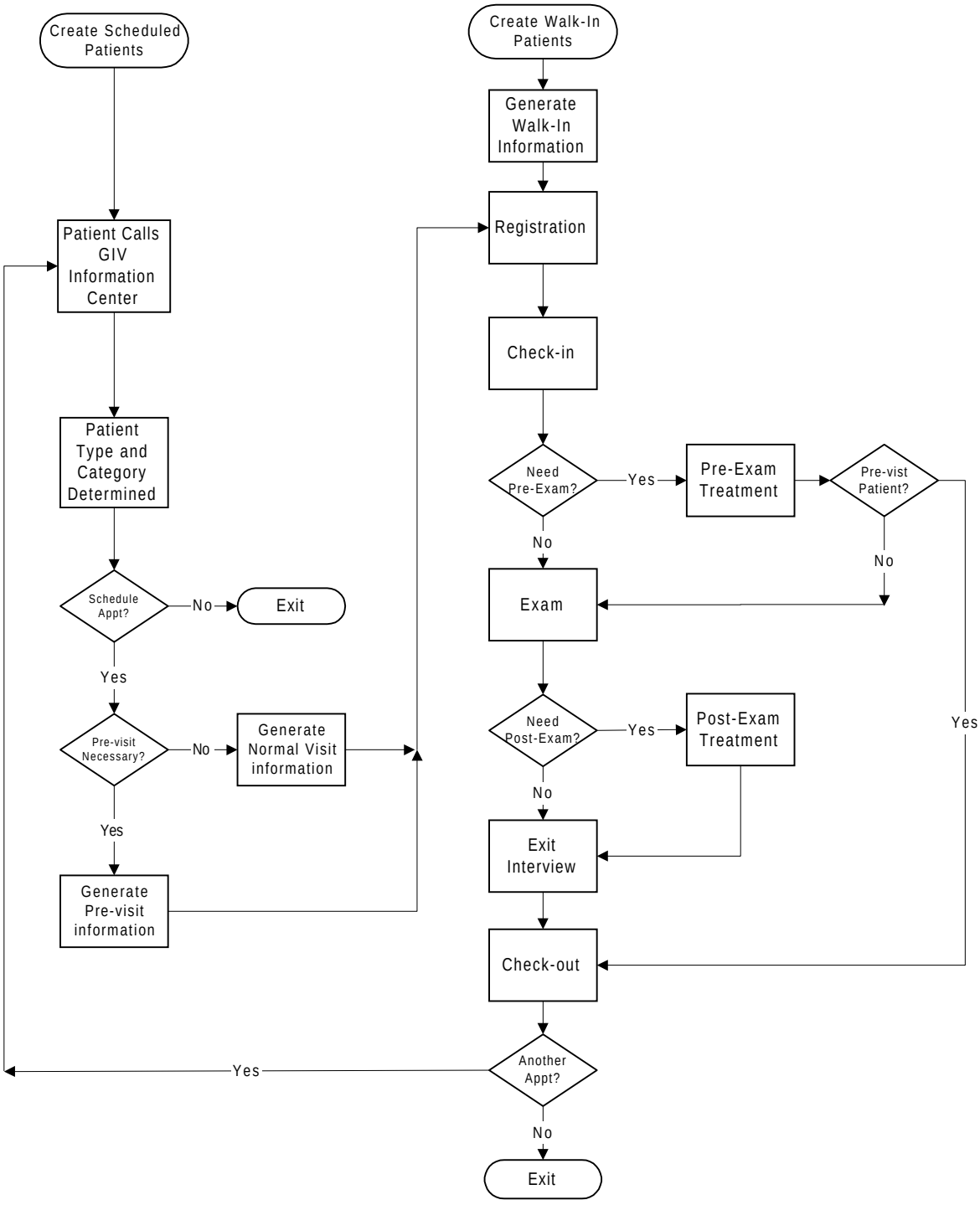


Figure 4.10: Patient Flow

In the check-in room, patients wait until serviced by a predetermined medical staff member, or by a medical staff member with higher qualifications, if the former is busy. For example, if the patient requires the service of a medical assistant, and all medical assistants are unavailable (e.g., attending to other patients or taking notes), then the next highest qualified type of medical staff member, which in this model is a nurse, assists the patient. This staffing hierarchy allows patients to be serviced more promptly during their visit. Physician Assistants are at the top of the hierarchy and can deliver all of the services medical assistants and nurses can provide. Next in the hierarchy are nurses and then finally medical assistants. Physicians are excluded from this hierarchy since they rarely provide the types of services that medical assistants would provide. During the *check-in* procedure, the medical staff member collects preliminary vital medical information from the patient, such as weight measurement, blood pressure, and temperature.

After completing the check-in process, if a patient requires a pre-examination, then the patient moves to either an available specialty room or an available examination room. If none is available, the patient proceeds to the internal waiting area. If the patient requires an examination, the patient moves either to an examination room or to the internal waiting area (if all the examination rooms are occupied). The *pre-examination* is when a medical staff member collects additional medical information, such as x-rays and blood tests, prior to the examination. This is done in an examination room or a specialty room with the patient. The *examination* is when a medical staff member performs a standard examination in an examination room on a patient. Upon completion of the examination, the patient may or may not be required to undergo a post-examination. The *post-examination* is when a medical staff

member collects additional information from a patient in the specialty room or the examination room *after* the examination. An *exit interview* (after the post-examination) is when a medical staff member conducts a final consultation, issues a diagnosis, prescribes medicine, and dismisses the patient from an examination room

Note, that if a patient finishes one procedure (e.g., pre-examination, examination) and requires another procedure, then the patient exits the examination room from where they have just been treated and moves to another available examination room, if and only if there are no other patients waiting for the room in the internal waiting area. Otherwise, the patient moves to the internal waiting area and are served using a first-come-first-serve service rule. This allows for a better utilization of examination room space.

After receiving medical treatment, a patient moves back to the waiting room/receptionist area to check-out at the registration window. Similar to the initial registration visit, the patient either waits for a free registration clerk in the waiting room area or moves directly to an available registration clerk. The *check-out* is the process where the registration clerk conducts clerical duties such as recording information, billing the patient, and scheduling follow-up appointments.

Although the patient flowchart (Figure 4.10) depicts the flow of all patients through a minimum number of procedures, including registration, check-in, and check-out, during patient data collection, several categories of patients that did not meet this flow were identified. This small group of patients registered, conducted one type of examination, and

then checked out. Therefore, the flow in Figure 4.10 depicts the typical route a patient will take and not a rigid requirement for all patients. The probabilities associated with each patient is determined by which category the patient is classified into. The categorization of patients are covered in detail in Chapter 4.3.1.

4.3.3 Medical Staff Member Dynamic Object Flow

Medical staff member dynamic objects are created at model execution in the medical staff office area for the medical assistants, nurses, and the NP/PAs, and in the physician office for the physicians. These medical staff members are categorized into the following two functional groups: 1) physicians, and 2) the rest of the medical staff member, composed of medical assistants, nurses, and NP/PAs. Physicians wait in the physician office until a patient enters a room in the rooms area and requests the assistance of a physician. The physician proceeds to the room where the patient is waiting for assistance, provides the necessary service and/or treatment to the patient, and returns to their office. Upon returning, the physician conducts note-taking activities for a period of time determined randomly using an exponential distribution with a mean of *MEAN_NOTE_TAKING_TIME*. The *MEAN_NOTE_TAKING_TIME* is set in the constants panel. Upon completion, checks the examination rooms in the rooms area to determine if another patient requires attention.

Medical staff members assist patients in the same manner as the physicians except for one major difference. The medical staff members, upon returning back to their desk in the

medical staff office area, conducts note-taking activities for a period of time equal to twice the length of the service provided to the patient they most recently treated.

4.3.4 Companion Dynamic Object Flow

Companions are essential objects of the Queston simulation model. Although companions do not interact with any medical staff members or clinic personnel, they do utilize the waiting room chairs. Therefore, to determine the number of chairs required in the clinic, companions were included in the Queston simulation model. In the Queston simulation model, companions accompany the patient into the clinic. All companions in the simulation model are depicted by a unique icon of a face. Upon entering, the companion immediately seeks out a chair or stands in the waiting room buffer, whereas the patient proceeds directly to the registration clerk (if one is available). Companions wait in the waiting room until the patient completes their office visit, at which point, the companions exit the clinic through the entrance object with the patient. These objects, like the patients, are destroyed in the entrance object.

Each time patients enter the entrance object, zero to three companions are created in the entrance object with a certain probability given in Table 4.2. These probabilities were initially determined by the domain experts and validated by the data collection at a Christiansburg, Virginia clinic. The sample statistics gathered at this clinic showed the mean number of companions per patient to be 0.45, which is close to the expected number of

companions generated in the Queston simulation model (mean of 0.41 companions per patient).

Table 4.2: Patient's Probability of Having Companions

Number of Companions	Probability of Occurrence
0	0.70
1	0.20
2	0.09
3	0.01
Total	1.00

4.4 Input Data Requirements

It was necessary to identify input data as early as possible in the QPN simulation life-cycle, since the input data collection and the input data fitting phase required a significant effort. The domain experts provided hypothetical data and their best guesses in order to expedite the development of the Queston simulation model. Forms were created to guide the domain experts to the exact nature of the required input data. These forms will be described in the Chapter 4.4.2.1.

The input data provided by the domain experts was validated by comparing the results of the model to the data collected in a time study at a Christiansburg, Virginia clinic. Although the resources needed to collect an extensive and complete volume of data at the clinic was unavailable, the data collected gave the domain experts the insights to further refine the data provided earlier in the simulation life-cycle.

4.4.1 Queston Information Center

The Queston Information Center is located in Radford, Virginia, and will be the center of all inbound calls to all clinics in the QPN. All calls arriving into the Queston Information Center will be for the Dallas Clinic, since only one clinic is being modeled (in Dallas, Texas) for this project. The Queston Information Center object in the Queston simulation model consists of a telephone switch that handles all inbound calls, a user defined number of operators, and a call queue where the calls may queue up before being handled by an operator.

4.4.1.1 Operators

When a call arrives into the Queston Information Center, the call is initially held in the call queue and an available operator is sought. If no operators are available, the call remains in the queue until an operator becomes available. Once an operator is available, the call is routed to this operator, and the operator is shown to be busy by displaying an icon of a person talking on the phone. The Queston simulation model will eventually contain more than one clinic and the operator will determine the clinic associated with each call.

Some of the calls answered by an operator will be questions concerning billing, medication, or already scheduled appointment times. The remaining calls are from patients wishing to schedule an appointment at the clinic. The domain experts suggested that 75% of all incoming calls are scheduling calls. Upon receiving the call, the operator opens the

appointment book and searches for an available time slot based on the patient's desired date, time, and clinic location. After booking the appointment (scheduling a patient to be generated at a specific time at a clinic), the operator object destroys the call, and becomes available for another call (depicted graphically by an idle operator).

4.4.1.2 Appointment Book

In the Queston simulation model, the appointment book is programmed into the clinic object. This was done strictly to increase the ease in programming this appointment book object into the simulation model. However, in the QPN, the electronic appointment book will reside in the Queston Information Center, but be accessible by the clinic. The appointment covers fifteen months, from October 1, 2002 through December 31, 2003. The clinic will be open only during weekdays; weekends and holidays are blocked off as unavailable in the appointment book. The holidays blocked off include two days each for Thanksgiving day 2002 and 2003, two days each for Christmas 2002 and 2003, one day each for New Years 2003, Memorial day 2003, Independence day 2003, and Labor day 2003.

Figure 4.11 schematically displays the breakout of the appointment book by day, time slots, and physician appointments. In the Queston simulation model, the appointment book is a series of nested object lists. Three object lists are used to create the appointment book: day object list, time slot object list, and physician appointment object list. These object lists are created from the VSE's VSList class. The VSList class is a preprogrammed class that defines the logic of a group of similar objects. These objects are ordered from one to a user specified number of objects.

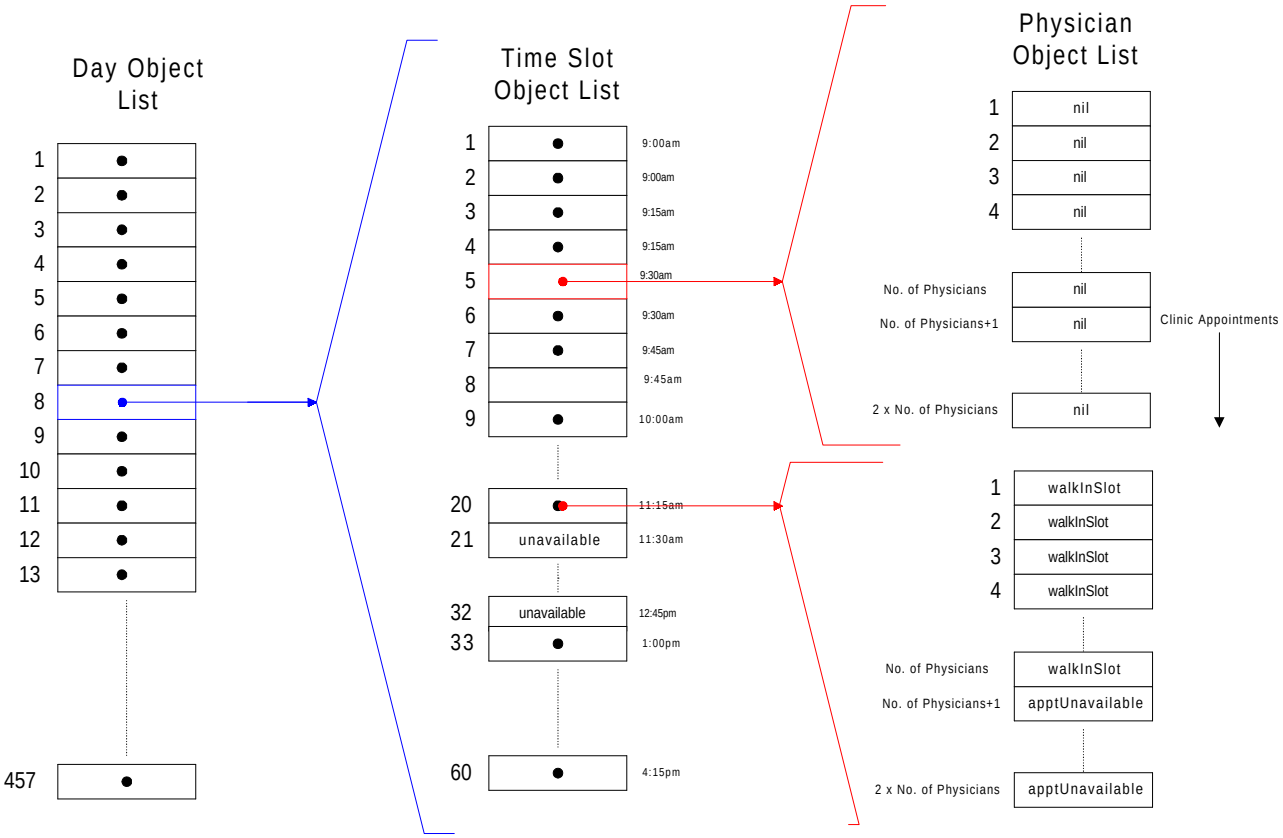


Figure 4.11: Time slot Number and Appointment Times

The day list is an object list that contains information on each day for a user defined number of days (the constant *LAST_RUN_DAY* in the constants panel). For the Queston simulation model, *LAST_RUN_DAY* was set to 457, since the model used a three month warm-up period and a one year simulation run. Each day object contains 60 time slot object lists-two time slot object lists for every fifteen minutes between 9:00 AM to 4:15 PM Each time slot object contains a physician appointment object list with the number of objects in each physician appointment object list equaling twice the number of physicians in the clinic. At the beginning of the simulation model execution, all physician appointment list objects are initially set to *nil* which indicates that the appointment object is available for an appointment. For example, in a four physician clinic, each physician appointment object list consists of eight objects. However, half of the objects in each physician appointment object list (the last one half of the physician appointment objects) are reserved solely for clinic appointments. These appointment objects are available for use by patients requiring a clinic visit but does not require the services of a physician. An example of a clinic appointment patient are patients who need to regularly visit the clinic to monitor their blood pressure. Since these patients do not require the services of a physician, these patients are restricted to a physician appointment object reserved for clinic appointments; they cannot request an appointment in a physician time slot. Starting with the second time slot object, every other physician appointment object list has all the objects reserved for clinic appointments defined as *apptUnavailable* and are made unavailable for appointments. This results in each physician having two available physician time slot and one available clinic time slot every fifteen minutes during the office hour except during the lunch break. After excluding all 15 minute time slots during the lunch break (11:30 a.m to 12:45 PM), 43 time slots are available for

patient appointments. The remaining 17 time slots are defined with the string *apptUnavailable*. Furthermore, every eighth time slot object list (one time slot object list per hour) has all physician appointment objects initialized with the string *walkInSlot* to allow walk-in patients to enter the clinic.

Each physician may have seven physician appointments, three clinic appointments, and one walk-in patient per hour. However, more walk-in patients may be treated by a physician if these additional walk-in patients enter the clinic in place of a physician appointment or a clinic appointment. The maximum number of physician appointment patients, clinic appointment patients, and walk-in patients one physician may service in one day are forty three, twenty four, and five, respectively, for a daily maximum of seventy two patients. This maximum was never reached during any of the simulation runs. However, a two physician clinic serviced 124 patients (of a maximum of 144 patients) on Friday, December 26, 2003 (the first working day after the Christmas holiday). The clinic experienced a large number of patients on this particular day because of two scheduling factors. First, the scheduling algorithm searching for appointments on December 24, 2003 and December 25, 2003 would most likely look for appointments on the following day (December 26, 2003); patients who may have wanted an appointment on the 24^{th} and 25^{th} had to wait until the 26^{th}. Second, December 26, 2003 falls on a Friday which typically experiences a higher than normal number of appointments due to the scheduling algorithms search pattern, which is detailed in Chapter 4.4.2.4.

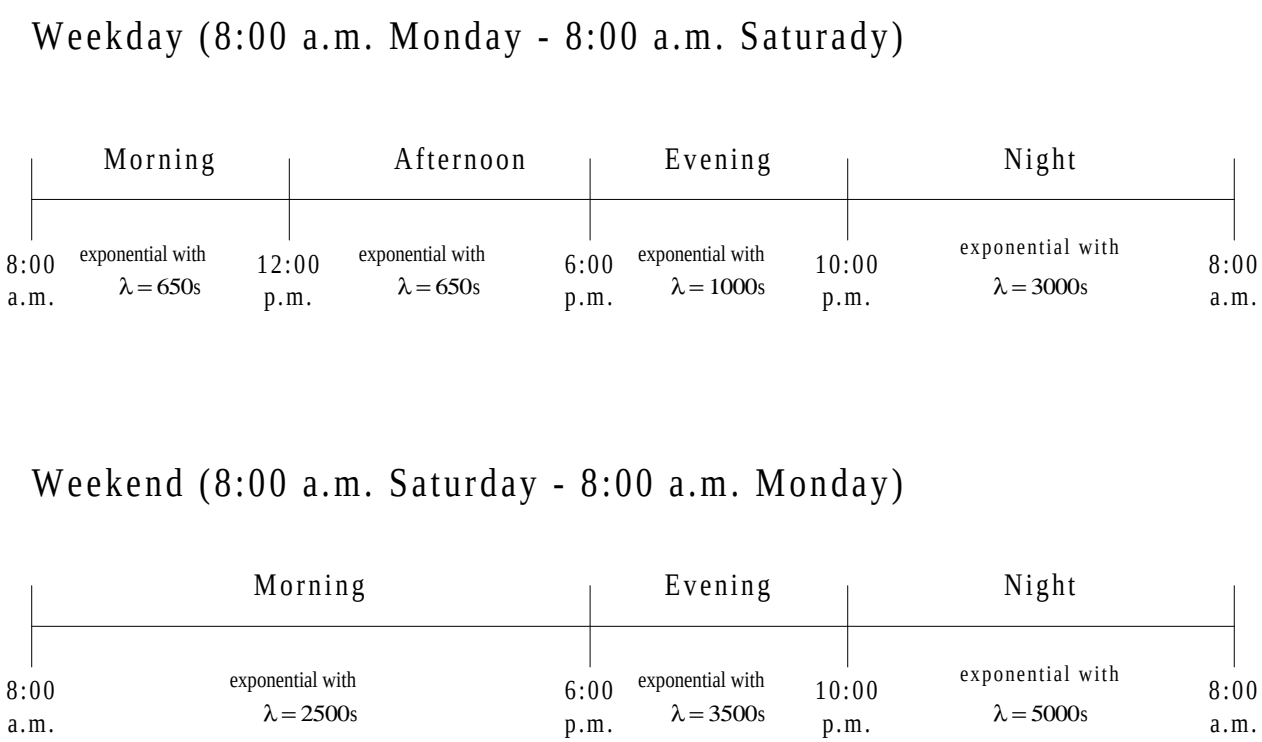


Figure 4.12: Call Interarrival Rate into the Queston Information Center

4.4.1.3 Call Interarrival Rates

The number of phone calls generated at the Dallas clinic area and routed to the Queston Information Center ultimately impacts the number of patients entering the clinic. Once again, due to the economical and technical infeasibility of collecting actual phone call data, the domain experts supplied an estimate of the daily number and type of calls that a clinic receives. Using an iterative process of trial runs with the input data, presenting the results to the domain experts, and making readjustments, the interarrival rate was systematically shaped into both a realistic and a functional form for use in the Queston simulation model (see Figure 4.12).

The arrival of phone calls into the Queston Information Center is modeled as an inhomogenous Poisson process. During the weekday, the call arrival rate is greatest during the *morning time period* (between 8:00 AM and 12:00 PM) and *afternoon time period* (between 12:00 PM and 6:00 PM). The rate then decreases through the *evening time period* ()between 6:00 PM and 10:00 PM), and reaches the lowest rate during the *night time period* (between 10:00 PM and 8:00 AM). Similarly, during the weekend, calls arrive at a greater rate during the *day time period* (between 8:00 AM and 6:00 PM), decreases through the *evening time period* (between 6:00 PM and 10:00 PM), and reaches its lowest rate during the *night time period* (between 10:00 AM and 8:00 AM). In general, calls arrive less frequently during the weekend, defined as the time between 8:00 AM Saturday through 8:00 AM Monday, compare to the weekdays. To model this inhomogenous Poisson process in the Queston simulation model, the technique of thinning was employed [Lewis and Shedler 1979]. The model generates calls using a stationary Poisson process with the maximum

arrival rate $\lambda^M = \max_t \{\lambda(t)\}$, where $\lambda(t)$ is the arrival rate at time t of the simulation run, and rejects all calls with probability $1-(\lambda^\tau)/(\lambda^M)$, where $\lambda^\tau = \{\lambda(\tau)\}$ is the arrival rate during the time period of interest [Law and Kelton 1991]. This Poisson process with the rate λ^M has the property that the interarrival time are identically and independently distributed exponential random variables with mean $1/\lambda^M$. Table 4.3 shows the rate of the different time periods as wells as the probability of rejecting or accepting calls during that time period based on $1/\lambda^M = 650$ seconds.

Table 4.3: Thinning Probabilities For Each Time Period

Time period	Mean Exponential Call Interarrival Rate (sec)	Probability of Rejecting	Probability of Accepting
Weekday Morning	650.0 $(1/\lambda^M)$	0.0%	100.0%
Weekday Afternoon	650.0	0.0%	100.0%
Weekday Evening	1000.0	35.0%	65.0%
Weekday Night	3000.0	78.3%	21.7%
Weekend Day	2500.0	74.0%	26.0%
Weekend Evening	3500.0	81.4%	18.6%
Weekend Night	5000.0	87.0%	13.0%

In order to identify the current day as a weekday or a weekend in the simulation model, a modulus function formula was developed. If N represents the current day during the simulation run (N = 1, 2, 3, … 457), then D (day of the week) = (N)modulus(7), the remainder of N divided by 7. This formula yields a number between 0 and 6. Since the first day of the run (N=1) is Tuesday, October 1, 2002, D = (1)modulus7 = 1 is assigned as a Tuesday. The following table shows how the days from October 1, 2002 to December 31, 2003 are assigned.

Table 4.4: Day of the Week Tabulation

Simulation Run Date	Simulation Day Number	D	Day of the Week
October 1, 2002	1	1	Tuesday
October 2, 2002	2	2	Wednesday
October 3, 2002	3	3	Thursday
October 4, 2002	4	4	Friday
October 5, 2002	5	5	Saturday
October 6, 2002	6	6	Sunday
October 7, 2002	7	0	Monday
October 8, 2002	8	1	Tuesday
…	…	…	…
December 29, 2003	455	0	Monday
December 30, 2003	456	1	Tuesday
December 31, 2003	457	2	Wednesday

4.4.2 Patient Input data

4.4.2.1 Patient Definition

Information defining the patient is an important piece of information provided by the input data collection effort, and critical to the Queston simulation model. Unfortunately, it also proved to be very challenging to identify and validate. The American Medical Association's (AMA) *Physician's Current Procedural Terminology* [AMA 1996] codifies patient evaluation and management services provided in a physician's office. The Physician's Current Procedural Terminology defines the characteristics of five general patient levels (described in Table 4.5) for both a new patient and an established patient and these were used by our domain experts as a general guideline in developing the input data for the Queston simulation model patient input data.

Table 4.5: American Medical Association's Categorization of Patients

	New Patient	Established Patient
Level 1	- code 99201 - a problem focused history and examination - straightforward medical decision making - approximately 10 minutes of face to face time with the physician - e.g., filling prescriptions, sunburns,	- code 99211 - patient does not require the presence of a physician - minimal problem - approximately 5 minutes of face to face time with the physician - e.g., test results, blood pressure check
Level 2	- code 99202 - an expanded problem focused history - an expanded problem focused examination - straightforward medical decision making - approximately 20 minutes of face to face time with the physician - e.g., severe cystic acne, urinary infection, allergic rhinitis	- code 99212 - a problem focused history and examination - straightforward medical decision making - approximately 10 minutes of face to face time with the physician - e.g., sore throat and headaches, ankle sprains, poison oak exposure
Level 3	- code 99203 - a detailed history and examination - medical decision making of low complexity - approximately 30 minutes of face to face time with the physician - e.g., lower back pain radiating, acute knee injury	- code 99213 - expanded problem focused history - expanded problem focused examination - medical decision making of low complexity - approximately 15 minutes of face to face time with the physician - e.g., diabetes, cirrhosis
Level 4	- code 99204 - a comprehensive history and examination - medical decision making of moderate complexity - approximately 40 minutes of face to face time with the physician - e.g., chest pain, diabetes	- code 99214 - a detailed history and examination - medical decision making of moderate complexity - approximately 25 minutes of face to face time with the physician - e.g., rheumatoid arthritis, enteritis
Level 5	- code 99205 - a comprehensive history and examination - medical decision making of high complexity - approximately 60 minutes of face to face time with the physician - e.g., cancerous, heart murmurs	- code 99215 - a comprehensive history and examination - medical decision making of high complexity - approximately 40 minutes of face to face time with the physician - e.g., lymphoma, memory loss

Note that the complexity of the patient's ailment increases with each successive level. This results in an increased burden on the physician's time and decision making abilities with each higher patient level. Using these five general categories , the domain experts expanded their definition of the patients into ten distinct categories for use in the Queston simulation model. Table 4.6 presents the Queston simulation model's ten categories of patients. Similar to AMA's level 1 patients, category 1 patients require only minor services and are generally of very low severity. On the other hand, category 5 patients, like level 5 patients, may require immediate service for a life threatening illness or condition. Furthermore, the domain experts included both pre-visit patients (admitted for tests or results), and new patients as two separate categories.

Table 4.6: Queston Simulation Model Patient Category Definition

Patient Category	Example of Ailment or Service Needed
1A	Blood Pressure Check, Tuberculosis, Test Reading
1B	Immunization, Phlebotomy
1C	Dressing Change
2	Sore Throat, Fever, Fatigue, Headache
2PV	Pre-visit for Category 2 Patients
3	Hypertension, Diabetes, Asthma, Flu
3PV	Pre-visit for Category 3 patients
4A	General New Patients
4B	Rheumatoid Arthritis
5	Chronic Ailment Complications

Patient Category: _____

Probability of Occurrence: _____

Scheduling Lead Time Distribution: _____

Probability of No-Shows: _____

Revenue Generation Distribution: _____

Cost Distribution: _____

Patient Time Slot Scheduling Rule: _____

Normal Visit Information:

	Probability of	Distribution/Probability in the Utilization of:						Parameters: mean and standard deviation (in minutes)					
	Occurrence	Clerical	Techn.	Med. Asst.	Nurse	NP/PA	Physician	Clerical	Techn.	Med. Asst.	Nurse	NP/PA	Physician
Registration	1		n/a	n/a	n/a	n/a	n/a		n/a	n/a	n/a	n/a	n/a
Check-In	1	n/a	n/a					n/a	n/a				
Pre-Exam	0≤ ≤1	n/a						n/a					
Exam	0≤ ≤1	n/a	n/a					n/a	n/a				
Post Exam	0≤ ≤1	n/a						n/a					
Exit Interview	0≤ ≤1	n/a	n/a					n/a	n/a				
Check-Out	1		n/a	n/a	n/a	n/a	n/a		n/a	n/a	n/a	n/a	n/a

Figure 4.13a: Normal Visit Patient Category Collection Form

Patient Category: _____

Probability of Occurrence: _____

Scheduling Lead Time Distribution: _____

Probability of No-Shows: _____

Revenue Generation Distribution: _____

Cost Distribution: _____

Patient Time Slot Scheduling Rule: _____

Pre-Visit Information (if necessary):

	Probability of	Distribution/Probability in the Utilization of:						Parameters: mean and standard deviation (in minutes)					
	Occurrence	Clerical	Techn.	Med. Asst	Nurse	NP/PA	Physician	Clerical	Techn.	Med. Asst.	Nurse	NP/PA	Physician
Registration	1		n/a	n/a	n/a	n/a	n/a		n/a	n/a	n/a	n/a	n/a
Check-In	1	n/a	n/a					n/a	n/a				
Pre-Exam	1	n/a						n/a					
Check-Out	1		n/a	n/a	n/a	n/a	n/a		n/a	n/a	n/a	n/a	n/a

Figure 4.13b: Pre-Visit Patient Category Collection Form

Once patients were categorized, input data on each patient category was provided by the domain experts. Two forms were created to assist the domain experts in identifying all the key aspects of each patient category (see Figures 4.13a and 4.13b). The first form collected information on normal visit category patients, while the second form collected information pertaining to pre-visit category patients (category 2PV and 3PV only). The domain experts identified the following information for each patient category:

- the probability of occurrence (how likely this patient will be visiting the clinic)
- the min/max/mode of their scheduling lead time (number of days between the patient's phone call and the appointment)
- probability of a no-show
- the min/max/mode of the amount of revenue and cost generated
- the most likely time slot scheduling rule (i.e., how the patient will search a particular day for an available appointment time slot)
- the probability of undergoing each of the seven service processes (see Figure 4.14)
- the probability of requiring a particular type of medical staff members for each service process (e.g. requiring a physician 50%, a physician assistant 30%, or a nurse 20%, for an examination)
- the min/max/mode of the time to service the patient by each medical staff members

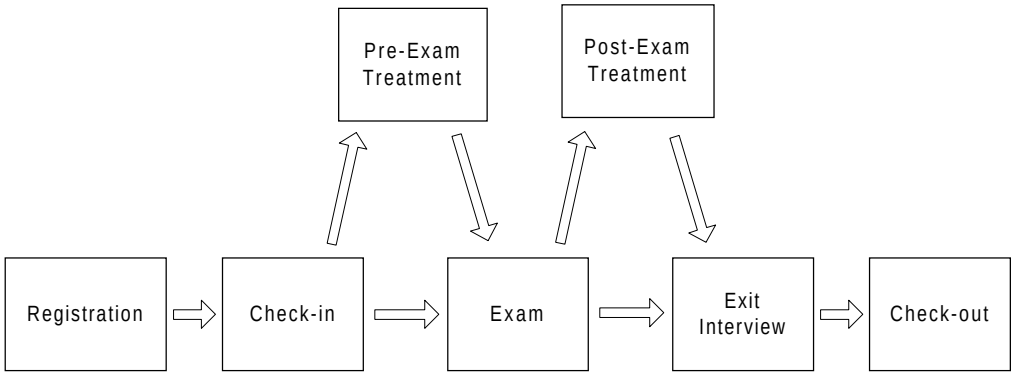


Figure 4.14: Patient Service Flow

The input data for the distribution on the scheduling lead time, revenue and cost generated, and the time to process were in the form of the minimum, maximum, and the mode times. These values were then fit to a triangular distribution, based on guidance from the domain experts. The process of fitting the input data for each patient category was an iterative process of reviewing the input data from the domain experts, conducting trial runs with the input data on the Queston simulation model (as well as analytical calculations of the mean times), and presenting the results back to the domain experts. In this iterative manner, the patient category input data was created until the domain experts were satisfied with the results. The final compilation of the patient category input data is presented in Appendix A.2.

4.4.2.2 Walk-in Patients

The Queston simulation model allows walk-in patients to enter the clinic to accurately measure demand on the clinic's resources. The domain experts indicated that a typical clinic has a mean of four walk-in patients per day. The walk-in patients were generated using the same probability of occurrence as the regular (scheduled) patients and pre-visit patients. In the Queston simulation model, walk-in patients are generated in the Entrance object and admitted in the next available time slot, or in a scheduled time slot if a patient is late (for their appointment) by more than 5 minutes. Note that a late arriving patient is still admitted as scheduled and is serviced by the physician the patient was scheduled with. Moreover,

once in the clinic, walk-in patients exhibits the same behavior as a scheduled patient, including having companions.

The arrival of walk-ins in the Queston simulation model is modeled as a Poisson process. To create four walk-in patients per day (on average), the Queston simulation model would need to generate a walk-in patient an average of every 21,600 seconds (86,400/4). However, the four walk-in patients had to be created during office hours and not in the middle of the night. To accomplish this, the model employed the acceptance/rejection technique [Law and Kelton 1991]. This technique can be used to create walk-in patients, since these patients arrive according to a Poisson processes. To apply this technique, walk-in patients are created at a constant arrival rate to maintain an average of four walk-in patients during the office hours. The walk-in patients that fall within the office hours are accepted and the ones generated outside office hours are rejected. Since the clinic business hours are 7.5 hours long, four walk-in patients every 7.5 hours equates to one walk-in patient every 6,750 seconds. In the Queston simulation model, walk-in patient interarrival times are independently and identically distributed (IID) exponential random variables with mean 6,750 seconds, and based on equation 4.1, either accept or reject each such arrival. When a walk-in patient is generated, the system time is checked to determine if it is between 9 AM and 4:15 PM If this condition is satisfied, then the walk-in patient is accepted, the next walk-in patient is scheduled, and this new walk-in patient waits for an opportunity to enter the clinic. If the condition fails, another walk-in patient is scheduled, and the current walk-in patient is destroyed.

$$32{,}400 + 86{,}400\,(N-1) \leq t \leq 59{,}400 + 86{,}400\,(N-1)$$

where N is the day number from 1 to 457, and t is the time the walk-in patient is generated

Equation 4.1: Acceptance/Rejection Condition

To address walk-in patients arriving during the lunch break several policies were considered (see Figure 4.15):

1) reject all walk-in patients arriving into the clinic during the lunch break
2) reject all walk-in patients arriving into the clinic during the lunch break but increase the interarrival rate accordingly
3) reschedule walk-in patients for a later (random) time
4) reschedule walk-in patients for the opening time of the clinic after the lunch break (1 PM)

Policy one would result in a mean of 3.75 walk-in patients per day. Policy three would result in a mean of less than four walk-in patients per day, since patients can be rescheduled for when the clinic is closed (and then rejected). However, both policy two, after readjusting the interarrival rate, and policy four would result in a mean of four walk-in patients per day. The domain experts determined that policy four provides the most realistic scenario and was used in the Queston simulation model. In real life, walk-in patients do arrive to a clinic during lunch break, and upon discovering when the clinic will reopen, reschedule themselves to when the clinic reopens. In the Queston simulation model, if walk-in patients arrive during the lunch break, they reschedule themselves to arrive at t=46,800 + 86,400*N (time the clinic reopens after the lunch break), where N is the day number.

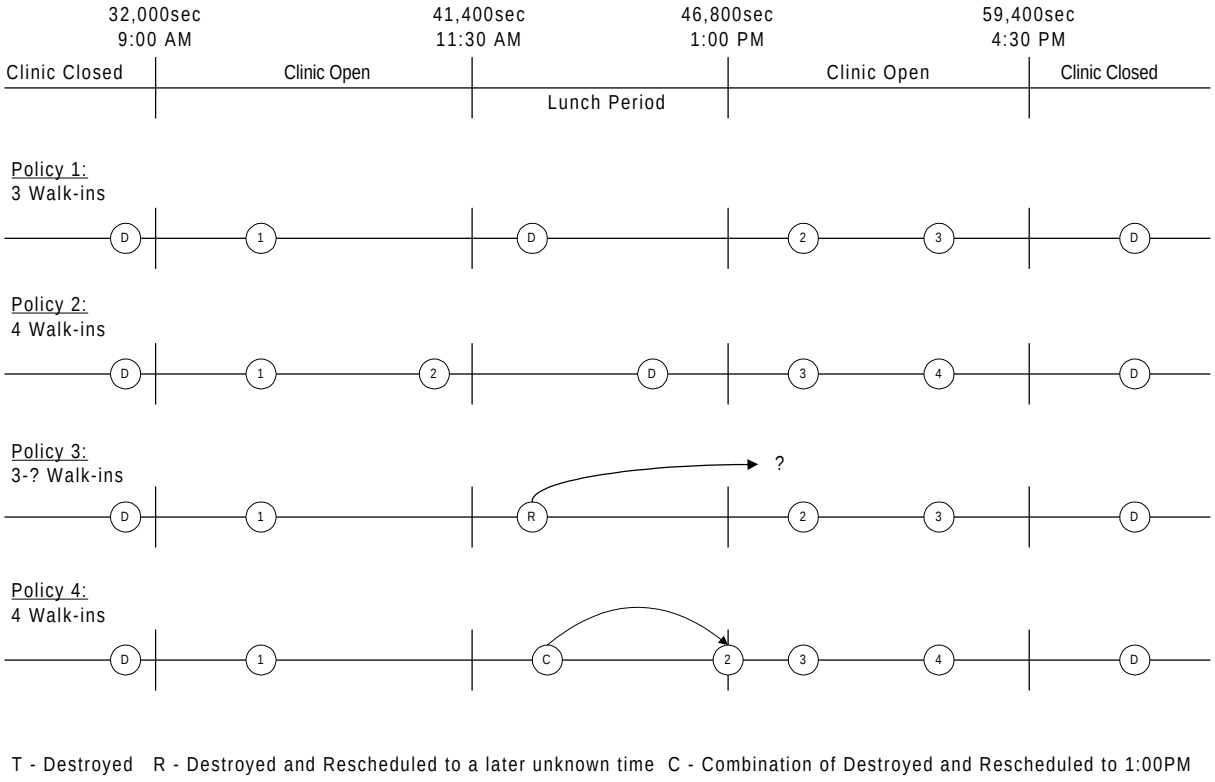
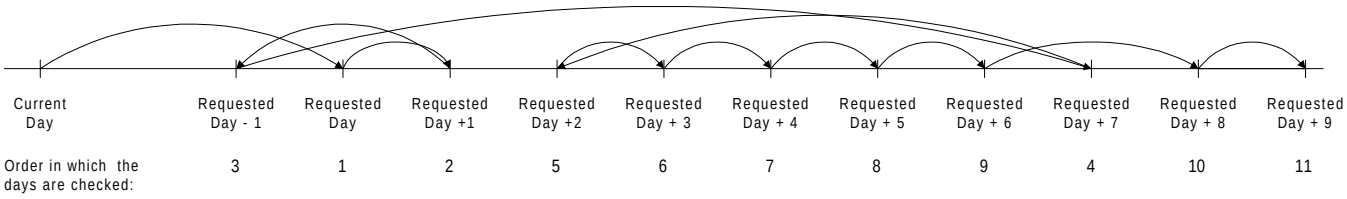


Figure 4.15: Lunch Break Walk-In Patient Policies

Specific Appointment Scheduling Algorithm



Sequential Appointment Scheduling Algorithm

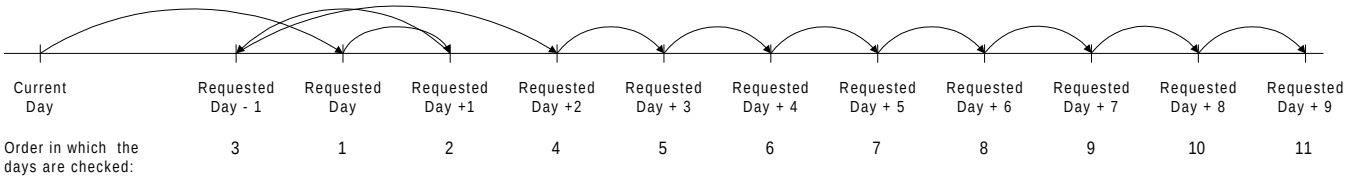


Figure 4.16: Appointment Day Search Algorithms

4.4.2.3 No Shows and Arrival Deviations

Scheduled patients often do not arrive to the clinic exactly at the time for their appointment and occasionally do not arrive at all. To capture how this affects in the Queston simulation model, the domain experts assigned a no-show probability for each category. When scheduled patients arrive to the clinic, they are randomly selected as a no show (using the no-show probability) and destroyed. If the patient is not deemed a no-show, they enter the clinic exactly on time, earlier than scheduled, or later than scheduled, with arrival distribution normal with mean zero and standard deviation 250 seconds. This was obtained from observing actual patients entering the clinic during the time study at the Christiansburg, Virginia clinic.

4.4.2.4 Patient Time Slot Scheduling Rules and Appointment Day Scheduling Algorithms

The patient time slot scheduling rule is a systematic method by which patients in each category schedule the time of day for their appointment. Four time slot scheduling rules were adopted by the domain experts for the Queston simulation model. These are described in Table 4.7.

The domain experts determined a combination of time slot scheduling rules to use for each patient category. For example, 50% of category 4 patients use time slot scheduling rule ALL, 25% use rule AM, and 25% use rule PM. Three of the four rules are sequential, since clinics often schedule most of their patients at the beginning of the morning or afternoon work

period in order to have some flexibility at the end of the period and to avoid working into the lunch break or overtime.

Table 4.7: Time Slot Scheduling Rules

Rule	Description
S	Specific time of day. The patient requests one specific time during the day for the appointment. If that time is unavailable then another date is searched.
AM	Sequentially only during the morning. The patient requests an appointment anytime during the morning office hours. The appointment book is searched sequentially from 9 AM to 11:15 AM for an opening. If that time is unavailable then another date is searched.
PM	Sequentially only during the afternoon. The patient requests an appointment anytime during the afternoon office hours. The appointment book is searched sequentially from 1 PM to 4:15 PM for an opening. If that time is unavailable then another date is searched.
ALL	Sequentially all day. The patient requests an appointment anytime during the day. The appointment book searches for an opening during the day in two ways: 1) sequentially from 9 AM to 4:15 PM, or 2) from 1 PM to 4:15 PM followed by a search from 9 AM to 11:15 AM If an available time slot cannot be found then another date is searched.

Scheduling lead time is the number of days from the day a call is made to the Queston Information Center requesting an appointment to the day the appointment time is initially check in. Category 1 and 2 patients often request an earlier date than the average category 3, 4, or 5 patients. This is due to the minor but urgent nature of the ailment such as the need to alleviate allergies, change dressings, and treat a fever, whereas, category 5 patients may be dealing with arthritis, planned surgery, and chronic illness, which can be planned for further into the future. New patients tend to have a longer scheduling lead time than all other categories. A minimum, a maximum, and a mode number of days were provided for each patient category by the domain experts. A triangular distribution using the minimum, maximum, and mode values generated a lead time for each patient that called to schedule an appointment.

Search algorithm were developed to determine how the patient is likely to search for a date when an appointment is requested. Once again, after numerous consultations with the domain experts, two search algorithms were agreed upon - a *Specific Appointment Day Search algorithm* and a *Sequential Appointment Day Search algorithm.* Figure 4.16 graphically illustrates how these algorithms search for the dates, based on the patient category's time slot scheduling rule and scheduling lead time.

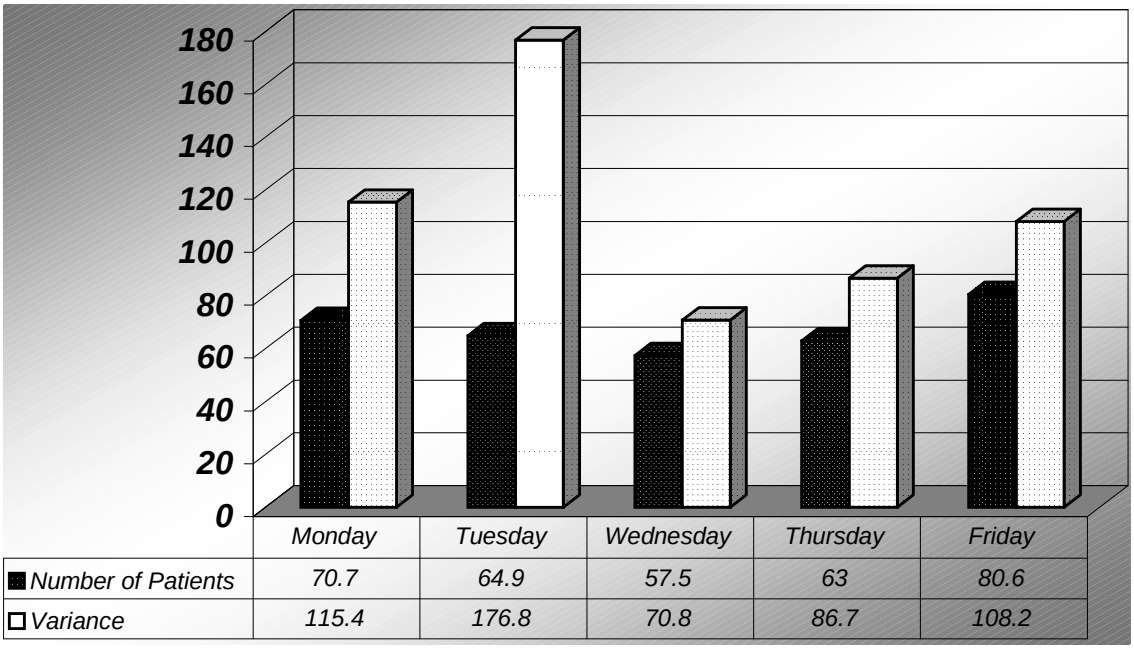
The Specific Appointment Day Search algorithm is employed when a patient is looking for a specific time slot (using time slot scheduling rule S). The algorithm first adds the scheduling lead time (days) to the current day (day the phone call requesting an appointment occurred) to arrive at the requested day. If the two time slots for the requested appointment time (two time slots since each 15 minute period has two time slots) is unavailable, the next day after the requested day is searched. On failing to find an available time slot, the algorithm continues its search on the requested day minus one day, requested day plus seven, requested day plus two, requested day plus three, requested day plus four, and so on, until a day is found. The domain experts indicated that a person requesting a specific time was more likely to search for that same time one week ahead of the requested date before searching two, three, etc., days ahead due to some time or day constraint they may have.

The Sequential Appointment Day Search algorithm is used when a patient is scheduled with one of the sequential time slot scheduling rules (rules AM, PM, or ALL). Patients using the AM time slot scheduling rule start their search for an open time slot at 9 AM and searches

forward for an available time slot until 11:15 AM Patients using the PM time slot scheduling rule start their sequential search at 1 PM and searches forward for an available time slot until 4:15 PM Forty percent of the patients using the ALL time slot scheduling rule start at 9 AM and search forward for an available time slot until there are no more time slots to be searched. The remaining sixty percent of the patients using the ALL time slot scheduling rule start their sequential search at 1 p.m and search forward for an available time slot until 4:15 PM. Note that if a PM time slot cannot be found, then the algorithm starts to search sequentially from 9 AM until 11:15 AM Like the Specific Appointment Day Search algorithm, the Sequential Appointment Day Search algorithm arrives at the requested day in a similar fashion, and upon failing to find a time slot using the appropriate time slot scheduling rule, searches the requested day plus one and requested day minus one as before. Unlike the Specific Appointment Day Search algorithm however, it next searches the requested day plus two, requested day plus three, and so on. Patients using this time slot scheduling rule are typically more flexible in their search for a day than those searching for a specific time slot.

For both algorithms, if the requested day falls on a Saturday, the appointment day is rescheduled to the prior Friday. If the requested day falls on a Sunday, the appointment day is rescheduled to the following Monday. For example, if the first day checked for an time slot falls on a Saturday, the algorithm checks the prior Friday for an time slot. However, if the requested day plus N, where N = 1, 2, …457, falls on a Saturday or a Sunday, then the algorithm continues its search on the following Monday (i.e., if after checking several days unsuccessfully for an time slot, the algorithm selects a Saturday or a Sunday, the algorithm moves the search to the following Monday).

There is a higher probability of searching for an appointment on a Monday or a Friday compared to Tuesday, Wednesday, or Thursday, since the algorithm reschedules Saturday or a Sunday searches to a Monday or a Friday. This results in additional appointments being scheduled on a Monday or a Friday than on the other days in the week. Figure 4.17 shows the mean and the variance of the number of patients admitted into the clinic by day of the week from a one year Queston simulation model run. Note that Monday begins with a large number of patients, then the number steadily decreases during the middle of the week, and then increases after Wednesday to peak on Friday. These results were validated by the domain experts as being typical of most family practice clinics.



	Monday	Tuesday	Wednesday	Thursday	Friday
Number of Patients	70.7	64.9	57.5	63	80.6
Variance	115.4	176.8	70.8	86.7	108.2

Figure 4.17: Number of Patients Admitted into the Clinic Per Day

4.4.3 Random Variate Stream

The VSE provides an extensive library of twenty two continuous random variate streams and five discrete random variate streams for modelers to use when developing a simulation model [Balci et al. 1996]. The Queston simulation model uses the Exponential, Uniform, Triangular, and the Normal Random Variate Streams built into the VSE software package. In the Queston simulation model, random numbers were generated to determine:

- The patient arrival deviation time (i.e., the deviation in the patient's arrival time from their scheduled appointment). This time is generated from a normal distribution with mean 0 seconds and standard deviation 250 seconds,
- The number of new patients per physician per day. This is generated from a discrete uniform distribution with a minimum 0 and a maximum 4.
- The interarrival time between phone calls during a specific time of day and day of week. This is generated from an exponential distribution with mean 650 seconds and then rejected with a probability of 1 – (650)/(current mean interarrival rate)
- The initial time slot number for a patient using an S time slot scheduling rule. These are generated from a discrete uniform distribution with a minimum 0 and a maximum 43.
- The interarrival time between walk-ins. This is generated from an exponential distribution with mean 6250 seconds. The walk-in patients are then thinned (see Chapter 4.4.2.2 for details).
- The patient's service time for particular procedures (e.g., check-in, registration, examination) by particular medical staff member (medical assistant, nurse, physician assistant, physician). This is generated from a triangular distribution with a minimum, maximum, and a mode listed in the Revised Patient Category Input Data in Appendix A.2.
- Whether a patient is a no-show. This is determined using a *Bernoulli trial* - a method of assigning a value of either success or failure based on the probability of success *p*.

If *p* (the probability of a no-show for the patient) is greater than a random number generated using U(0,1) , then the patient is tagged as a no-show and is destroyed when the patient is created.

- The time slot scheduling rule of patient categories with more than one possible time slot scheduling rule. Let P_S, P_{ALL}, P_{AM}, and P_{PM} represent the probability of each of the four time slot scheduling rule (see Chapter 4.4.2.2) for all patient categories, where $P_S + P_{ALL} + P_{AM} + P_{PM} = 1$. A U(0,1) random variate *R* is generated and used as follows:
  - if $R \leq P_S$ then the patient's time slot scheduling rule is S,
  - else, if $R \leq P_S + P_{ALL}$ then the patient's time slot scheduling rule is ALL,
  - else, if $R \leq P_S + P_{ALL} + P_{AM}$ then the patient's time slot scheduling rule is AM,
  - else, the patient's time slot scheduling rule is PM,
- Whether a patient needs a particular service process (e.g., registration, check-in, exit-interview). Using a Bernoulli trial where *p* is the probability of having a particular service process, if *p* is greater than a U(0,1) random variate, then the patient will require the service process; otherwise the patient will not require the service process.
- Which type of medical staff member will be servicing the patient in a particular service process. Let P_{MA}, P_N, P_{PA}, and P_{PH} represent the probability of the patient requiring a medical assistant, a nurse, a physician assistant, or a physician, respectively, for a particular service process, where $P_{MA} + P_N + P_{PA} + P_{PH} = 1$. A U(0,1) random variate *R* is generated and used as follows:
  - if $R \leq P_{MA}$, then the patient will be treated in the particular service process by a medical assistant,
  - else if $R \leq P_{MA} + P_N$, then the patient will be treated in the particular service process by a nurse,
  - else if $R \leq P_{MA} + P_N + P_{PA}$, then the patient will be treated in the particular service process by a physician assistant,
  - else, the patient will be treated in the particular service process by a physician.

- The patient's scheduling lead time. This is generated from a triangular distribution with a minimum, maximum, and a mode listed in the Revised Patient Category Input Data in Appendix A.2.

4.5 Data Collection

Throughout the simulation life-cycle, obtaining sufficient input data was the most difficult aspect of the modeling effort. The domain experts provided invaluable initial input data, based on their experience and judgement. To assist the domain experts and to glean additional insights into the manner clinics schedule and provide assistance to patients, a time study was undertaken at a family practice clinic in Christiansburg, Virginia. Unfortunately, the resources necessary to conduct a complete and comprehensive data collection at the Christiansburg clinic were unavailable. Note that the term *data* is in reference to the data collection effort in the clinic and refers to both the input data and the output measures of the Christiansburg clinic. Each day of data collection required three full time personnel to record the patients as they entered the clinic, received treatment, and exited the clinic. Consequently, only two days of data collection was allocated for this endeavor. The goal of this time study focused on observing and understanding the different processes within the clinic, whether it was scrutinizing the way the clinic schedules their patients, to the procedures the staff follows in treating the patients. However, it was infeasible to collect a sufficient volume of input data from the Christiansburg clinic to fit a distribution that the Queston simulation model would be able to use as its input data.

Visit Service Information

Date: _____

Clinic: _____

Scheduled Physician: _____

Actual Physician Seen: _____

Patient Scheduled Appointment Time: _____ No Show? Yes ____

Time when Patient entered Clinic: _____ Cancellation? Yes ____

Time when Patient exited Clinic: _____ New Patient? Yes ____

Type of Ailment (Category number): _____ Scheduled ___ or Walk-in ___

Number of Companions: _____ Referred ___ or Emergency ___

Age: _____ Male ___ or Female ___

Process	Starting Time	Finishing Time	Total Time of the Process	Serviced by:				
				Techn.	Med. Asst.	Nurse	NP/PA	Physician
Registration				n/a	n/a	n/a	n/a	n/a
Check-In								
Pre-Exam								
Exam								
Post Exam								
Exit Interview								
Check-Out				n/a	n/a	n/a	n/a	n/a

Figure 4.18: Patient Visit Data Collection Form

The time study was conducted during November 18 and 20, 1996, at a clinic in Christiansburg, Virginia. The second day of the time study obtained only one half of the first day's data, since the clinic closed after lunch To assist the data collectors in recording the correct and relevant data, a data collection form was created. This form is shown in Figure 4.18. The information collected on this form consisted of:

- the time a patient entered the clinic
- the time a patient exited the clinic
- whether the patient was scheduled or a walk-in patient
- the type of ailment being treated
- the elapsed time of each process (e.g., time of registration, check-in)
- which clerical or medical staff member treated the patient at each process
- how many patients did not show up (i.e., no shows)
- how many patients were new
- how many companions accompanied the patient

The complete dataset collected during the day and a half are presented in Appendix C.1 and C.2. Table 4.8 and 4.9 presents a summary of all the data collected.

Table 4.8: Processing Time of the Patients

	Registration	*Check-in*	*Pre-Examination*	*Examination*	*Post-examination*	*Exit-Interview*	*Check-out*
Count	11	37	38	31	3	7	39
Mean	222	67	191	577	403	226	201
Standard Error	87	7	20	45	108	92	26
Median	71	60	171	571	480	85	143
Standard Dev.	288	44	124	250	187	245	164
Sample Var.	83196	1953	15266	62259	35033	59785	26888
Skewness	1.4	2.3	2.0	0.5	-1.5	1.3	2.4
Range	760	226	615	1060	350	638	822
Minimum	12	14	55	128	190	32	27
Maximum	772	240	670	1188	540	670	849

Note: Times are in seconds per patient, regardless of patient categories.

$$\text{Standard Error} = \frac{S}{\sqrt{n}}$$

$$S = \sqrt{\frac{\sum_1^n (x_j - X)^2}{n-1}}$$

$$X = \frac{\sum_1^n x_j}{n}$$

$$\text{Skewness} = \frac{n}{[(n-1)(n-1)]} \sum \left[\frac{(x_j - X)}{S} \right]^3$$

where S is the standard deviation, n is the number of observations, x_j is the jth observation, and X is the sample mean

Equation 4.2: Equations for Sample Statistics

Table 4.9: General Information on the Clinic

Description	Observation
Number of Days Observed	1.5
Number of Physicians	1
Number of PA/NP	0
Number of Nurses	1
Number of Medical Assistants	1
Number of Clerical Staff	1
Number of Patients Seen	42
Number of Patients Scheduled	34
Number of New Patients	4
Number of Walk-ins	12
Number of No-shows	3
Overtime of the Clinic (1st day only)	0 minutes
Maximum Number of Companions	3
Minimum Number of Companions	0
Mean Number of Companions	.45

A number of insights were extracted from both the data and from observing the processes. One of the key finding was that the medical staff members (nurses and medical assistants) stayed busy throughout the day even though the time that they interacted with patients was relatively brief. They were observed filling out records, taking notes, and conducting other tasks associated with the patients. Based on this observation, the Queston simulation model was reconfigured to split the total time a medical staff member services the patients into two objects: one third of the total time allocated to a medical staff member interacting face-to-face with patients and the other two thirds of the total time for conducting supporting activities (e.g., note taking) away from the patients. This activity was not witnessed to a great extent for the physician hence the Queston simulation model included only a distribution of time for similar non-patient activities for the physician after each patient encounter.

Another observation was that the data in Table 4.9 showed processing times much shorter than those provided by the domain experts. In particular, the check-in times were almost three to four times shorter. Additionally, the data showed the skewness of almost all the processes to be positive. After being presented with these observations, the domain experts reevaluated the patient category input data and supplied a modified set of input data for each patient category. They decreased, and in one case (category 1B) increased, the times for the service processes. Furthermore, the mode of each triangular distribution was modified to reflect the positive skewness of each distribution. The changes to the minimum, maximum, and the mode are presented in the Revised Patient Category Input Data in Appendix A.2. The

resulting percent change in the patient's average length of stay (ALOS) for each patient category are calculated analytically and listed in Table 4.10.

Table 4.10: Change to the Patient's ALOS using the Revised Patient Category Input Data

Patient Category	Percent Change in the Patient's ALOS
1A	-3.7%
1B	-2.9%
1C	1.4%
2	-9.5%
2PV	-7.5%
3	-12.5%
3PV	-8.0%
4A	-22.7%
4B	-22.7%
5	-20.3%

The total number of patients admitted to the clinic during the time study (42 patients) was lower than what our one-physician simulation model would average during a day and a half (50.25 patients). This is explained by the clinic's scheduling policy which differed from the Queston simulation model. The clinic schedules patients an average of one patient per fifteen minutes, whereas, the simulation model schedules two patients per fifteen minutes. Moreover, the simulation model assumes that the QPN clinics would have a physician assistant to meet the additional demand, while this resource was not available at the clinic visited.

The data in Table 4.8 also shows that twelve walk-in patients were admitted; this number is much higher than the six walk-in patients the two physician clinic Queston simulation model

averages during a similar time period. The domain experts felt that this was an anomaly and that without collecting more data on this statistic, it would not be prudent to change their earlier assumptions based on this small sample.

The arrival behavior of patients to the clinic for their scheduled appointment also resulted in some modifications to the Queston simulation model. Up to this point, all patients in the simulation model arrived on schedule. However, after observing patients arriving as early as 34 minutes and as late as 11 minutes for their appointments at the clinic, the simulation model included an *arrival deviation* distribution for each scheduled patient. The arrival deviation from scheduled appointment time distribution is normal with mean zero and standard deviation of 250 seconds. The domain experts believed that this would provide a more realistic figure than those collected from the Christiansburg clinic (see Figure 4.19).

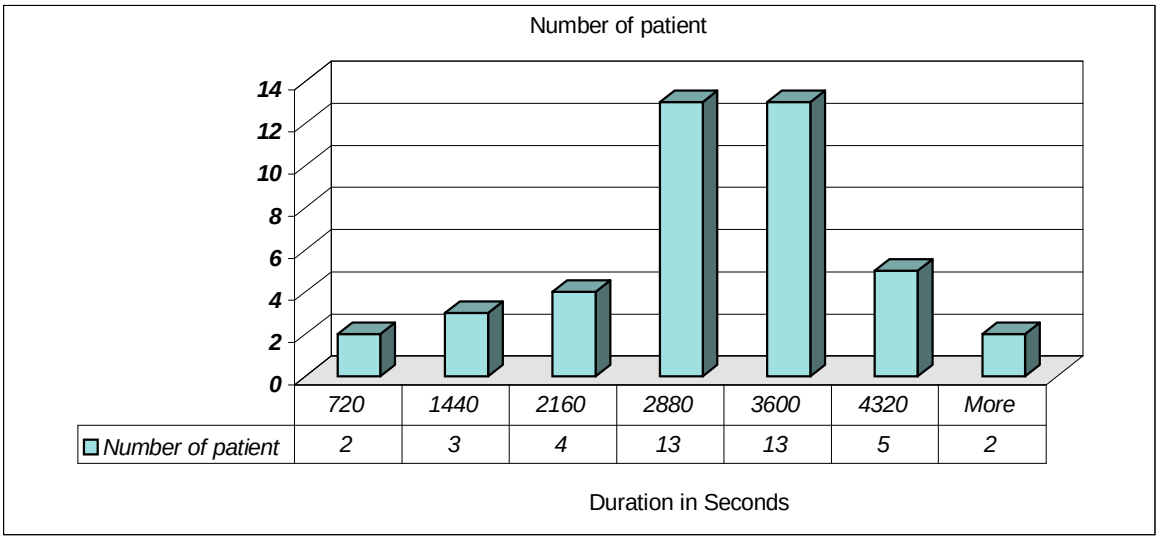


Figure 4.19: Arrival Deviation from Scheduled Appointment Time

Input data that could not be collected during the time study were the patient's scheduling lead times, the probability of occurrence for each patient category, and the cost and revenue generated for each patient. The scheduling lead time and the probability of occurrence of each patient category could not be collected because this would require a length of time study greater than several weeks, hence would be too costly in both time and in resources. The cost and revenue for each patient could not collected due to the confidentiality of such information.

The data collection effort proved to be very difficult, even in this one-physician clinic, since the data had to be gathered without interrupting the patient's service or being too conspicuous. The effort required to collect data in a multi-physician clinic would be even more challenging. However, in conducting this time study, the confidence in the Queston simulation model was further enhanced. The data collected from the clinic were presented to the domain experts along with the outputs from the simulation model. The domain experts were unable to discern which output came from which system. This test, called the *Turing test* [Schruben 1980], is described in detail in Chapter 6.2.3.

4.6 Transient Period Analysis

The mean scheduling lead time for an appointment request is 6.94 working days, hence allows the clinic appointment book to be filled quickly. This assumption was validated by comparing the mean number of patients admitted into the clinic for the first ten full weeks

(i.e., $\mu_{\text{first ten weeks}}$) following the warm up period versus the last ten full weeks (i.e., $\mu_{\text{last ten weeks}}$) of the same simulation run. Table 4.11 shows the 20 five day batch means for the number of patients admitted into the clinic for the first ten weeks after the warm-up and the last ten weeks of the simulated run. With the assumption that each batch mean is normally distributed, the p-value was obtained for a two-tailed hypothesis test [Levine et al. 1998]. The null hypothesis (H_0) states that the mean number of patients entering the clinic in the first ten weeks following the warm-up period is equal to the last ten weeks of the one year simulated run ($\mu_{\text{first ten weeks}} = \mu_{\text{last ten weeks}}$), while the alternate hypothesis (H_1) states that the two ten week samples are not equal. The student-t test statistic yields a p-value (two-tailed) of .908. Therefore, since the p-value is close to one, then the H_0 is not rejected in favor of the alternative hypothesis. This supports the assumption that the Queston simulation model appointment book has been quickly filled. For a detailed analysis on the initialization bias problem and the determination of the truncation point of the Queston simulation model refer to Swisher [1999].

Table 4.11: Warm Up Validation Data Points

Mean number of patients admitted into the clinic in the:	First Ten weeks	Last Ten weeks
	68.8	65.6
	68.4	68.4
	63.6	67.8
	62.6	65.8
	67.8	64.0
	67.4	70.8
	72.0	68.8
	63.0	63.2
	66.0	64.6
	66.0	64.8
Mean	66.5	66.4
Variance	8.6	6.0

4.5 Output Structure

Several output measures can be collected from the Queston simulation model. In one simulation run of a clinic with two physicians, three medical assistants, two nurses, one physician assistant, one operator, two registration booths, two check-in rooms, one specialty room, and six examination rooms, 101 separate files are created that provide output measures ranging from the number of walk-in patients to the number of calls each operator handled. All of the output files and file descriptions are displayed in Appendix B. However, in the experimentation and model development phase (summarized in Chapter 7), the output measures analyzed were

- average clinic overtime per day of the week
- number of total patients and number of clinic appointments in the morning and afternoon
- maximum number of patients in the internal waiting area and waiting room buffer
- total lunch time of the physician
- time distribution for the number of physicians busy
- ALOS for each patient category
- average waiting time per visit for each patient category
- average time in the internal waiting area or the examination room waiting for service
- time distribution for the number of check-in rooms or specialty room or examination rooms busy
- time distribution for the number of nurses or medical assistants or physician assistants busy
- utilization rate for each medical assistant, nurse, physician assistant, and physician

One objective for developing the Queston simulation model is to increase patient satisfaction. Unfortunately, it is difficult to directly measure whether a patient is satisfied with the services

rendered by the clinic without actually conducting a survey. In the Queston simulation model, several indicators of patient satisfaction are used to predict the level of satisfaction. The primary measure is the ALOS (i.e., the total time elapsed from when the patient enters to when they exit the clinic). This ALOS is calculated by subtracting the system time when the patient exits the clinic in the entrance object with the system time when the patient entered the clinic from the entrance object. A second measure is the average time waiting while in the clinic, either waiting for service at the registration booth, the check-in room, the specialty room, or the examination room. To determine each patient category's daily average waiting time in the clinic, the total patient visit time was subtracted by the total staff interaction times for each patient category each day. A third measure used is the number of patients waiting in the waiting room buffer. Patients in the waiting room buffer, unable to find an available chair, are most likely to be unhappy standing while waiting to be treated. The time spent inside the waiting room buffer is recorded by the patient object in the variable waitingRoomBufferWaitingTime. Upon exiting the clinic through the entrance object, the value of the waitingRoomBufferWaitingTime is added to the total for each patient category. At the end of the day (12:00 AM), the accumulated waiting time for each patient category is divided by the total number of patients for each patient category that entered the clinic that day.

Other goals of the Queston simulation model are to determine how to maximize staff utilization and facility utilization. These are effectively captured by the utilization rates for each object in the Queston simulation model. An assumption was made in the way that staff utilization output measures were captured. The time that a medical staff member is

physically engaged with a patient plus the note-taking time after each patient engagement is considered busy time.

Finally, since the simulation model is being executed for 457 days, and each day is considered a separate data point, it was decided to collect data for each day at 12 midnight. For example, the first data point collected will occur at 12:00:00 midnight on January 2, 2003 for the information from January 1, 2003.

Chapter 5 Queston Simulation Model Description

The Queston simulation model allows the end user to understand the healthcare clinic system while containing sufficient complexity to capture important characteristics of this system; the amount of detail included in the model is a function of the model's purpose and objectives. The Queston simulation model is programmed in VSE. The objects needed to build the Queston simulation model were not preprogrammed and available for use in a template in VSE. However, VSE provides all the tools necessary to create both user defined classes and methods (class and instance) for objects. Hence, the Queston simulation model was programmed in an English-like, object-oriented language to create and save the classes and methods (class and instance methods) that defines the objects in the Queston simulation model.

Since the complete Queston simulation model encompasses over eight thousand lines of code on four hundred pages, the object's class, class methods, and instance methods are summarized in this chapter. The complete code is available upon request from Biopop.

5.1 Object Classes and Methods

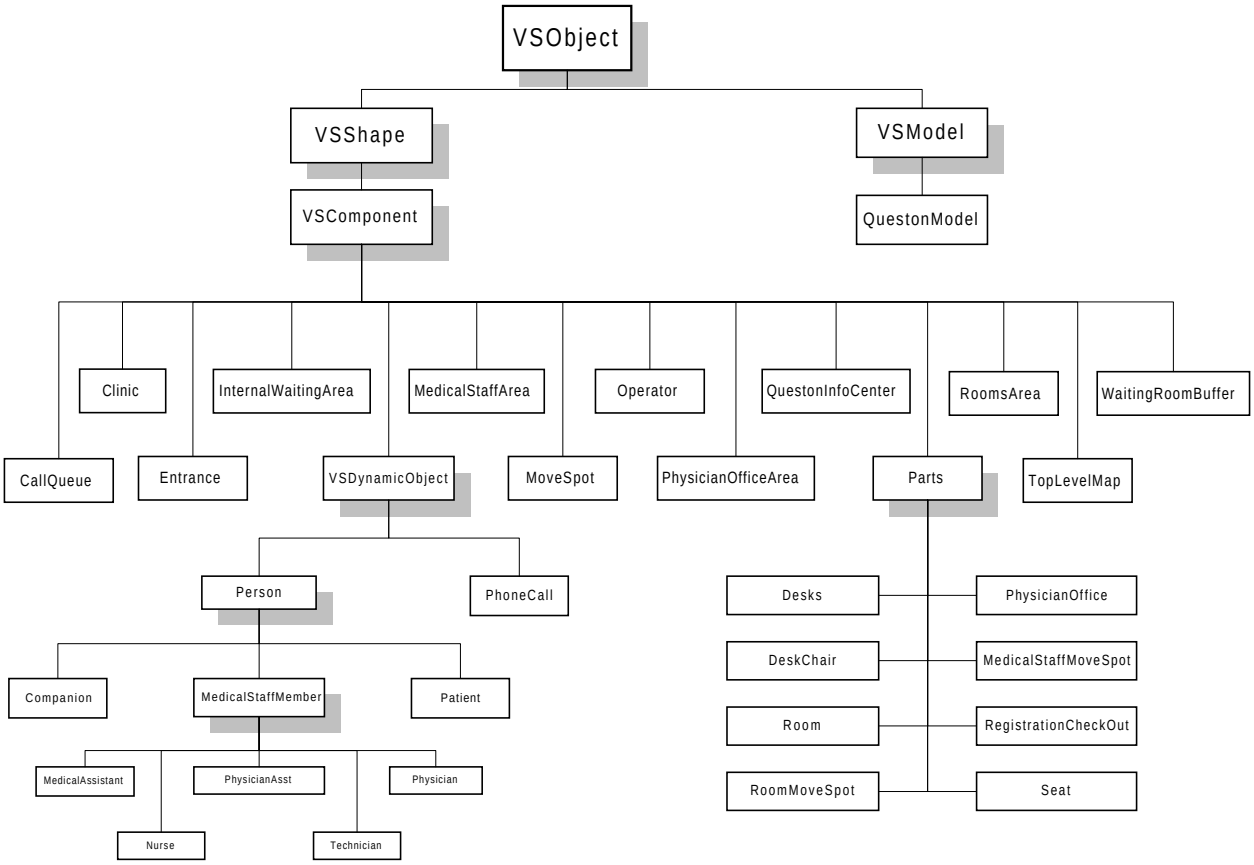


Figure 5.1: Queston Simulation Model Object Classes

As described in Chapter 3.2.2, each object in the Queston simulation model inherits functionality from both their class and superclass. The complete object class hierarchy is depicted in Figure 5.1. Note that the class hierarchy is *not* the same as the decomposition hierarchy of the Queston simulation model. For example, an object in the Room class inherits all the functionality and instance methods built (i.e., programmed) into the Parts class, which in turn inherits all the functionality of the VSObject class, and so forth. However, objects in the Room class cannot access instance methods or variables in the Person class without sending a message (via an instance method) requesting the information.

That information is delivered back (via another message), since the object of the Person class cannot directly access or pass the information to the object of the Room class.

All classes starting with the letters VS, such as VSObject or VSObject, have preprogrammed class and instance methods, and hence, objects that are subclasses of these VS classes will have access to the classes' methods. For example, since the method "destroyAtTime:" is a preprogrammed instance method in VSObject, any object with a class that is a subclass of VSObject can use this instance method. These classes and the associated class and instance methods are programmed by the software developers of VSE to assist the modeler when using frequently used methods. However, even with these preprogrammed methods (class or instance methods), modelers will still be required to program their own methods. Eventually, once a template is created, future models could potentially be created without programming a single line of code.

In the Queston simulation model, there are thirty-two user defined classes and three hundred and fifty user defined instance and class methods. Many of these methods are instance methods that request information on a variable, send information of a variable, or set a variable to a value between objects of different classes. Chapter 5.2.2 describes all the user-defined classes and some of the more important class and instance methods used to program the Queston simulation model in VSE. The complete method listing is displayed in Appendix E.

Each method listed in Chapter 5.2.2 has four specifications that are not displayed (due to the space constraints of this thesis). These specifications are *Parameters*, R*eturns*, *Declarations*, and *Logic*. Method parameters (either class or instance) are specified after each colon in the method's name and are declared in the *Parameters* section of the method's specification. These parameters can be an integer, character, class reference, object reference, enumeration, or a real number. The number of colons in the method indicates the number of parameters in the method; a one-to-one mapping exists between colons and parameters. For example, the instance method **patientNeedsPhysician:at:** is sent from a room object where a patient just entered the physician office area using the following partial code:

patientNeedsPhysician:2 at:self;

Since there are two colons in this instance method, two parameters are attached to this message and sent to the physician office area. The first parameter (an integer value of 2) indicates which physician the patient requires and the second parameter (an object reference to itself) indicate where the physician is needed. Although a colon in a method's name clearly indicates the presence of parameter, the method's name does not reveal if the method has a *Returns*. For example, the instance method **changeWalkInStatus** in the Patient class returns no value back to the object calling this instance method but the instance method **patientNumber** in the Patient class returns an integer to the object calling this instance method. In the *Returns* section of the method's specification, the parameters requested is "returned" or sent to the requesting object. In the *Declarations* section, local variables are declared for use in the method. These local variables are declared, used, and deleted each time the method is called. The last specification is the *Logic* of the method. All the code specific to the method is written in the *Logic* section of the method's specification.

Some of the predefined VSObject methods such as **dynObjArrived:**, **dynObjDeparted:**, **dynObjEnteredModel:**, and **dynObjAscended:** are listed in Chapter 5.2. These methods are automatically called by VSE each time a dynamic object arrives to a deep object (**dynObjArrived:**), departs a deep object (**dynObjDeparted:**), is created in the model (**dynObjEnteredModel:**), or ascends up from a deeper object to the current object (**dynObjAscended:**). The method is sent to the object it *entered into*, *departed from*, is *created in*, or *ascended from*. VSE allows the user to override the logic of any of its VS class's predefined methods by simply creating a method with the same name. For example, the Clinic class, which is a subclass of VSObject class, overrides the method **dynObjAscended:** method by creating a method with the same name called **dynObjAscended:** and including the logic specific for its functionality. For further details on this subject refer to the VSE User Manual [Balci 1996].

5.2 Description of Classes, Instance Methods, and Class Methods

5.2.1 Formating

In this chapter, only the user-defined classes and its methods are presented. The highest superclass will be defined first followed by its subclasses. The QuestonModel class that inherits from the superclass VSModel, is presented first, followed by the thirteen subclasses of VSObject. Next, the eight subclasses of the class Parts, are described. Finally, the ten

subclasses of VSDynamicObject are presented. Essentially, the classes will be described in order from top to bottom in Figure 5.1.

In the class descriptions, the class' superclasses are listed from left to right after the class name; each class is a subclass of those listed immediately to the right. For example, the class QuestonModel has the description

Inherits from: VSModel : VSObject

This means that the QuestonModel class is a subclass of VSModel, which in turn is a subclass of VSObject. As mentioned in Chapter 5.1, an object of any given class responds to the instance methods defined in the class to which it belongs, but it responds to any of the instance methods of its superclasses.

Next, a brief description and overview of each class is provided. This is followed by the class' instance methods and description, grouped into different categories based on the instance method's functionality in the class. For example, the instance methods **changeToAfternoonRate**, **changeToEveningRate**, and **changeToMorningRate** are all grouped together in the category Reset Call Rates because these methods are related in it's functionality; it resets the interarrival rates. Finally, the class methods are described and grouped into similar types of categories. The categories for each class and the methods in each category are presented alphabetically.

5.2.2 Queston Simulation User-Defined Classes

CLASS: QuestonModel

Inherits from: VSModel : VSObject

Class Description

The QuestonModel class defines and creates the calendar list for the 457 days. Additionally, object references are created to point to (i.e., reference) the clinic main door object, rooms area object, the clinic object, waiting room buffer object, internal waiting area object, medical staff office area object, physician office area object, and the Queston Information Center object. Furthermore, since this class is a subclass of VSModel, it inherits a number of methods that are essential to the simulation model (e.g., the instance method **clock** which returns the value of the current simulation clock).

Instance Methods

Category: Calendar and Scheduling

fillCalendarDateList

This method creates and initializes a list called calendarDate containing values from October 1, 2002 to December 31, 2003.

Category: Object Reference

calendarDate:

This method receives the date parameter, passed from another object and returns an object reference to the date on the calendar list. Futhermore, the method is used by the requesting object to reference a particular day on the calendar list.

clinic

This method returns the object reference to the clinic object.

internalWaitingArea
mainDoor
medicalStaffArea
moveSpot
physicianOfficeArea

questonInfoCenter

roomsArea

seat

waitingRoomBuffer

All of the above methods return an object reference to the respective object in the Queston simulation model.

Category Miscellaneous

availableRegistration

This method returns the object reference to the first registration window object available to assist a patient.

availableSeat

This method returns the object reference to the first empty seat object.

largestReal

This method returns the largest integer value of the seat number.

Category Notifications

checkInRoomIsAvailable

This method informs all the seats that service is available in a check-in room. If there are no patients waiting in the seats for a check-in room, then a message is sent to the waiting room buffer object stating that a check-in room is available.

patientIsLeavingClinic:

When a patient finishes their visit and leaves the clinic, this method checks the seats to see if the patient arrived with any companions and if so, sends the companions to the exit (main door object).

regCheckOutIsAvailable:

This method informs the seat objects that service is available at a registration window object. If there are no patients waiting in the seats for a registration window, then a message is sent to the waiting room buffer object stating that a registration window object is available.

Category Start Up

replicationStarted

This method is started at the beginning of each replication. In the case of the Queston simulation model, this method occurs just once. The method assigns the object references to the clinic object, internal waiting area object, main door object, medical staff office area object, move spots object, physician offices object, Queston Information Center object, rooms area object, all the seats objects, and the waiting room buffer object.

CLASS: TopLevelMap

Inherits from: VSObject : VSShape : VSObject

Class Description

This class defines the QuestonMedicalPractice object and assigns the object reference to the Queston Information Center object. Additionally, phone call objects created from the clinic object and ascended to the QuestonMedicalPractice are routed to the Queston Information Center object.

Instance Methods

Category: Notification

dynObjAscended:

This method routes phone call objects from the clinic object to the Queston Information Center object.

Category: Start Up

replicationStarted

This methods assigns an object reference to the object called “Queston Information Center”.

CLASS: QuestonInfoCenter

Inherits from: VSObject : VSShape : VSObject

Class Description

This class routes phone calls entering the Queston Information Center to the call queue and from the call queue to an operator. Furthermore, at model execution, a list of operator objects are created and initialized.

Instance Methods

Category: Access

listOfOperators

This method returns an object reference to the operator object list.

Category: Miscellaneous

availableOperator

This method checks each operator object (from one to *NUMBER_OF_OPERATORS*) for availability. If an available operator object is found, an object reference to that operator object is returned, otherwise the method returns a object reference value of nil.

Category: Notification

dynObjArrived:

This method routes phone calls into the call queue when a phone call arrives into the Queston Information Center object,.

dynObjAscended

This method executes when a phone call object ascends from the call queue deep static object. It directs the phone call object to the available operator object.

Category: Start Up and Shut Down

replicationStarted

At model execution, this methods creates a list of operator objects, initializes each operator object, and sets the image to the "Idle Operator Image". The number of operator objects created is determined by the value of *NUMBER_OF_OPERATORS* in the constants panel.

Category: Statistics

createInfoCenterOutputData

recalculateDistribution:

resetAndPrintDayEvent:

These methods create an output file and collect output measures on the operators, such as an operator's utilization rate. All of the output measures collected by this method are presented in Appendix B.

CLASS: CallQueue

Inherits from: VSObject : VSShape : VSObject

Class Description

The CallQueue class defines the call queue object in the Queston simulation model. The call queue object queues phone call objects to be served on a first-come-first-serve basis. It notifies and sends a phone call to an available operator upon notification from an available operator.

Instance Methods

Category: Notification

dynObjArrived:

Each time call enters the call queue, this method checks all the operators for availability. If an available operator is found, the operator number is set to the phone call's *destination* variable and the phone calls is sent to the call queue's exit. If no available operators are found, the call waits in the call queue. The variable destination is assigned to all dynamic object that indicates to which object the dynamic object will move to next.

dynObjDeparted:

This method executes each time a phone call departs the call queue. The total number of phone call remaining in the call queue is calculated and displayed on the call queue object.

operatorIsAvailable:

This method is invoked by an operator who just became available. If there is a phone call in the call queue, the phone call in the call queue with the longest length of time is notified that an operator is available. The available operator number is set to the phone call's *destination* variable.

Category: Statistics

createQueueOutputData
recalculateDistribution:
resetAndPrintDayEvent:

These methods create an output file and collect output measures on the call queue, such as the time 0, 1, 2, 3 …. phone calls are in the call queue. All of the output measures collected by this method are presented in Appendix B.

CLASS: Operator

Inherits from: VSObject : VSShape : VSObject

Class Description

This class defines the operator object. Phone calls are handled by the operators. Seventy percent of the time, the phone call results in the scheduling of a patient.

Instance Methods

Category: Access

isAvailable

This method returns a boolean of *True* if the operator is available and a *False* if the operator is not available

operatorNumber

This method returns the operator number.

Category: Notification

dynObj:stoppedEngagement

This method is invoked when an operator finishes a call. It creates a new scheduled patient if the call was successful (75% of the time). The scheduled patient is scheduled into the appointment book based upon their variables (e.g., lead time, time slot scheduling rule, physician required). The patient's scheduled arrival time is set or patients who will be no-shows are predetermined by setting the *stateOfPatient* variable to *NO_SHOW*. However, if the patient is not a no-show, the patient's *status* is set to *CREATED*. The *status* variable tracks the patient's progress from creation, through the service processes, and finally to its destruction.

dynObjArrived:

This method changes the operator image to the "Busy Operator Image" and the operator engages the phone call for a variable amount of time. This random length of time is determined in the operator class method **replicationStarted**.

Category: Settings

setAVailabilityTo:

setOperatorNumberTo:

These methods set an operator's availability and number to a given value.

Category: Statistics

createOperatorOutputData

resetAndPrintDayEvent

These methods create an output file and collect output measures on each operator, such as the total number of calls handled and the total number of calls resulting in a scheduled patient. All of the output measures collected by this method are presented in Appendix B.

Class Methods

Category: StartUp

replicationStarted

This class method creates two random variate streams. The first random variate stream is IID U(0,1). The second random variate stream generates the length of a phone call. This call length is an IID exponential random variable with mean *MEAN_PHONE_CALL_TIME*. The value of the *MEAN_PHONE_CALL_TIME* is set in the constants panel.

CLASS: Entrance

Inherits from: VSObject : VSShape : VSObject

Class Description

This class defines how the entrance object processes patients (both walk-in patients or scheduled) and companions into and out of the clinic. All objects exiting through the

clinic are destroyed here. Furthermore the entrance object collects a majority of the output measures, since all patients and companions enter and exit the clinic through this object. All of the patient's time related output measures (e.g., ALOS, total waiting times) are collected in the entrance object since the time a patient enters and exits the clinic can be easily recorded.

Instance Methods

Category: Access

overTimeHoursOfClinic

This method returns the value of the *overTimeHourOfClinic* variable.

Category: Notification

dynObjArrived:

This method routes arrivals based upon whether they are patients or companions and if they are exiting or entering the clinic. If an arrival is a companion (only possible if the companion is exiting since companions are created in the Entrance and cannot enter the Entrance unless arriving from the clinic), the companion is destroyed. If the arrival is a patient entering the clinic, the number of companions are determined and these companions are created. Furthermore, if the patient is a walk-in with the variable *stateOfPatient* equal to *THINNED*, then the patient is destroyed. This variable is set when the walk-in patient is generated in the **dynObjEnteredModel:** method. Additionally, if the *stateOfPatient* is equal to *NO_SHOW*, then the patient is destroyed. Finally, if the patient's variable *status* is *NEED_CHECKOUT*, meaning the patient just checked out and is exiting the clinic, the patient is destroyed.

dynObjEnteredModel:

This methods creates new walk-in patients. Using the technique of thinning, one walk-in patient is created on average every 6,750 seconds (see Chapter 4.4.2.2). Walk-ins created during lunch or outside office hours have their variable *stateOfPatient* set to *THINNED* and another walk-in patient is generated.

Category: Start Up and Shut Down

replicationStarted

This method creates the initial walk-in patient. Note that due to thinning, this initial walk-in patient is destroyed when arriving to the clinic, since the clinic is not open at model initialization.

Category: Statistics

createPatientOutputData

resetAndPrintDayEvent

These methods create an output file and collect thirty-six unique output measures (e.g., number of walk-in patients per day, ALOS of patients in each patient category per day, the daily overtime of the clinic, and the total number of companions per day). All of the output measures collected by this method are presented in Appendix B.

Class Methods

Category: StartUp

replicationStarted

This class method creates two random variate streams. The first random variate stream is IID U(0,1). The second random variate stream generates the random interarrival times for walk-in patients, distributed IID exponential with mean *MEAN_WALKIN_INTERARRIVAL_TIME*. The value of the *MEAN_WALKIN_INTERARRIVAL_TIME* is set in the constants panel

CLASS: Clinic

Inherits from: VSObject : VSShape : VSObject

Class Description

The clinic class plays an integral part in the Queston simulation model. It creates and sends phone calls to the Queston Information Center and it schedules patients in the appointment book (created in this class).

Instance Methods

Category: Appointment Book

appDayNo:timeSlotNo:physicianNo:

This method returns the object reference to a specified physician time slot object for a specified time slot object on a specified day object.

createAppointmentBook

This method creates an appointment book using a series of nested object lists. Three object lists are used to create the appointment book: day list, time slot list, and physician appointment list. The day list is an object list that contains information on each day for a user defined number of days (constant *LAST_RUN_DAY* in the constants panel). Each day object contains 60 time slot object lists (two time slot object lists for every fifteen minutes between 9:00 AM to 4:15 p.m). After excluding the 15 minute time slots during lunch (11:30 a.m to 12:45 PM), 43 time slots are available for patient appointments. The remaining 17 time slots are defined with the string *apptUnavailable* and are unavailable for appointments. Each time slot object contains a physician appointment object list with the number of objects in each physician appointment object list equal to twice the number of physician appointments in the clinic. Half of the objects in each physician appointment object list are reserved solely for clinic appointments. Additionally, starting with time slot object two, every other physician appointment object list has all objects reserved for clinic appointments defined as *Unavailable*. This results in each physician having two available physician time slots and one available clinic time slot every fifteen minutes during office hours, excluding lunch. Furthermore, for every eighth time slot object list (one time slot object list per hour), all physician appointment objects are defined with the string *Walk-in Slot* to allow walk-in patients to enter the clinic. At the beginning of the model execution, all physician appointment list objects are initially set to *nil* or *Unavailable* or *Walk-in Slot*, depending on the particular time slot. See Chapter 4.4.1.2 for further details.

setApptOnDayNo;timeSlotNo:physicianNo:

This method sets the value *apptScheduled* to a specified physician time slot for a specified time slot on a specified day.

setLateApptTakenOnDayNo:timeSlotNo:physicianNo:

This method sets the value *lateAppointmentTaken* to a specified physician time slot for a specified time slot on a specified day.

setLateArrivalOnDayNo:timeSlotNo:physicianNo:

This method sets the value *lateAppointment* to a specified physician time slot for a specified time slot on a specified day.

setNoShowOnDayNo:timeSlotNo:physicianNo:

This method sets the value *noShowScheduled* to a specified physician time slot for a specified time slot on a specified day.

setWalkInFilledOnDayNo:timeSlotNo:physicianNo:

This method sets the value *walkinFilled* to a specified physician time slot for a specified time slot on a specified day.

Category: Appointment Scheduling

calculateRandomSlotNumber

This method randomly selects one of the 43 potential time slots to start (initialize) searching for an appointment. Each time slot has a probability of 1/43 of being chosen.

findSlotForWalkIn:afterTime:

This method attempts to locate an open time slot for a walk-in patient. If the walk-in patient requires a physician, all physician time slots are checked until the end of the day for one that is available. Similarly, if the walk-in patient does not need a physician, all slots (including both physician and clinic time slots) are checked for one that is available. If an available time slot is found, the walk-in patient is scheduled for that time slot and the slot is set to *walkInFilled*.

scheduleApptTimeFor:

This method uses the patient's time slot scheduling rule to locate an available time slot, using either the specific appointment day scheduling algorithm or the sequential appointment day scheduling algorithm. The starting search day is also calculated. Upon finding an available time slot, this method returns the time slot number to the operator.

scheduleApptSpecificallyFor:onDayNo:inSlot:

This method uses the specific appointment day scheduling algorithm to schedule a patient who is looking for a specific time slot on a specific day. The method first adds the scheduling lead time (days) to the current day (day the phone call requesting an appointment occurred) to arrive at the requested day. If the two time slots for the requested appointment time are unavailable, then the day after the requested day is

searched. On failing to find an available time slot, the algorithm continues its search on the requested day minus one day, requested day plus seven, requested day plus two, requested day plus three, requested day plus four, and so on, until an available time slot is found. See Chapter 4.4.2.4 for more details.

scheduleApptSequentiallyFor:onDayNo:

The Sequential Appointment Day Search algorithm is used when a patient is using one of the sequential time slot scheduling rules (rules AM, PM, or ALL). Patients using the AM time slot scheduling rule start their search for an appointment starting at 9 AM and searches forward for an available appointment until 11:15 AM Patients using the PM time slot scheduling rule start their sequential search at 1 PM through 4:15 PM Forty percent of the patients using the ALL time slot scheduling rule start at 9 AM and searches forward for an available appointment until no more appointments can be searched for that day. The remaining sixty percent of the patients using the ALL time slot scheduling rule start their sequential search at 1 p.m through 4:15 PM, and if an appointment cannot be found, sequentially searches from 9 AM until 11:15 AM Like the **scheduleApptSpecificallyFor:onDayNo:inSlot:** method, it arrives at the requested day in a similar manner, and upon failing to find a time slot using the appropriate time slot scheduling rule, searches the requested day plus one and requested day minus one as before. Unlike the **scheduleApptSpecificallyFor:onDayNo:inSlot:** method, it searches the requested day plus two, requested day plus three, and so on. See Chapter 4.4.2.4 for more details.

scheduleNewPatientSequentiallyFor:onDayNo:

Unlike regular patients, new patients require two consecutive time slots. This method searches in a similar manner to the **scheduleApptSequentiallyFor:onDayNo:** method but for two consecutive time slots.

scheduleNewPatientSpecificallyFor:onDayNo:inSlot:

This method searches for two specific consecutive time slots for a new patient using a search method similar to **scheduleApptSpecificallyFor:onDayNo:inSlot:**.

Category: Display

doctorIsAvailable

This method increases the number of available physicians by one. This value is displayed in a counter over the physician office area object.

doctorIsUnavailable

This method decreases the number of available physicians by one. This value is displayed in a counter over the physician office area object.

updateDateDisplay

This method displays the current simulation day number and day of the week above the clinic object.

updateTimeDisplay

This method displays the approximate time (*A.M.* or *P.M.*) at the top of the clinic object. This is only an approximate time sinve the time is only updated each time a patient enters the clinic. For example, if a patient enters the clinic at 11:59.59 AM, the time will display *A.M.* and remain at this value until the next patient enters, which is likely to be at some time after 12 noon. At that time, this method is called and the display is updated to *P.M.*

Category: Notification

companionAscended:

This method sends the companion object to their next location based upon their previous location and final destination. For example if a companion object ascends from the entrance object, the next location is the first available seat.

patientAscended:

This method sends the patient object to their next location based upon their previous location and final destination. For example if a patient object ascends from the entrance object, the next location is a registration window, if one is available, otherwise it is to an available seat.

staffMemberAscended:

This method sends a medical staff member object (e.g., medical assistant, nurse, physican assistant, physician) to their next destination based upon their current location (object it is in) and the variable *destination* (variable held by any dynamic object that indicates to which object the dynamic object will move to next). For example if a nurse object (with the variable *destination* set to the medical staff office area object) ascends from the rooms area object, the next location the nurse will move to is the medical staff office area object.

dynObjArrived:

This method destroys any phone calls with the variable *thinned* equal to *True*. It also routes arriving patients (patients that has been created and enter the clinic object) to the entrance object. Note that the "Entrance" object resides inside the "Dallas, Texas Clinic" object (i.e., the entrance object is a deeper static object of the clinic deep static object) but the entrance class is not a subclass of the clinic class.

dynObjAscended:

If a patient object ascends from the clinic object, the method **patientAscended:** is called, but if a companion ascends from the clinic object,the method **companionAscended:** is called. Finally, if a medical staff member ascends from the clinic object, the method **staffMemberAscended:** is called.

dynObjEnteredModel:

This method creates new phone calls. Phone calls are generated with a minimum mean interarrival time of 650 seconds and are thinned with a probability of rejection determined by divided 650 seconds by the current time's mean interarrival time (*meanPhoneCallInterarrivalTime)*. If rejected, the phone call's *thinned* variable is set to *True*.

Category: Reset Call Rates

changeToAfternoonRate
changeToEveningRate
changeToMorningRate
changeToNightRate
changeToWeekendDayRate
changeToWeekendEveningRate
changeToWeekendNightRate

These methods reset the *meanPhoneCallInterarrivalTime* variable to the current time period's phone call interarrival rate (set in the constants panel).

Category: Start Up and Shut Down

replicationStarted

This method initializes values and creates the registration windows objects and the initial phone call object from the clinic object to the Queston Information Center object.

Category: Statistics

setFlagsAndCreateOutputFiles createClinicOutputData

This method initiates all output measure collection efforts by the different objects collecting output measures. At approximately midnight, this method notifies the clinic object, the entrance object, the internal waiting area object, the rooms area object, the Queston Information Center object, the medical staff office area object, each operator object, each seat object, each physician office object, each registration window object, each room object, and each medical staff member desk object to collect the data from the previous day and copy the data to its respective output file.

calculateMaxNosInSeat

resetAndPrintEachDaysData

recalculateDistribution

resetAndPrintClinicsDayEvent:

These methods create an output file and collect output measures, such as the distribution times of the seats being utilized. All of the output measures collected by this method are presented in Appendix B.

Class Methods

Category: StartUp

replicationStarted

This method creates five random variate streams:

1) A IID U(0,1) random variate stream.
2) A random physician number generated U(0, *NO_OF_PHYSICIANS*) and rounded up to the nearest integer.
3) A random time slot number generated IID U(0,1).
4) A random arrivial deviation time (i.e., time between when the patient was scheduled to arrive and the patient's actual time) generated Normal (0,250).
5) The phone call interarrival times are a IID exponential random variates with mean *minimumInterarrivalTime*. The *minimumInterarrivalTime* is the minimum interarrival mean of all the time periods (morning, afternoon, evening, night, weekend day, weekend evening, weekend night). The method of thinning rejects

phone calls with a probability of (*minimumInterarrivalTime*)/(*meanPhoneCallInterarrivalTime*). Using this procedure, more calls are accepted during the time periods when the interarrival mean is lower (call rates are higher).

CLASS: MedicalStaffArea

Inherits from: VSObject : VSShape : VSObject

Class Description

This class creates and defines the medical staff member, their desks, and their chairs. It also defines the staffing hierarchy that allows a higher skilled available medical staff member to treat a patient.

Instance Methods

Category: Notification

dynObjArrived:

This method moves the medical staff member objects to the appropriate move spot object (objects in the medical staff office area that are invisible and used to guide the objects in the medical staff office area object) in the medical staff office area based on the medical staff member's final destination.

medicalAssistantIsAvailable:
nurseIsAvailable
physAssistantNursepracIsAvailable:

These methods add an available medical assistant, nurse, or a physician assistant to the respective availability list. In addition, the methods update the number of each medical staff member in each list and displays this number on top of the medical staff office area object.

patientNeedsServiceOf:at:

This method sends the appropriate medical staff member to the room from which the message was received. It scans the specific staff member list for the first available staff member (the staff member with the longest time on the list) and sends this object to the requesting room. This method also allows a higher skilled available staff member to treat a patient whose needed medical staff member is unavailable.

Category: Reference

deskChairList

medicalStaffAreaMoveSpotList

These methods return an object reference to the deskChairList object and to the medicalStaffAreaMoveSpotList object, respectively.

Category: Start Up and Shut Down

replicationStarted

This method initializes and creates the medical staff member (excluding the physician) and their desks, chairs, and move spots. The number of each medical staff member created is determined by the value set in the constants panel. The number of medical assistants, nurses, and physican assistants created are determined by the value of the contants NO_OF_MEDICAL_ASSISTANTS, NO_OF_NURSES, and NO_PA_NPS, respectively. For each desk, a move spot is created to allow medical staff members to move from one medical staff office area to another.

CLASS: PhysicianOfficeArea

Inherits from: VSObject : VSShape : VSObject

Class Description

Similar to the MedicalStaffArea class, this class creates and defines the physician offices objects and the physician object.

Instance Methods

Category: Acess

listOfPhysicianOffices

This method returns the list reference of the available physicians list.

Category: Notification

dynObjArrived:

This method routes the physician object to its own office when arriving to the physician office area.

dynObjAscended:

This method moves the physician to the phyician office area exit when the physician comes out of his/her office.

patientNeedsPhysician:at:

This method looks for an available physician and sends the available physician object to the examination room which is sending the message.

Category: Start Up and Shut Down

replicationStarted

This method creates the physician offices and the physicians within the offices. The number of offices created is determined by the value of NO_OF_PHYSICIANS set in the constants panel.

CLASS: RoomsArea

Inherits from: VSObject : VSShape : VSObject

Class Description

The class defines and creates the check-in rooms, specialty rooms, and the examination rooms. Additionally, it routes patients and staff inside the rooms area object.

Instance Methods

Category: Access

listOfCheckInRooms
listOfSpecialtyRooms
listOfExamRooms

These methods return an object reference to the listOfSpecialtyRooms object and to the listOfExamRooms object..

Category: Availability

availableCheckInRoom
availableSpecialtyRoom
availableExamRoom

These methods check the respective room list for the first available room and returns an object reference to the available room. For example, availableCheckInRoom searches all check-in rooms and upon finding an available check-in room, returns the object reference to this room. If there are no available rooms, the value of *nil* is returned.

Category: Notification

dynObjAscended:

This method sends another message based upon the class of the arriving object. For example if the arriving object is a Patient class, then a message **patientAscended:** is sent to itself.

patientAscended:

This method sets the patient's destination based upon the patient's *status* variable and moves this patient to the move spot outside of the room that they exited.

staffmemberAscended:

This method sends the medical staff member to the move spot outside of the room they exited, independent of their next destination.

Category: Start Up and Shut Down

replicationStarted

This method initializes and creates the check-in, specialty, and examination rooms. The number of each type of room is determined by the value set in the constants panel. The number of check-in rooms, specialty rooms, and examination rooms are determined by the value of the contants NUMBER_OF_CHECKIN_ROOMS, NUMBER_OF_SPECIALTY_ROOMS, and NUMBER_OF_EXAM_ROOMS, respectively.

Category: Statistics

createRoomsAreaOutputData

recalculateDistribution:for:

resetAndPrintDayEvent:

These methods create an output file and collect output measures, such as the distribution of time each check-in, examination, or specialty room is busy. All of the output measures collected by this method are presented in Appendix B.

CLASS: WaitingRoomBuffer

Inherits from: VSObject : VSShape : VSObject

Class Description

The class defines the waiting room buffer object. Additionally, this class processes patients and companions waiting in the waiting room buffer object.

Instance Methods

Category: Notification

checkInIsAvailable:

This method is invoked when a check-in room becomes available and there are no patients in a seat waiting for check-in. The method checks to see if any patients in the waiting room buffer need to check-in, and if a patient is found who requires a check-in, they are sent from the waiting room buffer object with their *destination* variable set to the check-in room object that sent this message.

dynObjArrived:

This method increases the number of objects in the waiting room buffer by one and displays this on top of the waiting room buffer object.

dynObjDeparted:

This method decreases the number of objects in the waiting room buffer by one and displays this on top of the waiting room buffer object.

patientIsLeavingClinic:

This method checks all the objects in the waiting room buffer to see if any companions belong to the patient who is leaving the clinic, and if a companion object is found, they are sent to the exit with the companion's *destination* variable set to the entrance object.

registrationIsAvailable:

This method is invoked when a registration window becomes available and there are no patients in a seat waiting to register. The method checks to see if any patients in the waiting room buffer need to register, and if a patient is found who needs to

register, they are sent from the waiting room buffer object with their *destination* variable set to the registration window object that sent this message.

seatIsAvailable:

This method is invoked when a person leaves a seat. It pulls the first person from the waiting room list and sends them to the unoccupied seat. Since the list is filled sequentially, the person waiting the longest length of time in the waiting room buffer is sent to the seat.

Category: Start Up and Shut Down

replicationStarted

This methods initializes the waiting room list.

Category: Statistics

createWaitingRoomBufferOutputData

resetAndPrintDayEvent:

These methods create an output file and collect output measures on the waiting room buffer, such as the maximum number of people in the waiting room buffer per day. All of the output measures collected by this method are presented in Appendix B.

CLASS: InternalWaitingArea

Inherits from: VSObject : VSShape : VSObject

Class Description

Similar to the WaitingRoomBuffer class, the class defines the internal waiting area object. Additionally, the class processes the patients waiting in the waiting room buffer object.

Instance Methods

Category: Notification

dynObjArrived:

This method is invoked when a patient arrives in the internal waiting area. First, the number of patients waiting in the internal waiting area is increased by one and

displayed on top of the internal waiting area object. It then checks to see if any examination or specialty room became available while the patient was in transit to the internal waiting area. If the requested room is available, the patient's destination variable is set to the available room and sent out of the internal waiting area.

dynObjDeparted:

This method decreases the number of patients waiting in the internal waiting area by one and displays this value on top of the internal waiting area object.

examRoomIsAvailable:

This method is called by an examination room that has become available. It sends the patient who has been waiting the longest for an examination room to the next available examination room.

specialtyRoomIsAvailable

This method is called by a specialty room that has become available. It sends the patient who has been waiting the longest for a specialty room to the next available specialty room.

Category: Start Up and Shut Down

replicationStarted

This method

Category: Statistics

createInternalAreaOutputData

recalculateDistribution

resetAndPrintDayEvent:

These methods create an output file and collect output measures, such as the maximum number of patients waiting in the internal waiting area per day. All of the output measures collected by this method are presented in Appendix B.

CLASS: MoveSpot

Inherits from: VSObject : VSShape : VSObject

Class Description

This class defines the eight shallow objects (called move spots in the clinic object) that direct patients, companions and medical staff member members to their appropriate destination.

Instance Methods

Category: Notification

dynObjArrived:

This method sends people to the appropriate location when they arrive in a move spot. Each of the eight move spots determine where the person needs to go and directs them to either the next move spot, or to their final or intermediate destination. For example if the physician's final destination is a physician office, the intermediate destination need to be visited first, which is the physiciaian office area.

CLASS: Parts

Inherits from: VSObject : VSShape : VSObject

Class Description

The class defines several methods that can be used by this class's subclass.

Instance Methods

Category: Access

isAvailable

This method returns the boolean of *availability.*

isUnavailable

This method returns the opposite boolean value of *availability*. For example if availability has a value of *True* than this method return *False*.

Category: Settings

setAvailabilityTo:

This methods sets the value of the variable *availability.*

CLASS: Desks

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

At model execution, this class intializes the desk objects.

Instance Methods

Category: Start Up

replicationStarted

This method sets all desk object's *availability* to *True*. Note that the desks are shallow objects.

CLASS: DeskChair

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

This class defines the deskchair objects.

Instance Methods

Category: Access

deskChairNumber

This method returns the integer value of the desk chair number.

Category: Notification

dyObj:stoppedEngagement:

This method informs the medical staff member area that a medical staff member has stopped taking notes and adds the medical staff member to the available list. This method also informs the check-in and examination rooms that the medical staff member is available to service the next patient.

dynObjArrived:

This method engages the medical staff member in note taking when the medical staff member returns to their desk.

Category: Setting

setStaffTypeOccupyingTo:

This method sets the person created in the deskchair object to a given medical staff member. This method is executed only at the beginning of the simulation run.

Category: Start Up and Shut Down

replicationStarted

This methods initializes all medical staff member's *availability* to *True*.

createMedAsstOutputData
createNurseOutputData
createPhysAsstOutputData
resetAndPrintMedAsstDayEvent:
resetAndPrintNurseDayEvent:
resetAndPrintPhysAsstDayEvent:

These methods create output files and collect output measures for each medical staff member, such as the medical staff's utilization rate per day. All of the output measures collected by this method are presented in Appendix B.

Class Methods

Category: StartUp

replicationStarted

This class method generates a random note taking time using IID exponential variates with mean *MEAN_NOTE_TAKING_TIME*. The value of the *MEAN_NOTE_TAKING_TIME* is set in the constants panel.

CLASS: MedicalStaffMoveSpot

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

This class defines the shallow objects (called moveSpots) in the medical staff office area object that directs medical staff members to and from their desk chair objects.

Instance Methods

Category: Access

dynObjArrived:

This method sends medical staff members to the appropriate location or moveSpots. The moveSpots are created in the MedicalStaffArea object.

CLASS: PhysicianOffice

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

This class defines the number of new patients each physician can schedule per day. Furthermore, this class defines when the physician can take their lunch break.

Instance Methods

Category: Notification

dynObj:stoppedEngagement

This method informs the check-in and examination rooms that the physician is available. It also sends a message to the clinic level that the physician is available, which updates the physician availability display. Additionally, if the physician completes taking notes between 11:30 AM and 1 PM and all morning appointments requiring his services have been taken care of, then they can start their lunch break.

dynObjArrived:

This method engages the physician in note taking when arriving to their office.

recordMorningArrival

This method increments the value of the variable *noOfMorningArrivalsLeftForDoc* by one. This variable counts of the number of morning patients in the clinic.

recordMorningDeparture

This method decreases the value of the variable *noOfMorningArrivalsLeftForDoc* by one. Furthermore, if the last morning patient leaves the clinic and the time is between 11:30 AM and 1 PM, the physician then takes a lunch break.

tellDoctorThatItsLunchTime

This method is invoked every day at 11:30 AM If the physician is idle in the office and has no more morning appointments to meet, then the method tells the physician to go to lunch.

Class Methods

Category: StartUp

replicationStarted

This class method creates two random variate streams. One is a random note taking time for the physician, generated using an exponential distribution with a mean of *MEAN_NOT_TAKING_TIME* and the other is a random maximum number of new patient for each physician for the day. This value is generated using a U[0,4] and rounding to the nearest integer.

CLASS: RegistrationCheckOut

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

This class defines the registration window object. It defines how the registration window object handles patients for both the registration and the check-out.

Instance Methods

Category: Notification

dynObj:stoppedEngagement:

This method sends the patient who has just finished registering or checking-out to the next appropriate location. If the patient finished registration, then the next location is an available check-in room. If a check-in room is unavailable, then the patient takes a seat, if one is available; otherwise, the patient object is sent to the waiting room buffer. If the patient finished a check-out, then the next location is the clinic exit (entrance object).

dynObjArrived:

This method engages the person in either a registration or a check-out at the registration window. The registration window's *availability* is set to *False*.

dynObjDeparted:

When the patient departs the registration window, this method sets the *availability* to *True* and sends a message to all the seat objects that the registration window is available.

CLASS: Room

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

The class defines the functionality of the room objects for all room types (check-in, examination, specialty). It identifies the needs of each patient and requests services of medical staff members.

Instance Methods

Category: Notification

dynObj:stoppedEngagement:

This method sends another message depending on the class of the person who has stopped an activity. If the object is a patient class, then the **patient:stoppedEngagement:** method is sent to the patient, otherwise, the **technician:stoppedEngagement:** method is sent to the technician object in the specialty room.

dynObjArrived:

This method identifies what class of person arrives and performs different operations based upon their class. For example, if the class of person is a patient, this method checks to see what process (e.g., check-in, examination, post-examination) the patient and medical staff member requires, sends a message to the appropriate object requesting assistance, and changes the picture on the room object to the image of the patient object. If the class of person arriving is a medical staff member, then a message is sent to the medical staff member to engage the patient in the required service and changes the display to show the image of "Check-in Busy Image" if the room is a check-in room, "Technician Busy Image" if the room is a specialty room, 'PA NP Examination Image" if the service is provided by a physician assistant, "Nurse Examination Image" if the service is provided by a nurse, "Medical Assistant

Examination Image" if the service is provided by a medical assistant, or a "Physician Examination Image" if the service is provided by a physician.

dynObjDeparted:

If the person departing is a patient class, then this method resets the room object's *availability* to *True*, sends a message to the internal waiting area object that the room is available, and changes the display to "CheckIn Room Image" if the room is a check-in room, "Specialty Room Image" if the room is a specialty room, or "Examination Room Image" if the room is an examination room. Otherwise, the departing object is a medical staff member and the method does nothing.

medicalStaffMemberIsAvailable

This method tells the medical staff office area object that a patient needs the service of a particular medical staff member.

patient:stoppedEngagement:

Upon completion of a service, this method checks the patient's *status* to check what process (e.g., check-in, examination) the patient needs to undergo next, resets the patient's *destination* to the next location depending on the patient's next process, and sends the patient to the room exit. Though the patient may need to undergo the next process in an examination room, all patients must exit the room, move to another available examination room, or to the internal waiting area if there is no available examination room.

physicianIsAvailable

This method tells the physician office area object that a patient needs the service of a particular physician.

technician:stoppedEngagement:

This methods is invoked when the technician finishes taking notes. This method resets the technician's *availability* to *True*.

Category: Start Up and Shut Down

replicationStarted

This method initializes all the rooms at model start up.

Class Methods

Category: StartUp

replicationStarted

The method generates random note-taking time as IID exponential random variates with mean *MEAN_NOT_TAKING_TIME*.

CLASS: RoomMoveSpot

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

This class defines the shallow objects (called roomMoveSpot) in the rooms area object that directs patients and medical staff members to the appropriate room.

Instance Methods

Category: Notification

dynObjArrived

This method sends both patients and medical staff member to the appropriate location or roomMoveSpot object. The roomMoveSpot object are created in the RoomsArea class.

CLASS: Seat

Inherits from: Parts : VSObject : VSShape : VSObject

Class Description

This class defines the seat objects and its functionalities.

Instance Methods

Category: Notification

dyObjArrived:

This method checks to see if either a check-in room or a registration room (depending on the patient's need) became available for a patient while enroute to the seat.

dynObjDeparted:

This method resets the seat object's values when a person leaves. The seat's *availability* is set to *True*.

serviceIsAvailable

This method is invoked when a registration or a check-in room becomes available. It set's the patient's destination and sends them to the correct move spot.

CLASS: PhoneCall

Inherits from: VSDynamicObject : VSObject : VSShape : VSObject

Class Description

The class defines the functionality of the phone call object. Note that this class and all the following classes are all subclasses of the VSDynamicObject class and inherits all the methods associated with the VSDynamicObject class, including the methods **moveTo:takingTime:** and **moveTo:usingPath:**.

Instance Methods

Category: Notification

enteredModelAtObject

This method changes the default image of the phone call object to the "Phone Call Red Cross" image.

Category: Settings

setDestinationTo:
setThinnedTo:

These methods set the variables *destination* and *thinned* to a given value.

CLASS: Person

Inherits from: VSDynamicObject: VSObject : VSShape : VSObject

Class Description

The class defines several methods (**destination**, **destinationNumber**, **setDestinationNumberTo:**, and **setDestinationTo:**) that will be used frequently by this class's subclasses.

Instance Methods

Category: Access

destination

This method returns the object reference of the object's *destination* variable. The variable destination is assigned to all dynamic object that indicates to which object the dynamic object will move to next.

destinationNumber

This method returns an integer value of the object's *destinationNumber* variable. The destinationNumber is an integer value assigned to a group of similar objects. For example, if a physician is going to the physician office area, the destinationNumber will be set to the physician office number corresponding to his physician number. This will insure that the physician will go to the correct office.

Category: Settings

setDestinationNumberTo:

This method sets the object's *destinationNumber* variable to the given value.

setDestinationTo:

This method sets the object's *destination* variable to the given value.

CLASS: Companion

Inherits from: Person : VSDynamicObject: VSObject : VSShape : VSObject

Class Description

This class defines the functionalities of the companion object.

Instance Methods

Category: Access

companionNumber

This method returns an integer value of the companion object's *companionNumber* variable.

Category: Notification

enteredModelAtObject

This method changes the default image of the companion object to the "Companion Image" image.

CLASS: Patient

Inherits from: Person : VSDynamicObject: VSObject : VSShape : VSObject

Class Description

This class defines and creates the patient object. It assigns values to all the variables associated with each type of patient category, including walk-in patients.

Instance Methods

Category: Access

actualArrivalTime
checkInMedStaffMember
checkInTime
checkOutTime
doesntNeedMedStaffAtCheckIn
examinationMedStaffMember
examinationTime
exitInterviewMedStaffMember
exitInterviewTime
hasCompanions
isAMorningArrival
isANoShow
isAPreVisit
leadTime
needSpecialPostExam
needSpecialPreExam

numberOfCcompanions
patientArrivalType
patientCategory
patientNumber
physicianRequired
postExamMedStaffMember
postExamTime
preExamMedStaffMember
preExamTime
preVisitRequired
registrationTime
schedulingRule
status
waitStartTime
walkInFitInTime

For each of these methods the value of the variable with the same name as the method's name is returned to the object requesting the information. For example, the method **physicianRequired** returns an integer value of the patient object's *physiicanRequired* variable and the **walkInFitInTime** returns a real number value of the patient object's *walkInFitInTime* variable.

Category: Data Access and Setting for Objects

setInternalAreaStartTimeTo:
setSeatStartTimeTo:
setWaitingRoomBufferStartTimeTo:

These methods set the patient's variable indicated in the method's name to a given value. For example, the method **setSeatStartTimeTo:** sets the patient's *seatStartTime* variable to the given value.

setInternalAreaWaitingTime
setSeatWaitingTime
setWaitingRoomBufferWaitingTime

These methods calculate the total time spent waiting by the patients in the internal waiting area (**setInternalAreaWaitingTime**), the seat (**setSeatWaitingTime**), and the waiting room buffer (**setWaitingRoomBufferWaitingTime**).

checkInRoomWaitingTime
examRoomWaitingTime
internalWaitingTime
seatWaitingTime
waitingRoomBufferWaitingTime

These methods return the value of the patient's variable denotes by the method's name. For example, the method **seatWaitingTime** returns the real number value of the patient's *seatWaitingTime* variable.

Category: Initialization and Category Data

initializeItselfAsAWalkIn

This method is invoked from the dynObj:stoppedEngagement: in the Operator class and from the dynObjEnteredModel: in the Entrance class. It determines the category for the patient entering the model and sends a message to the patient to set their input data. Walk-ins, like regular patients are created as one of eight possible categories.

setCategory1Data
setCategory2Data
setCategory3Data
setCategory4Data
setCategory4PreVisitData
setCategory5Data
setCategory5PreVisitData
setCategory6Data
setCategory7Data
setCategory8Data

Each method sets all the values for the patient's variables that are needed for the clinic visit, according to the respective patient category. This method determines what kind of visit (pre-visit or a regular visit) the patient will have. It determines whether the patient will have certain types of services (e.g., check-in, examination, post-examination) and which medical staff member will deliver each such service. Futhermore, the length of the service, the time slot scheduling rule, and the scheduling lead time will be determined.

Category: Notification

moveToNextPlace

This methods moves the patient to the next object depending on where the patient is currently and the value of the patient's *destination* variable.

Category: Record Times

recordActualArrivalTime:

recordScheduledApptTime:

These methods record the simulated time in the model when the patient arrives to the clinic for an appointment in the variable *actualArrivalTime* and the simulated time when the patient is scheduled to arrive for the appointment in the variable *scheduledApptTime*, respectively.

Category: Settings

setHadAPreVistTo:

setIsAMorningArrivalTo:

setIsAPreVisitTo:

setNewNormalVisitLeadTimeTo:

setNumberOfCompanionsTo:

setPatientArrivalTypeTo:

setPatientNumberTo:

setPhysicianRequiredTo:

setStatusTo:

setWaitStartTimeTo:

setWalkInFitInTimeTo:

These methods set the patient's variable indicated in the method's name to a given value. For example, the method **setPatientNumberTo:** sets the patient's *patientNumber* variable to the given value.

Class Methods

Category: StartUp

replicationStarted

This method creates all the random variate streams necessary to create random service times for each patient category with each medical staff member, using a triangular distribution with the mean, maximum, and the mode provided in the Revised Patient Category Input Data in Appendix A.2. Similarly, the random variate streams to generate random lead times for each patient category are created using a triangular

distribution with the mean, maximum, and the mode provided in the Revised Patient Category Input Data in Appendix A.2.

CLASS: MedicalStaffMember

Inherits from: Person : VSDynamicObject: VSObject : VSShape : VSObject

Class Description

This class defines the medical staff member, including the technician object, the medical assistant object, the nurse object, the physician assistant object, and the physician object. It defines several methods, such as **deskNumber** and **noteTakingTime,** that will be used frequently by its subclasses. Each medical staff member will have a variable *noteTakingTime* that represents the note-taking time after a patient encounter.

Instance Methods

Category: Access

deskNumber

This method returns the *deskNumber* variable of the medical staff member.

noteTakingTime

This method returns the *noteTakingTime* variable of the medical staff member.

Category: Setting

setDeskNumberTo:
setNoteTakingTimeTo:

These methods set the object's *destinationNumber* and *noteTakingTime* variables, respectively, to the given value.

CLASS: Technician

Inherits from: MedicalStaffMember : Person : VSDynamicObject: VSObject

Class Description

This class creates and initializes the technician object.

Instance Methods

Category: Notification

enteredModelAtObject

This method resets the default image of the technician object to the “Technician Image”.

CLASS: MedicalAssistant

Inherits from: MedicalStaffMember : Person : VSDynamicObject: VSObject

Class Description

This class creates and initializes the medical assistant object.

Instance Methods

Category: Notification

enteredModelAtObject

This method resets the default image of the medical assistant object to the “Medical Assistant Image”.

CLASS: Nurse

Inherits from: MedicalStaffMember : Person : VSDynamicObject: VSObject

Class Description

This class creates and initializes the nurse object.

Instance Methods

Category: Notification

enteredModelAtObject

This method resets the default image of the nurse object to the “Nurse Image”.

CLASS: PhysAssistantNursePrac

Inherits from: MedicalStaffMember : Person : VSDynamicObject: VSObject

Class Description

This class creates and initializes the physician assistant object.

Instance Methods

Category: Notification

enteredModelAtObject

This method resets the default image of the physicianAssistant object to the "PA NP Image".

CLASS: Physician

Inherits from: MedicalStaffMember : Person : VSDynamicObject: VSObject

Class Description

This class creates and initializes the physician object.

Instance Methods

Category: Notification

enteredModelAtObject

This method resets the default image of the physician object to the "Physician Image".

Category: Setting

setAvailabilityTo:

setPhysicianNumber:

setStatusTo:

These methods sets the object's *availability*, *physicianNumber*, and *status* variables, respectively, to a given value.

Chapter 6 Verification, Validation, and Testing

Simulation model verification, validation, and testing (VV&T) play an integral role in the success of any simulation project. The simulation model must provide an accurate (i.e., *valid*) representation of the system it is modeling, since the decisions reached using the simulation model can determine the success or failure of the project. If the model is valid, then the decisions made using the simulation model would be similar to decisions made using the real system (if it exist). The Queston simulation model life-cycle uses numerous VV&T techniques to ensure accuracy and increase confidence in the results. This chapter discusses the importance of VV&T, presents a brief description of the VV&T techniques applied during the design and construction of the Queston simulation model, and presents examples to illustrate how these techniques were applied to the Queston simulation model.

There are numerous reasons why VV&T is an important steps in successful simulation projects. Glasow et al. [1996] identifies several benefits in conducting a thorough VV&T study:

- increased confidence in the model
- reduced risk of recommending an incorrect decision
- increased reusability for future applications

- reduced future costs by reducing the likelihood of model redesign or reprogramming
- better analysis

However, there are limitations and drawbacks of VV&T. In particular, the additional cost and time of conducting a thorough VV&T. Some of these techniques may take an extended amount of time to correctly apply. Moreover, some of the techniques must be applied repeatedly throughout the simulation life-cycle. Additionally, VV&T cannot guarantee that the results will be correct, nor that they will be correctly analyzed and interpreted. Lastly, VV&T is truly never completed since simulation models cannot be absolutely verified (since this would require testing every possible module and decision point in the logic of the simulation model for all possible input data) or absolutely validated (which would require a matching of every possible set of input data and output measures from the actual system being modeled to those in the simulation model).

Fishman and Kiviat [1968] are credited with first using the terms model verification and model validation. The distinction between the two terms is most easily understood in terms of their focus. Model *verification* is the process of determining that a model implementation accurately represents the developer's conceptual description and specification with respect to programming. Model *validation* is the process of determining the degree to which a model is an accurate representation of the real-world system [Caughlin 1995]. The simulation model cannot achieve full credibility unless *both* model verification and model validation have been applied throughout the entire simulation model development process. The term model *testing* is used to refer to the

implementation of these verification and validation techniques to identify inaccuracies or errors in the model [Balci 1998].

Numerous techniques for simulation model VV&T have been derived from software engineering VV&T. However, there are fundamental differences between the two. Balci [1994] lists several of these differences:

1) Simulation results are obtained by multiple experiments with the simulation model (experimental models), while a computer program is executed just once.
2) Simulation output results are descriptive and must be carefully interpreted to arrive at a desired solution, while results from a computer programs are prescriptive and provides exact analytical solutions for a set of input parameters.
3) Simulation validation compares the model with the system or process being modeled, whereas computer engineering validation compares the input and the output of the program against the requirement specification determined at the onset of the software design.

The reader is referred to several comprehensive papers by Whitner and Balci [1989] and Balci [1998]. Together they present and discuss seventy-seven verification and validation techniques applicable to simulation modeling. Additionally, the Defense Modeling and Simulation Office's Verification, Validation, and Accreditation Recommended Practices Guide serves as an excellent reference for VV&T techniques [Glasow et al. 1996].

6.1 Verification Techniques

Balci [1994] defines *model verification* as substantiating that a model is transformed from one form into another form with sufficient accuracy. These forms represent the different transformations the simulation model experiences from the problem definition, to model formulation, to a conceptual model, and finally to the programmed model. The verification process determines whether or not the model was built correctly to its specifications and confirms that the model functions as it was originally conceived, specified, designed, and coded. However, verification does not determine whether the specification or the design is accurate; model *validation* determines whether the simulation model's specifications, design, and the transformations are an accurate representation of the real world system. The Queston simulation life-cycle follows some of Kleijnen's [1995] recommended guidelines for verifying a simulation model:

- uses general good programming practices such as modular programming (objects would be the modules in the Queston simulation model),
- checks for intermediate simulation outputs through tracing,
- uses animation to detect programming and conceptual errors.

Though all such techniques discussed in the following sections were applied, statistical tests were specifically not used. Statistical techniques require that the system being modeled be completely observable, i.e., that all input data required for model validation can be collected from the real system. Therefore, this technique cannot be applied, since the Queston simulation model is based on a proposed system, hence input data is unavailable.

6.1.1 Assertion Checking

Assertion checking is a verification technique that compares a model's execution against the behavior that the model is designed to capture in the real system [Balci et al. 1996]. This technique has the benefit of documenting the intentions of the modeler. A drawback to assertion checking is that these additional statements may degrade the model execution time. VSE provides the modeler with an "Include Assertion" setting that can be disabled so that assertion statements are not included in the logic that are executed and hence, enhance model execution performance [Balci et al. 1996]. This technique was used throughout the development of the Queston Simulation model. Examples that illustrate this technique are

Example 1: assert numberOfOperatorsCurrentlyBusy >=0 with msg "number of operators is less than zero"

Example 2: assert [enteredPerson patientArrivalType] is WALKIN with msg "not a walk-in";

In Example 1, when this statement is executed, the variable *numberOfOperatorsCurrentlyBusy* is assumed to be greater than zero; otherwise, the program will halt its execution with the message "number of operators is less than zero." Likewise, for the second example, the program will halt its execution and give the message "not a walk-in" if the entering person is not a walk-in patient.

6.1.2 Debugging

Debugging is an iterative process of revealing errors in a computer program or a simulation model, identifying causes of the error, correcting the errors, and re-executing the program or

model until no further errors are identified. Identifying errors proved to be a challenge throughout the QPN simulation life-cycle. Four methods were used to detect errors: brute force debugging, debugging using backtracking, cause elimination, and visualization/animation [Beizer 1990].

Brute force debugging was frequently used when errors could not be determined by initial inspection. This simple method required the use of numerous print statements to identify the error (or at least the location of the error) from the print statements. Examples of such print statements are provided below.

```
tell VSEModel to printStringToTranscript:"\n Scheduled patient randomly with rule ";
tell VSEModel to printIntegerToTranscript:schedulingRule;
tell VSEModel to printStringToTranscript:"  on Day ";
tell VSEModel to printIntegerToTranscript:dayNumber;
tell VSEModel to printStringToTranscript:"  in Slot ";
tell VSEModel to printIntegerToTranscript:k;
tell VSEModel to printStringToTranscript:"  with Physician No. ";
```

In the above statements, each command to print the value of a variable is preceded by a **printStringToTranscript:** code that identifies the value. By placing these print statements throughout the simulation model, errors can be identified by scanning through the VSE Transcript window.

Debugging using backtracking is a process of scanning through the code backward to identify sources of error. This can become very tedious and complex in object-oriented programming

where message passing is frequent between numerous objects. In the case of the Queston Simulation model, debugging using backtracking was seldom used due to the large number of interactions between the objects. One example of backtracking was when an error occurred when phone calls in the call queue remained in the queue even though an operator became available. The code was scanned from the time the particular operator object became available, to the message that was sent to the call queue object from the operator object, and finally to the message that was received by the phone call object from the call queue object. It was discovered that the call queue object did not correctly process the message from the available operator and this error was quickly corrected.

Cause elimination, requires input data to be manipulated to test a hypothesis that a particular part of the logic is flawed. An example of the use of cause elimination in the Queston simulation model is illustrated with an error occurring when a person reaches the Entrance. The walk-ins were suspected of causing this error. Therefore, a test run of the simulation model, where the input data was changed to exclude walk-in patients from the entering population, was executed. When the simulation model executed without errors, the test confirmed that there was an error associated with the processing of the walk-in patients. Constructing this type of test may require a large effort and can be time consuming, hence other methods were used whenever possible.

Visualization/animation was the most frequently used debugging method. Seeing the movement of the objects as the simulation model executed resulted in a faster identification of the source of any errors because the graphics easily distinguished obvious discrepancies

from the model's specifications. For example, a problem occurred when a number of companions would not leave the clinic even after all the patients completed their treatment and departed the clinic. This error was quickly identified by observing that a particular category of patient entering the clinic with companions did not leave the clinic with them. Other examples of errors revealed through visualization are when patients would move through the walls of the clinic and not follow the designated move spots, patients would not sit in the first available chair even though a chair was available, phone calls would not move from the call queue to an available operator, and patients would not move to an available registration window.

6.1.3 Desk Checking

Desk checking is a verification process of thoroughly examining the simulation model to ensure correctness, completeness, consistency, and unambiguity; it can be viewed as an inspection or walkthrough of the simulation model. Beizer [1990] suggests that all the following tests should be conducted as part of desk checking: syntax checking, reference checking, convention violation checking, detailed comparison to specification, reading the code, and control flow graph analysis and path sensitizing. To avoid the ineffectiveness of reviewing one's own code, the Queston Simulation model was repeatedly checked for syntax, logical flow, and arithmetic errors by several operations research analysts. This desk checking was conducted every few months during the programming stages to ensure correctness in the programming of the Queston simulation model. The process required the analysts to inspect, in the order of model execution, each object's methods, line-by-line, for syntax errors and deviation from specification until the

simulation model has been completely checked (i.e., every line of logic in the simulation model had been inspected).

6.1.4 Execution Monitoring

Execution monitoring reveals errors (flaws) in the model by examining information about activities and events that take place during the model execution [Glasow et al. 1996]. Similar to brute force debugging, execution monitoring requires the inserting of print statements in the model's logic to gather data to provide information about the model's dynamic behavior. The VSE's Transcript window (Figure 6.1) depicts sample output generated by these statements in the Queston simulation model. Problems and errors can be determined by observing the contents of the VSE Transcript window. For example, in Figure 6.1, the model has been instrumented (i.e., print statements have been inserted into the logic) to display the daily number of new patients for physician two on the Transcript window. If the Transcript window displays a day where physician two serviced greater than four new patients, then this would indicate an error in the model, since the maximum allowable number of new patients for each physician in a day can be no greater than four.

The VSE has an additional tool, called the Inspector, that provides assistance in monitoring the execution of the simulation model. The Inspector allows the modeler to display all the variables of an individual object at any given simulated time. For example, by pausing the simulation (i.e., temporarily stopping the model execution), clicking on an object (such as a patient object), and opening the Inspector, all its variables (such as its next destination,

current state, and patient category type for the patient object) are displayed. Figure 6.2 depicts some of the variables of the patient object. A complete list of variables for the patient object is presented in Chapter 5.2. This tool is useful for monitoring objects and understanding when or where an error (e.g., model stops executing, the picture of the object is incorrect, objects move to an incorrect location, no walk-ins are entering the clinic) may have occurred. For example, if an error occurs each time a category five patient requests an examination, the Inspector provides the modeler with a tool to select a category five patient by its distinct icon as it enters the clinic and monitor it throughout its entire clinic visit. At any point during the patient's visit, the value of the patient's variables can be checked against the modeler's expectation for any deviation. For example, the variable **state** of a patient object may indicate that the current state at which the patient is in, such as needing examination, needing check-in, and needing pre-examination, is not what the modeler had expected at that point in time. Further investigation can then be made into correcting the error.

6.1.5 Execution Tracing

The VSE Simulator provides a method trace window feature for model verification (Figure 6.3). The information presented in the method trace window during the model execution includes the simulation clock value, method name, line number of the method call, and a unique serial number for the object. *Execution tracing* is tracking the line-by-line execution of the simulation to determine whether the simulation's behavior is proceeding correctly. Note that watching a model execution for even one simulated day of operation of the clinic can be extremely tedious.

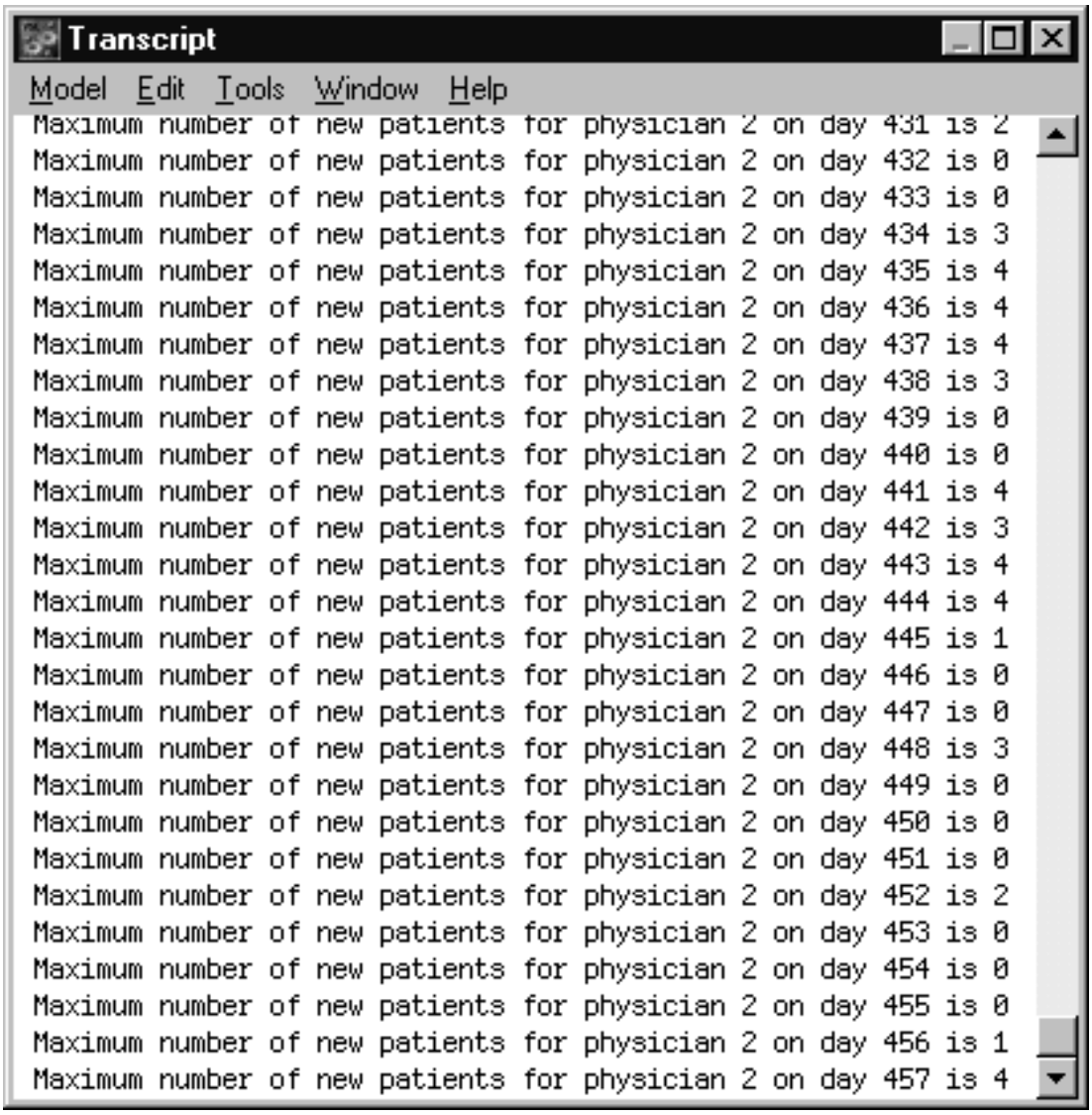


Figure 6.1: VSE Transcript Window

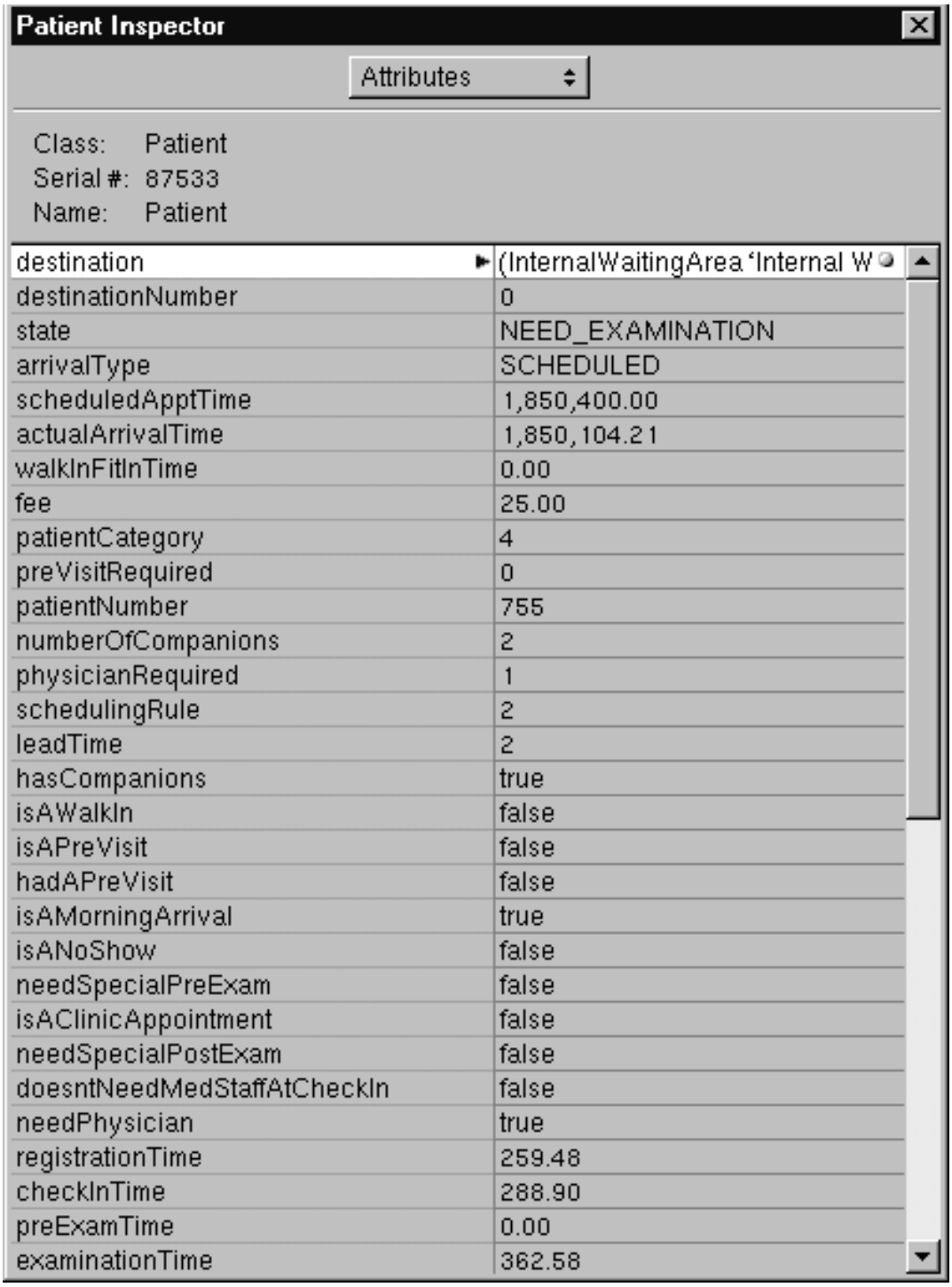


Figure 6.2: VSE Inspector

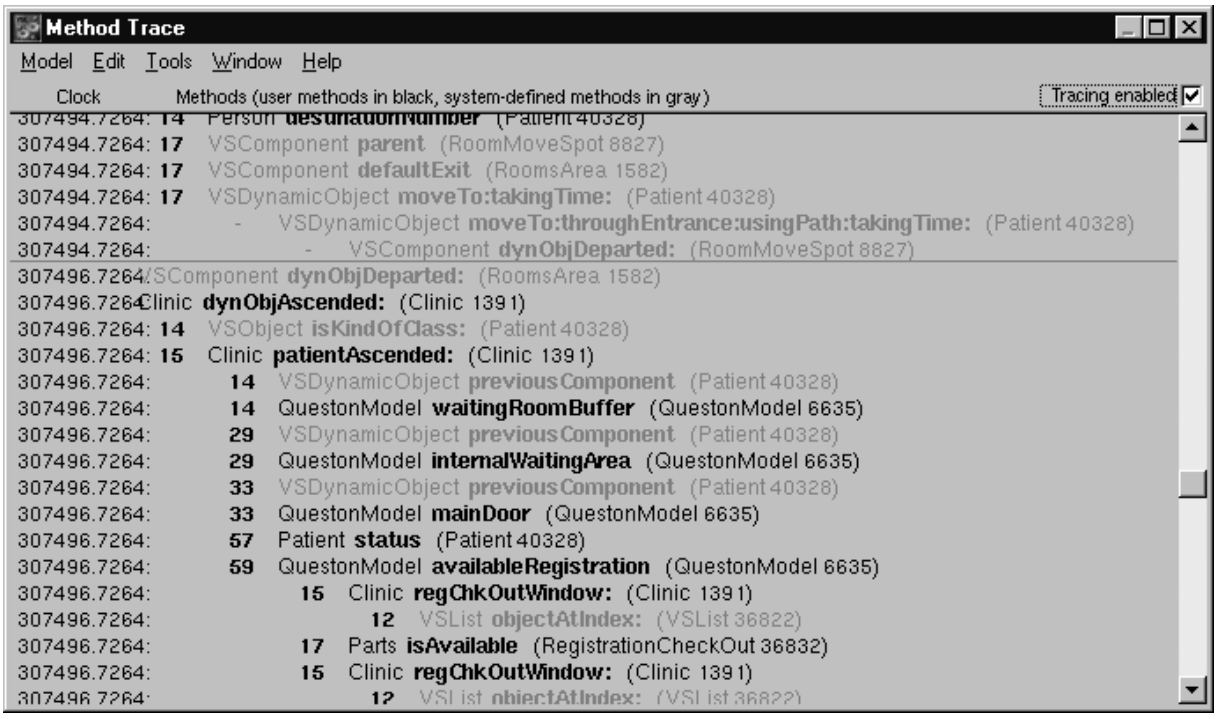


Figure 6.3: VSE Method Trace Window

Therefore, the best use of this feature is to begin watching a execution just prior to when the simulation model fails or when something in the model is seen to go against modeler's expectations. This requires re-executing the model until just prior to the simulated time of interest and then observing the method trace window. Gathering and presenting this data while the model is executing slows down the execution of the simulation model. For this reason, the tracing feature on the VSE can be enabled or disabled anytime during the model execution. If enabled, and a runtime error occurs, the method trace window shows all the methods called and by which objects up to the simulated time of the error. Each object in the simulation model is assigned a unique number that is displayed in the method trace window. This allows for tracking of specific objects simply by selecting the suspect object and using the Inspector to verify the

object number. During the QPN simulation life-cycle, this tool proved to be invaluable for identifying programming and logical errors.

Figure 6.3 depicts a sample method trace during the model execution. In this case, the model was executing without errors. The Method Trace window displays the following information [Balci et al. 1996]:

1. Simulation clock value.
2. Method name: The method names are listed with indentation to show the calling sequence. For example, Figure 6.3 shows that "isAvailable" method of class "Parts" was called at line 17 of method "availableRegistration" of class "QuestonModel" which was called at line 59 of method "patientAscended:" of class "Clinic".
3. Line number – indicates the line number of the line where the method call is made.
4. Serial Number – is a unique identification number of each object internally maintained by VSE. Serial numbers are used to distinguish objects that may have the same name.

6.1.6 Special Input Testing

Exhaustive testing of all possible input data in any complex simulation model is impractical, too expensive, and may not even be possible. *Special input testing* assesses model accuracy by subjecting the model to a variety of input data. In the Queston simulation model, this was accomplished in two ways: changing the values in the constants panel or changing the values within the logic of the appropriate methods. There are numerous types of special input tests, but two in particular were used for the Queston simulation model verification testing: extreme input testing and stress testing [Glasow et al. 1996]. Both of these tests are based on

the principle that faults, both in design and implementation, are most likely to occur and are most visible when extreme input data are used (i.e., the model is stressed).

Extreme input testing is conducted by using the minimum, the maximum, or a mix of the minimum or maximum values for the input data. In the Queston simulation model, test runs were frequently executed with both large and small staffing sizes between one and four physicians. Additional tests were conducted on the number of examination rooms and different patient mixes (Experiment XII in Chapter 7).

Similar to extreme input testing, *stress testing* is accomplished by increasing the congestion in the model to observe any exhibition of invalid behavior. A test was made on the Queston simulation model, for the cases of 6 physicians, 20 staff members, 20 examination rooms, with an associated increase in the patient population. Although the simulation took over twelve hours to execute, ten hours more than the model with two physicians, six staff members, and six examination rooms, the model performed as expected (without errors). The associated increase in the numbers of objects resulted in a slow execution time for the model since the number of interactions between the objects increased. For example, in a two examination room model, if a patient required service for an examination, the patient would look in examination room one and then look in examination room two for an available room before having to go to the internal waiting area. However in the same scenario with a ten examination room model, the patient would (potentially) have to check all ten examination rooms to find one available.

6.1.7 Visualization/Animation

Seeing objects displayed during a simulation execution is very helpful for detecting errors, such as improper object movement (or non-movement), errors in patient processing, and inaccuracies in the patient mix. For example arrival of the dynamic objects (patients and companions) and their movement through the waiting room, the check-in room, the examination room, and finally their departures can all be observed. Observations can also be made on which medical staff member treated what patients and for how long. Viewing the model during execution is very useful for uncovering errors; however, seeing is not believing in every case. Viewing an object's behavior during model execution does not guarantee model correctness, hence, visualization/animation should be used with caution.

There is an additional benefit to visualization other than just uncovering errors in a model. The graphical images of an object's movement (other then the intended behavior) can provide the modeler with immediate feedback necessary to determine *why* the model is not performing correctly. For example, if a patient concludes a registration and enters the waiting room buffer, even though chairs are available, the most likely culprit is that either the message (asking if a chair is available) did not get sent by the patient to all the chairs, or the patient did not receive the message response from the chairs.

VSE is a visual simulation language, with its strength in the visualization of the model, hence it was natural to incorporate the visualization/animation verification technique during the programming stages. After the creation of each module or methods, the first verification test

to determine if the module or method was functioning correctly was visual inspection. Visualization/animation proved invaluable for quickly and easily identifying errors and flaws.

6.2 Validation Techniques

Model validation is substantiating that the model "behaves sufficiently close to reality for the intended purposes" [Balci 1994]. Model validation addresses the credibility of the model in its depiction of the actual system; due to the approximations and assumptions in creating the model, the model and the actual system will not be identical. The question then becomes, does the model provide a practical correspondence to the actual system? If the model is not valid, then any conclusions derived from the model would be questionable [Law and Kelton 1991]. The QPN simulation life-cycle used face validation, Turing tests, extreme condition tests, and animation/operational graphics to judge whether the Queston simulation model and the proposed QPN system is sufficiently close for the purpose of this modeling endeavor.

6.2.1 Face Validation

Face validation is the process of asking domain experts or people knowledgeable in a field whether the model results and behaviors of the different objects are reasonable [Whitner and Balci 1989]. This test is most useful as a preliminary validation test in the early stages of model development. However, this test has been criticized as meaningless, since if the

output from a simulation model could be predicted *a priori*, then there would be no need to build the simulation model [Emschoff and Sisson 1970].

Face validation was conducted on the Queston simulation model once the initial model was completed. The model output measures (e.g., physician and staff utilization rates, overtime hours, and patient waiting times) were analyzed by the domain experts. They judged these output measures against what they expected to observe from their experience. The initial Queston simulation model yielded extremely long average daily clinic overtime (two to four hours of overtime per day, on average) which prompted the domain experts to question some of the input data they had provided earlier. After analyzing these outputs and examining the results of the data collected from the time measurement survey of the Christiansburg, Virginia clinic, the domain experts revised some of the patient category input parameters. These changes are described in Chapter 4.5 and the Revised Patient Category Input Data are displayed in Appendix A.2.

6.2.2 Turing Test

The *Turing Test* is based upon the ability of domain experts to differentiate between two sets of output measures, one from the model and one from the real world system (if it exists), under the same input data [Schruben 1980]. If the domain experts can differentiate the outputs, their reasons for differentiating can provide valuable information into how problems in the model representation can be corrected.

To validate the Queston simulation model, the domain experts were presented with input data (Revised Patient Category Input Data-Appendix A.2) and output measures (e.g., average daily overtime, average patient waiting time, average patient ALOS) from the Queston simulation model and asked to differentiate these with the data collected from the Christiansburg clinic. The data that was presented to them from the Christiansburg clinic were the number of patients with appointments, number of walk-in patients, overtime of the clinic, average patient waiting times, and check-in times. After much deliberation the domain experts were unable to distinguish the difference between these model output measure and those of the Christiansburg clinic.

6.2.3 Extreme condition test

Extreme condition test checks the credibility of a model output for any extreme (i.e., endpoints such as zero or the maximum allowable value) and unlikely input parameters (e.g., if no category one patients are scheduled, none should arrive and the output should reflect that) [Sargent 1998]. This test was used frequently in the QPN simulation life-cycle to validate the model logic. One such test involved varying the frequency of calls entering the Queston Information Center. With a zero call arrival rate, the model behaved correctly by not scheduling any patients to arrive at the clinic. Additionally, with the call arrival rate set to four times the typical level, patients were scheduled and arrived four times faster than normal into the clinic. Another test involving extreme condition testing varied the arrival rates of companions from between zero to four times the typical level. Once again, the model displayed the correct behavior, which further validated the model.

6.2.4 Animation/Operational Graphics

Viewing the *animation/operational graphics* of the Queston simulation model as it executes and comparing it with the operation of a real clinic can identify discrepancies between the model and the system [Balci 1994]. Animation allows the programmer and the domain experts to observe the Queston simulation model to determine if the simulation model is behaving correctly. Furthermore, the values of certain output measures (e.g., number of patients in the waiting room buffer, number of patients in the internal waiting room) graphically depicted on the screen while the simulation model is executed, can provide added credibility to the simulation model [Sargent 1998]. The Queston simulation model employs a counters to assist modelers in reaffirming model validity. Counters for the number of patients in the waiting room buffer, patients in the internal waiting area, available staff members, and available physicians are all graphically displayed, as are the day of the week, the current date, the day number and the approximate time of day. To illustrate the use of visualization, the value of the counter can be observed during the model execution for inappropriate values (negative values) or behavior (number increasing and not decreasing).

6.3 Summary

The art of simulation is best learned through experience. With each project, a modeler can become more adept at avoiding mistakes. However, the modeler cannot be certain that they

developed a valid model unless an extensive VV&T study is conducted. This is the result of restricting the boundaries of the model and making assumptions. Fortunately, there are a set of tests that can be applied to help check the validity of models. This set of tests, applied to the Queston simulation model, identified numerous errors and inconsistencies that would not have been uncovered without a thorough VV&T assessment.

Chapter 7 Preliminary Experimentation and Development

7.1 Experimentation

During the design of the Queston simulation model, a number of changes were made in the model's assumptions and input data to increase confidence in the model. Fifteen test experiments and two input data sets were used during the design and programming phase that led to the final Queston simulation model. These experiments describe the various changes to the configurations, to the input data set, and to the way patients, phone calls, and scheduling were handled. Unless otherwise stated, each successive experiment incorporated the previous changes. These experiments and changes to the model are described in Table 7.1.

7.2 Preliminary Results from Initial Baseline Model

An initial baseline model, using the configuration described in Table 7.2 and the input data in Appendix A.1, was executed, with the results presented in the Table of Experiments in Appendix D. The results from the baseline model show that average daily overtime was extremely long on Mondays (82.6 minutes) and on Fridays (95.6 minutes) while on Wednesdays there was very little overtime (12.3 minutes). Furthermore, it was observed that on average 47 patients arrived

in the morning, whereas only 19 patients arrived in the afternoon, physicians rarely had a lunch break, a maximum of 31 patients on average waited in the waiting room buffer, and the patients' ALOS were extremely long at 133.0 minutes (some patient categories such as category 2, 3, 4a, 4b, and 5 averaged almost two to three hours). These results were deemed unrealistic by the domain experts, precipitating changes to the model assumptions and the input data.

Table 7.1: Experiments

Experiment	Detailed Descriptions
Experiment I	Initial baseline model (see Table 7.2)
Experiment II	Sixty percent of the patients using sequential all day rule start their search on a particular day at 9 AM, while forty percent start their search starting at 1 PM.
Experiment III	Each physician was assigned only one clinic appointment every fifteen minute time slot instead of the two from the prior model.
Experiment IV	Only one third of each process time (e.g., check-in time, examination time) of the patients is spent in the examination room or check-in room face to face with the medical staff member. The other two thirds are assigned as note-taking time when they return back to the staff room.
Experiment V	Check-in times for each patient category types are reduced to an average of 5 minutes.
Experiment VI	Change to staff hierarchy (e.g., an available physician assistant or a nurse provides service to a patient if the required medical assistant is busy).
Experiment VII	Trial Run: Configuration: 1 Physician, 1 Nurse, 1 Physician Assistant, and 2 Medical Assistants, 1 Check-in room, 3 Examination rooms, 1 Registration Window.
Experiment VIII	Trial Run: Configuration: 4 Physicians, 4 Nurses, 4 Physician Assistants, and 4 Medical Assistants, 4 Check-in rooms, 12 Examination rooms, 3 Registration Windows.
Experiment IX	Similar to Experiment II, changed from the previous rule of 60% of all patients using sequential all day rule starting their search on a particular day at 9 AM to 40%, and 40% start their search starting at 1 PM to 60%. Goal was to distribute more patients to the afternoon time slots than in the morning slots.
Experiment X	Similar to Experiment IV, changes were made to let the physicians take the full amount of time required of the patients in the examination room and not split the time up 1/3 with the patient and 2/3 as note taking time.
Experiment XI	Changed input data: mode was decreased. Second baseline model.
Experiment XII	Changed the patient mix: percent of category 4B and 5 were decreased from .08 to .05 and category 1B and category 1C were increased from .1 to .13 and .02 to .05 respectively.
Experiment XIII	Time slot scheduling rule for category 5 patient was changed from 20-80 to 50-50 (AM-PM).
Experiment XIV	Reduced the number of medical assistants from three to one.
Experiment XV	Increased the number of examination rooms from six to ten.

Table 7.2: Baseline Configuration

Parameters	Values
Number of Medical Assistants	3
Number of Nurses	2
Number of Physician Assistants/Nurse Practitioners	2
Number of Physicians	2
Number of Examination Rooms	6
Number of Check-in Rooms	2
Number of Specialty Rooms	1
Number of Registration Windows	2
Number of Chairs	26
Mean Walk-in Interarrival Time	6750 sec
Mean Morning Call Interarrival Time	500 sec
Mean Afternoon Call Interarrival Time	100 sec
Mean Evening Call Interarrival Time	1000 sec
Mean Night Call Interarrival Time	3000 sec
Mean Weekend Daytime Call Interarrival Time	1250 sec
Mean Weekend Evening Call Interarrival Time	1500 sec
Mean Weekend Night Call Interarrival Time	3500 sec

The change in the scheduling scheme in Experiment II was motivated by the uneven distribution of patient flow during the morning and afternoon periods of Experiment I. Up to this point, patients using the sequential all day time slot scheduling rule start their search for an available time slot at 9 a.m and patients start filling in their appointments with the morning time slots and, if a time slot has not been found, check the afternoon time slots. This caused an imbalance between the number of patients entering the clinic in the morning hours versus the afternoon hours. Therefore, in Experiment II, sixty percent of patients with the sequential all day time slot scheduling rule would start their search on a particular day at 9AM, while forty percent start their search at 1 PM. This resulted in an average daily overtime that was slightly greater than the previous results but more evenly distributed over the five weekdays, averaging 62.2 minutes in

length. Additionally, the ratio of the average number of patients arriving in the morning to those arriving in the afternoon changed from 36 to 30. This patient redistribution of resulted in a reduction in the patients' ALOS from 133.0 minutes to 114.2 minutes. However, the maximum number of people in the waiting room buffer and in the internal waiting area remained high at 14 and 22, respectively.

In Experiment III, the number of clinic time slots (every fifteen minutes) available to each physician was reduced from two to one. This means that only one clinic appointment patient can be scheduled for a physician every fifteen minutes for the clinic, in contrast to the earlier policy of allowing a maximum of two clinic appointment patients every fifteen minutes. Prior to this change, most of the clinic appointment patients were scheduled in the first hour of either the morning or afternoon time period, hence causing resources (both medical staff and the exam rooms) to be strained in the beginning of both the morning and the afternoon periods. If these clinical appointment patients were distributed further apart throughout the morning or afternoon periods, then the workload on the clinic could be distributed more evenly throughout the day. The result was a slight decrease in the average daily overtime by 4 minutes and the patient ALOS by 4 minutes from 114.2 minutes to 110.1 minutes, and an increase of one percent in the physician utilization from .70 to .71. This experiment did not yield a significant improvement in performance (e.g., patient ALOS, average daily overtime, patient's average overall waiting time, maximum number of patient in the waiting room buffer), since the number of clinic appointment patients arriving into the clinic remained small compared to the patients who required service by the physician.

While collecting actual data from the clinic located in Christiansburg, Virginia, it was observed that the medical staff member spent most of their time working away from the patient being assisted; the medical staff member would treat the patient in the examination room or check-in area, and then return to their desk or office to take notes, conduct tests, and perform other tasks associated with that particular patient. Based on this observation, Experiment IV required medical staff members to spend one third of their time (e.g., for check-in, examination) in actual face-to-face contact with the patient, while the remaining time was spent taking notes after the medical staff member returned to their desk or office. This resulted in a decrease from 58.7 minutes to 32.1 minutes in the average daily overtime, a decrease in the maximum number of patients waiting in the waiting room buffer from 9 to 4, and a decrease in the ALOS of the patient visit for all patient categories from 110 minutes to 80 minutes. Additionally, the average time for the physician's lunch break increased from 13.5 minutes to 29.9 minutes. However, the maximum number of patients waiting in the internal waiting area increased from 22 to 27. All of the improvements (decrease in average daily overtime, maximum number of patients in the waiting room buffer, and ALOS) are attributed to the more efficient processing of patients during their visit. However, since the patient's face-to-face treatment time is shorter but the medical staff member's total treatment time stays the same, more patients are being processed with an associated increase in the maximum number of patients in the internal waiting area.

As noted in Chapter 4.5, an additional insight gained from the observation of the Christiansburg clinic was that the check-in processing time was not as long as initially suggested by the domain experts. Therefore, Experiment V incorporated check-in processing times 50% less than those in Experiment I. This change decreased the average daily overtime from 32.1 to 25.9 minutes, and

the maximum number of patients waiting in the internal waiting from 27 to 18. The output also indicated a slight increase in the physician's utilization (from .71 to .72) that can be partially explained by the slight decrease in the average daily overtime of the clinic. Therefore, physicians are treating the same number of patients for the same amount of time but over a shorter workday.

Up to Experiment V, patients generated in the model entered the clinic and waited for service by a predetermined medical staff member (e.g., nurse, medical assistant). If service was required of a medical staff member that was in great demand, the patient would experience a longer waiting period, even though a medical staff member with higher qualifications was available. In Experiment VI, a staff hierarchy was incorporated. If a patient needed the assistance of a specific type of medical staff member and all such medical staff members were busy, then a medical staff member with more medical experience (i.e., a higher level in the staff hierarchy) would treat the patient. Physicians are not included in the hierarchy since they do not conduct the types of services that a non-physician staff member would provide. The physician assistant/nurse practitioner is at the top of the hierarchy; they can perform any duties up to the services only a physician can provide. Next in the hierarchy is the nurse and the medical assistant. For example, if a patient requires a nurse and all of the nurses are busy, then any available physician assistant can treat the patient; however, a medical assistant can not treat the patient. The medical staffing hierarchy resulted in a increase in the average utilization of the physician assistant from .36 to .45. Moreover, the patient ALOS and the patient's average waiting time in the internal waiting area and the examination room all decreased slightly. Furthermore, the seating did not exceed maximum capacity of 26 people; every patient and companion entering the clinic had an available seat if required to wait. Additionally, the physician utilization did not increase,

suggesting that the physician assistants met the demand for service. Therefore, this model increased the utilization of the physician assistant, and lowered the utilization of the nurses and medical assistants, but did not significantly change the patient's ALOS or the average overall waiting time.

The next two experiments verified that the model could perform correctly with one physician (Experiment VII) and with four physicians (Experiment VIII). The number of medical staff members for each model was selected based on the number of physicians. Experiment VII did not include a nurse, so as to demonstrate that the model could execute correctly without one, and Experiment VIII's staffing was selected to determine if one of each medical staff member type (one medical assistant, one nurse, and one physician assistant) per physician was sufficient to operate the clinic. Table 7.3 shows the staff composition for the three different physician configurations. These two experiments produced similar output measures. For example, the average daily overtime for a single physician clinic is 34.7 minutes compared to 37.4 minutes for a four-physician clinic. The number of patients processed by the four-physician clinic was approximately four times that of the single physician clinic.

Table 7.3: Staff composition for physician practices

	One Physician	**Two Physician**	**Four Physician**
Number of Medical Assistants	2	3	4
Number of Nurses	0	2	4
Number of Physician Assistants	2	2	4

Experiment IX attempted to distribute patients throughout the day, similar to Experiment II. All previous two physician experiments (up to Experiment IX) had an average of 48 patients arriving

in the morning and 37 patients arriving in the afternoon. Since the clinic's normal operating hours are from 9:00 AM to noon (three hours) and from 1:00 PM to 5:00 PM (four hours), the patients would be more evenly distributed if the scheduling search rule was reversed to 40% starting their search in the morning and 60% starting their search in the afternoon. This resulted in an average of 37 patients arriving in the morning and 49 patients arriving in the afternoon, which is closer to the 3:4 ratio for morning to afternoon clinic operating hours. The results from this model revealed a decrease in the model clinic's performance (as compared to the results from Experiment VI), especially for average daily overtime of the clinic (from 24.2 minutes to 58.0 minutes) and the patient's ALOS (from 68.6 to 76.3 minutes). When a clinic schedules more patients during the afternoon period, the excess demand can only be met by overtime. Additionally, the average daily lunch break for the physician increased due to the lighter workload in the morning hours. Although, this experiment decreased the model clinic's performance in several output measures, the modeler and the domain experts all had greater confidence in the validity of the Queston simulation model since it more accurately reflected how clinics schedule patients throughout the day.

Validation issues play a key role in Experiment X. During our time study of the Christiansburg clinic, physicians were observed moving between patients with minimal or no note-taking time between patients. Therefore, Experiment X was a modification of Experiment IV, which assumed all staff, including physicians, would spend two-thirds of their time taking notes, to a more realistic rule that only non-physician medical staff spend two-thirds of their time taking notes. This experiment increased the clinic's average daily overtime from 58.0 to 66.8 minutes, since physicians were physically spending more time with patients. If a patient was required to

see another medical staff member after the physician, this medical staff member could not begin treating the patient until the physician has completed their service. This restriction decreased the performance of the model, from the previous experiment, and is reflected in an increase in the patient's ALOS (from 76.3 to 88.2 minutes), an increase in the patient's average overall waiting time (from 57.6 to 69.5 minutes), and an increase in the patient's average waiting time in the internal waiting area (from 21.8 to 30.9 minutes). In contrast, a patient's average waiting time in an examination room decreased from 24.7 minutes to 20.9 minutes since the physician is in the examination room with the patient rather than spending a third of the patient's service time in their office.

After analyzing the data from the time study and the results from the Question simulation model runs, the domain experts reevaluated the Initial Patient Category Input Data in Appendix A.1. Several modifications were made to the times and percentages in the Initial Patient Category Input Data in Appendix A.1 to ensure greater confidence in the results (see Chapter 4.5 for further details). The percentage of patients requiring the attention of the physician was increased for all of the patient categories. Additionally, the average time to process a category 4A and 4B was decreased by 22.7%, and category 5 patients by 20.3%. Furthermore, the mode times for most processes (e.g., check-in, pre-examination, examination, pre-examination) were decreased for all of the patient categories. All these changes are reflected in the Revised Patient Category Input Data in Appendix A.2.

Experiment XI, reflecting these changes, showed a significant decrease in the average daily overtime (from 66.8 minutes to 24.3 minutes), a patient's ALOS (from 88.2 minutes to 61.0

minutes), and patient's average overall waiting time (from 69.5 minutes to 44.2 minutes). The staff utilization remained relatively unchanged; however, the physician utilization dropped from .70 to .60. These results were expected since the medical staff members were not required to spend as much time with the patients as in the earlier experiments. This model was accepted by the domain experts, and hence, Experiment XI was used as the second baseline model for further experimentation.

7.3 Results from the Second Baseline Model

Experiment XII tested the model's sensitivity to the patient population mix. The probabilities that a category 4B or category 5 patient entered the clinic were both decreased from .08 to .05, while the probabilities that a patient entering the clinic was either a category 1B or 1C were both increased from .03 to .13, respectively. These changes greatly reduced the average daily overtime from 24.3 to 10.2 minutes per day and the average physician utilization from .60 to .54. A slight decrease was also observed in the average patient waiting time (from 44.2 minutes to 39.7 minutes). These results were expected since the new patient population mix decreased the overall patient population's ALOS from 61.0 minutes to 56.5 minutes by reducing the number of (longer visit) category 4B and category 5 patients and increasing the number of (shorter visit) category 1B and 1C patients. Surprisingly, the maximum number of people in the waiting room buffer increased from 1 to 7. This can be explained in part by the increase in the number of clinical appointments (from 19 to 22). Therefore the same number of patients are being serviced

by the same amount of resources (waiting room chairs), but in a shorter period of time, and hence, the greater likelihood of chairs being unavailable.

Experiment XIII tested the Queston simulation model's sensitivity to changes in the time slot scheduling rule (i.e., the time of day a patient category is scheduled to arrive at the clinic). All category 5 patient time slot scheduling rule was changed (from 20% starting the search for a time slot at 9:00 AM and 80% starting the search for a time slot at 1:00 PM) to 50% starting the search for a time slot at 9:00 AM and 50% starting the search for a time slot at 1:00 PM Similar to Experiment XII, Experiment XIII resulted in a (very low) average daily overtime per day of 3.5 minutes. Moreover, Tuesday and Thursday showed no average daily overtime hours, and Monday, Wednesday, and Friday produced average daily overtimes of 4.2 minutes, 4.5 minutes, and 8.6 minutes, respectively. In conclusion, since category 5 patients have such a long ALOS (88.5 minutes), the average daily overtime for the clinic can be almost completely eliminated by moving a few of these patients to an earlier time slot. However, shifting these patients to an earlier appointment time resulted in the physician losing an average of five minutes per day from their lunch break.

One of the key objectives in the Queston simulation model study is the maximization of resources while maintaining a high level of patient service satisfaction. Therefore, reduction in resources (e.g., requiring less examination rooms, smaller non-physician medical staff) must be achieved by participating clinics, while maintaining comparable or better clinic performance (e.g., average daily overtime, average physician utilization). Based on Experiment XIII, Experiment XIV tested the effects of staffing changes by reducing the

number of medical assistants from three to one. After running the simulation, Experiment XIV showed no significant changes in the average physician utilization rate, average daily overtime, patient ALOS, average overall waiting time, and average room utilization. However, the average daily total time both nurses are busy increased from 104 minutes to 157 minutes and the average daily total time both physician assistants are busy increased from 131 minutes to 179 minutes. This can be explained by the model's medical staff hierarchy (e.g., a patient that a medical assistant may have been required to attend to can be processed by a more experienced and available nurse or physician assistant), resulting in a decrease in both patient average waiting times and staff idle times. This experiment suggests the potential to decrease the size of the medical staff without significantly impacting in clinic performance.

The final set of experiments varied the number of examination rooms. From Experiment XIII, the number of examination rooms was increased from six to ten in Experiment XV. The results show that all examination rooms are used at some time; however, the examination rooms are being used more as a waiting room than as an examination room. Patients wait in the examination rooms for longer periods of time for assistance than in Experiment XIII since the medical staff members has not increased. This result is shown by the decrease in the maximum number of people in the waiting room buffer from 6 to 1 and the decrease in the maximum number of patients in internal waiting area from 17 to 15. However, the results show only a minimal improvement in the average daily overtime (from 3.5 minutes to 3.2 minutes per day), in patient ALOS (from 52.2 minutes to 52.0 minutes), and in patient's average overall waiting time (from 36.4 minutes to 36.1 minutes). The average physician

utilization remained unchanged at .56. Since ten examination rooms did not provide any significant improvements, it can be concluded that any more than six examination rooms would be a waste of resources. This experiment suggests that relaxing only one constraint (examination rooms) by providing additional examination rooms into the model, the model reaches a saturation point at which other resource constraints, such as a lack of medical staff member, will not allow any further improvements.

7.4 Summary

The Queston simulation model developed into a realistic representation of a healthcare clinic through fifteen model experiments. The first eleven experiments were used to validate and verify the model and to identify a more realistic second baseline model (Experiment XI). The second baseline model was used for the last four experiments to determine the model's sensitivity to patient population mix, patient time slot scheduling rule, staffing allocation, and examination room allocation. The results suggest that patient mix and patient scheduling can have a significant impact on average daily clinic overtime and average physician utilization. Furthermore, a limited reduction in staffing can occur, without causing any significant reduction in the clinic's average daily overtime, patient's average waiting times, or patient ALOS. For further information on such an analysis, see Swisher [1999].

Several insights can be extracted from these experiments. *Patient load balancing* is the technique of distributing patients throughout the day to balance demand (patients needing

service) with available resources (facility and medical staff members). This technique increases customer satisfaction through lower waiting times while maintaining high resource utilization. Experiment XIII demonstrated that waiting times decreased by evenly distributing (throughout the clinic hours) patient category 5's arrival time, which is a queueing problem. In an ideal world, if the patient arrival distribution is such that unlimited resources are available when patients arrive, then patients would experience no waiting time. However, in the absence of this perfect situation, fine tuning the scheduling of patients can have an immediate and significant impact. This was demonstrated in Experiment II, Experiment IX, and Experiment XIII, where patients were evenly distributed throughout the day.

Lastly, adding resources, whether in medical staffing or facilities (e.g., examination rooms), will not necessarily improve performance if these resources are underutilized or utilized in a manner that will not assist in patient throughput. There is an optimal number of staff and facility resources for a given set of patients and schedules. Having an additional amount of one resource will not necessarily improve a clinic's performance if this resource is not scarce or limited.

Chapter 8 Summary and Recommendations

In this chapter, the design and development of the Queston simulation model is summarized and recommendations for enhancements and future research are suggested.

8.1 Summary

In this thesis, the development of the Queston simulation model is presented for a single family practice healthcare clinic and the Queston Information Center. A life-cycle modeling methodology was used to structure the design and development of this model. The Queston simulation model is built using the VSE, an object-oriented, visual simulation software package. The visualization features of VSE enhanced the perceived value of the simulation model for non-technical users (since they can visually observe the animation of the QPN operations), such as the QPN upper management and the individually owned clinic decision-makers (e.g., physicians, clinic managers). A rigorous VV&T ensured that the model has been programmed correctly and that it provides a valid representation of the QPN. Fifteen simulation experiments were conducted to move the representation of the simulation model

towards the actual QPN system.

8.2 Recommendations

The Queston simulation model captures many features of the QPN and provides valuable insights into determining the input data that are critical to a family practice healthcare clinic's performance, such as the mix of patient categories entering the clinic and the balance of patient load throughout the work day. One possible enhancement to the Queston simulation model would be to increase the number of clinics. To truly understand the operations of the QPN and to determine the operational gains of a network of clinics, several clinics would have to be instantiated. Fortunately, the VSE has the capability to instantiate additional clinics. Furthermore, it would be useful to measure the impact of additional clinics in the Queston simulation model on the overall performance of clinics in the QPN. As noted in Chapter 7, balancing the patient load within a clinic greatly improved the performance of the clinic (e.g. patient waiting time, staff utilization). If this approach was extended to balancing patients within a group of clinics in a local area, it may provide further gains in clinic performance (e.g., average daily overtime, patient ALOS).

It is possible to improve the accuracy of the input data used in the Queston simulation model. The data collected in the time study at the Christiansburg, Virginia clinic only assisted the domain experts in modifying the patient input data. The small amount of input data

collected could not be fit to parametric distributions. Additionally, input data on the fees assessed per patient category, operating costs (e.g., staff normal and overtime hour wages, facility overheads), and fixed costs (facility) have only been estimated by the domain experts (see Appendix A.2). This input data along with a quantification of the patient's satisfaction (using waiting times), can be used as a measure of the clinic's overall performance by scoring the clinic based on its configuration (e.g., number of examination rooms, check-in rooms, registration windows, seats). This measure can be used in the output analysis to determine which configuration is optimal for a specific number of physicians (e.g., two physician clinic, three physician clinic).

An area of concern, and hence, a need for future research, is the warm up or the transient period associated with filling the appointment book in the simulation model. A three month warm up was deemed suitable for model development, but this assumption should not be used for the output analysis and comparison of configuration. A clear understanding of the initial transient problem and a determination of the truncation time needed to warm up the simulation model is required for output analysis.

There are many other model enhancements that can be implemented, but the current Queston simulation model captures the essential details of the QPN and addresses the performance capabilities of the QPN. The Queston simulation model provides the end user with a flexible tool that can be easily modified to model a typical family practice healthcare clinic. Moreover, the model can be used in future research when analyzing the physician-patient

encounter to identify optimal clinic configurations that maximizes clinic performance, including patient satisfaction, staff satisfaction, and clinic profit. Swisher [1999] provides a first step towards meeting this objective.

References

Aggarwal S and Stafford EF Jr (1976). A simulation study to identify important design parameters of a typical outpatient health system and to analyze measures of its performance. In: *Proceedings of the 1976 Summer Computer Simulation Conference*, Simulation Council, Washington, DC, USA,12-14 July, pp 544-553.

Alessandra AJ, Grazman TE, Parameswaran R, and Yavas U (1978). Using simulation in hospital planning. *Simulation* **30**:2:62-67.

Altinel IK and Ulas E (1996). Simulation modeling for emergency bed requirement planning. *Annals of Operations Research* **67**:183-210.

Amladi P (1984). Outpatient health care facility planning and sizing via computer simulation. In: Sheppard S, Pooch UW, and Pegden CD (eds). *Proceedings of the 1984 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Dallas, Texas, USA, 28-30 November, pp 705-711.

Badri M and Hollingsworth J (1993). A simulation model for scheduling in the emergency room. *International Journal of Operations and Production Management* **13**:3:13-24.

Baezner D, Lomow G, and Unger BW (1990). Sim++^{TM}: The transition to distributed simulation. In: Nicol D (ed). *Proceedings of the 1990 SCS Western Multiconference on Simulation: Distributed Simulation*. Society for Computer Simulation, San Diego, California, USA, 17-19, January, pp 211-218.

Bailey NT (1952). A study of queues and appointment systems in hospital outpatient departments, with special reference to waiting times. *Journal of the Royal Statistical Society* **A14**: 185-199.

Balci O (1994). Validation, verification, and testing techniques throughout the life-cycle of a simulation study. *Annals of Operations Research* **53**:121-173.

Balci O (1998). Verification, Validation, and Accreditation. In: Mediros DJ, Watson EF, Carson JS, and Manivannan MS (eds). *Proceedings of the 1998 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 13-16 December, pp 41-48.

Balci O, Bertelrud AI, and Esterbrook CM (1996). *Visual Simulation Environment Version 1.0: User's Guide*. Orca Computer, Inc., Blacksburg, Virginia, USA.

Balci O, Bertelrud AI, Esterbrook CM, and Nance RE (1998). The Visual Simulation Environment. In: Mediros DJ, Watson EF, Carson JS, and Manivannan MS (eds). *Proceedings of the 1998 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 13-16 December, pp 279-287.

Banks J (1996). Software for Simulation In: Charnes JM, Morrice DM, Brunner DT, and Swain JJ (eds). *Proceedings of the 1996 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Coronado, California, USA, 8-11 December, pp 31-38.

Banks J (1997). The future of simulation software: A panel discussion. In: Andradottir S, Healy KJ, Withers DE, and Nelson BL (eds). *Proceedings of the 1997 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 7-10 December, pp 166-173.

Banks J and Carson JS (1987). Applying the simulation process. In: Thesen A, Grant H, and Kelton WD (eds). *Proceedings of the 1987 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 14-16 December, pp 68-71.

Banks J, Carson JS, and Nelson BL (1996). *Discrete-Event System Simulation*. Prentice Hall, New Jersey, USA.

Beizer B (1990). *Software Testing Techniques*. Second Edition, Van Nostrand Reinhold, New York, New York, USA.

Benson D (1997). Simulation modeling and optimization using ProModel. In: Andradottir S, Healy KJ, Withers DE, and Nelson BL (eds). *Proceedings of the 1997 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 7-10 December, pp 587-593.

Blake JT and Carter MW (1996). An analysis of emergency room wait time issues via computer simulation. *INFOR* **34**:4:263-272.

Bodtker K, Wilson L, and Godolphin W (1992). Simulation as an aid to clinical chemistry laboratory planning. In: Anderson J (ed). *Proceedings of the 1992 Conference on Simulation in Health Care and Social Services*. Newport Beach, California, USA, 20-22 January, pp 15-18.

Booch G (1994). *Object-Oriented Analysis and Design with Applications*. Second Edition. The Benjamin/Cummings Publishing Company, Redwood City, California, USA.

Butler T, Reeves G, Karwan K, and Sweigart J (1992). Assessing the impact of patient care policies using simulation analysis. *Journal of the Society for Health Systems* **3**:3:38-53.

Butler TW, Karwan KR, and Sweigart JR (1992). Multi-level strategic evaluation of hospital plans and decisions. *Journal of the Operational Research Society* **43**:7:665-675.

Butler TW, Karwan KR, Sweigart JR, and Reeves GR (1992). An integrative model-based approach to hospital layout. *IIE Transactions* **24**:2:144-152.

Carlson RC, Hershey JC, and Kropp DH (1979). Use of optimization and simulation models to analyze outpatient health care settings. *Decision Sciences* **10**:412-433.

Carroll D (1996). MedModel-Healthcare simulation software. In: Charnes JM, Morrice DM, Brunner DT, and Swain JJ (eds). *Proceedings of the 1996 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Coronado, California, USA, 8-11 December, pp 441-446.

Carson JS (1996). AutoStat output statistical analysis for AutoMod Users. In: Charnes JM, Morrice DM, Brunner DT, and Swain JJ (eds). *Proceedings of the 1996 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Coronado, California, USA, 8-11 December, pp 492-499.

Carson Y and Maria A (1997). Simulation optimization: Methods and applications. In: Andradottir S, Healy KJ, Withers DE, and Nelson BL (eds). *Proceedings of the 1997 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 7-10 December, pp 118-126.

Carter MW, O'Brien-Pallas, Blake, McGillis, and Zhu (1993). Simulation, scheduling, and operating room. In: Anderson JG and Katzper M (eds). *1993 SCS Western Multiconference on Simulation: Simulation in the Health Sciences and Services*. Society for Computer Simulation, La Jolla, California, USA, 17-20 January, pp 28-30.

Cauglin D (1995). Verification, validation, and accreditation (VV&A) of models and simulations through reduced order metamodels. In: Alexopoulos C, Kang K, Lilegdon WR, and Goldsman D (eds). *Proceedings of the 1995 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 3-6 December, pp 1405-1412.

Coffin MA, Lassiter G, Killingsworth B, and Kleckley JW (1993). A simulation model of an X-ray facility. In: Anderson JG and Katzper M (eds). *1993 S CS Western Multiconference on Simulation: Simulation in the Health Sciences and Services*. Society for Computer Simulation, La Jolla, California, USA, 17-20 January, pp 3-7.

Cohen MA, Hershey JC, and Weiss EN (1980). Analysis of capacity decisions for progressive patient care hospital facilities. *Health Services Research* **15**:145-160.

Currie K, Iskander W, Michael L, and Coberly C (1984). Simulation modeling in health care facilities. In: Sheppard S, Pooch UW, and Pegden CD (eds). *Proceedings of the 1984 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Dallas, Texas, USA, 28-30 November, pp 713-717.

Davies R and Davies H (1994). Modeling patient flows and resource provision in health systems.

Omega **22**:2:123-131.

Dearie D, Gerson J, and Warfield T (1976). The development and use of a simulation model of an outpatient clinic. In: *Proceedings of the 1976 Summer Computer Simulation Conference*, Simulation Council, Washington, DC, USA, 12-14 July, pp 554-558.

Draeger MA (1992). An emergency department simulation model used to evaluate alternative nurse staffing and patient population scenarios. In: Swain JJ, Goldsman D, Crain RC, and Wilson JR (eds). *Proceedings of the 1992 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, Arlington, Virginia, USA, 13-16 December, pp 1057-1064.

Drevna MC and Kasales CJ (1984). Introduction to Arena. In: Sheppard S, Pooch UW, and Pegden CD (eds). *Proceedings of the 1984 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Dallas, Texas, USA, 28-30 November, pp 431-436.

Dumas MB (1984). Simulation modeling for hospital bed planning. *Simulation* **43**:69-78.

Dumas MB (1985). Hospital bed utilization: An implemented simulation approach to adjusting and maintaining appropriate levels. *Health Services Research* **20**:1:43-61.

Edwards R, Clague J, Barlow J, Clarke M, Reed P, and Rada R (1994). Operations research survey and computer simulation of waiting times in two medical outpatient clinic structures. *Health Care Analysis* **2**:164-169.

England W and Roberts S (1978). Applications of computer simulation in health care. In: Highland HJ, Hull LG, and Neilsen (eds). *Proceedings of the 1978 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Miami Beach, Florida, USA, 4-6 December, pp 665-676.

Evans GW, Gor TB, and Unger E (1996). A simulation model for evaluating personnel schedules in a hospital emergency department. In: Charnes JM, Morrice DM, Brunner DT, and Swain JJ (eds). *Proceedings of the 1996 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Coronado, California, USA, 8-11 December, pp 1205-1209.

Fetter RB and Thompson JD (1965). The simulation of hospital systems. *Operations Research* **13**:5:689-711.

Fitzpatrick KE, Baker JR, and Dave DS (1993). An application of computer simulation to improve scheduling of hospital operating room facilities in the United States. *International Journal of Computer Applications in Technology* **6**:4:215-224.

Freedman RW (1994). Reduction of average length of stay in the emergency room using discrete simulation. In: Anderson JG and Katzper M (eds). *Proceedings of Simulation in Health Sciences.* Society for Computer Simulation, Tempe, Arizona, USA, 24-26 January, pp 6-8.

Gabaeff SC and Lennon J (1991). New computerized technology for the design and planning of emergency departments. In: *Proceedings of the American Hospital Association*. American Hospital Association, Anaheim, California, USA, July, pp 1-15.

Garcia ML, Centeno MA, Rivera C, and DeCario N (1995). Reducing time in an emergency room via a fast-track. In: Alexopoulos C, Kang K, Lilegdon WR, and Goldsman D (eds). *Proceedings of the 1995 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 3-6 December, pp 1048-1053.

Gipps PJ (1986). The role of computer graphics in validating simulation models. *Mathematics and Computers in Simulation* **28**:4:285-289.

Glasow P, Muessig P, Sikora J, Youngblood S, Balci O, Solick S, and Page E (1996). *Verification, Validation, and Accreditation (VV&A) Recommended Practice Guide*. Office of the Director of Defense Research and Engineering, Defense Modeling and Simulation Office, USA.

Glover F, Kelly JP, and Laguna, ML (1996). New advances and applications of combining simulation and optimization. In: Charnes JM, Morrice DM, Brunner DT, and Swain JJ (eds). *Proceedings of the 1996 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Coronado, California, USA, 8-11 December, pp 144-152.

Godolphin W, Bodtker K, and Wilson L (1992). Simulation modeling: A tool to help predict the impact of automation in clinical laboratories. *Laboratory Robotics and Automation* **4**:5:249-255.

Goitein M (1990). Waiting patiently. *The New England Journal of Medicine* **323**:9:604-608.

Gonzalez, Lopez-Valcarcel BG and Perez PB (1994). Evaluation of alternative functional designs in an emergency department by means of simulation. *Simulation* **63**:1:20-28.

Hancock W and Walter P (1979). The use of computer simulation to develop hospital systems. *Simuletter,* **10**:4:28-32.

Hancock W and Walter P (1984). The use of admissions simulation to stabilize ancillary workloads. *Simulation* **43**:2:88-94.

Hancock WM, Magerlein DB, Storer RH, and Martin JB (1978). Parameters affecting hospital occupancy and implications for faculty sizing. *Health Services Research* **13**:276-289.

Harris RA (1985). Hospital bed requirements planning. *European Journal of Operational Research* **25**:121-126.

Hashimoto F and Bell S (1996). Improving outpatient clinic staffing and scheduling with computer simulation. *Journal of General Internal Medicine* **11**:182-184.

Hermann CF (1967). Validation problems in games and simulations with special reference to

models of international politics. *Behavioral Sciences*, **12**:3, pp. 216-231.

Ishimoto K, Ishimitsu T, Koshiro A, and Hirose S (1990). Computer simulation of optimum personnel assignment in hospital pharmacy using a work-sampling method. *Medical Informatics* **15**:4:343-354.

Iskander WH and Carter MW(1991). A simulation model for a same day care facility at a university hospital. In: Nelson BL, Kelton WD, and Clark GM (eds). *Proceedings of the 1991 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Phoenix, Arizona, USA, 8-11 December, pp 846-853.

Joines JA and Roberts SD (1998). Fundamentals of Object-Oriented Simulation In: Mediros DJ, Watson EF, Carson JS, and Manivannan MS (eds). *Proceedings 1998 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 13-16 December, pp 141-149.

Jones LM and Hirst AJ (1986). Visual simulation in hospitals: A managerial or a political tool? *European Journal of Operational Research* **29**:167-177.

Kachhal SK, Klutke GA, and Daniels EB (1981). Two simulation applications to outpatient clinics. In: Oren TI, Delfosse CM, and Shubl CM (eds). *Proceedings of the 1981 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 9-11 December, pp 657-665.

Kanon D (1974). Simulation of waiting line problems in a hospital setting. In: Anderson J and Forsythe JM (eds). *Proceedings of the 1st World Conference on Medical Information, Medinfo '74*, North-Holland, Stockholm, Netherlands, 5-10 August, pp 503-507.

Keller LF (1994). MedModel-Specialized software for the healthcare industry. In: Tew JD, Manivannan S, Sadowski DA, and Seila AF (eds). *Proceedings of the 1994 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Lake Buena Vista, Florida, USA, 11-14 December, pp 533-537.

Kho JW and Johnson GM (1976). Computer simulation of a hospital health-care delivery system. In: Sargent RG, Highland HJ, and Schriber TJ (eds). *Proceedings of the 1976 Bicentennial Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Gaithersburg, Maryland, USA, 6-8 December, pp 349-360.

Khoshafian S and Abnous R (1995). *Object Orientation*. Second Edition. John Wiley & Sons, New York, New York, USA.

Kirtland A, Lockwood J, Poisker K, Stamp L, and Wolfe P (1995). Simulating an emergency department is as much fun as… In: Alexopoulos C, Kang K, Lilegdon WR, and Goldsman D (eds). *Proceedings of the 1995 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 3-6 December, pp 1039-1042.

Klafehn KA (1987). Impact points in patient flows through a radiology department provided through simulation. In: Thesen A, Grant H, and Kelton WD (eds). *Proceedings of the 1987 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 14-16 December, pp 914-918.

Klafehn KA and Connolly M (1993). The simulation/animation of a new outpatient hematology laboratory. In: Anderson JG and Katzper M (eds). *1993 SCS Western Multiconference on Simulation: Simulation in the Health Sciences and Services*. Society for Computer Simulation, La Jolla, California, USA, 17-20 January, pp 12-15.

Klafehn KA and Owens D (1987). A simulation model designed to investigate resource utilization in a hospital emergency room. In: Stead W (ed). *Proceedings of the 11th Annual Symposium on Computer Applications in Medical Care*. Institute of Electrical and Electronic Engineers, Washington DC, USA, 1-4 November, pp 676-679.

Klafehn KA, Owens D, Felter R, Vonneman N, and McKinnon C (1989). Evaluating the linkage between emergency medical services and the provision of scarce resources through simulation. In: Kingsland L, III (ed). *Proceedings of the 13th Annual Symposium on Computer Applications in Medical Care*. Institute of Electrical and Electronic Engineers, Washington DC, USA, 5-8 November, pp 5-8.

Klassen KJ and Rohleder TR (1996). Scheduling outpatient appointments in a dynamic environment. *Journal of Operations Management* **14**:83-101.

Kleijnen JP (1995). Verification and validation of simulation models. *European Journal of Operational Research,* **82**:1, pp. 145-162.

Klein RW, Dittus RS, Roberts SD, and Wilson JR (1993). Simulation modeling and health-care decision making. *Medical Decision Making* **13**:4:347-354.

Kletke MG and Dooley TE (1984). A simulation model of a maternity ward: A key planning tool for hospital administrators. In: *Proceedings of the Conference on Simulation in Health Care Delivery System*. San Diego, California, USA, 2-4 February, pp 35-39.

Kraitsik MJ and Bossmeyer A (1993). Simulation applied to planning an emergency department expansion. In: Anderson JG and Katzper M (eds). *1993 S CS Western Multiconference on Simulation: Simulation in the Health Sciences and Services*. Society for Computer Simulation, La Jolla, California, USA, 17-20 January, pp 19-27.

Kropp D and Hershey J (1979). Recursive optimization-simulation modeling for the analysis of ambulatory health care facilities. *Simuletter* **10**:4:43-46.

Kropp D, Carlson RC, and Jucker JV (1978). Use of both optimization and simulation models to analyze complex systems. In: Highland HJ, Hull LG, and Neilsen (eds). *Proceedings of the 1978*

Winter Simulation Conference. Institute of Electrical and Electronics Engineers, Miami Beach, Florida, USA, 4-6 December, pp 195-201.

Kumar AP and Kapur R (1989). Discrete simulation application-scheduling staff for the emergency room. In: MacNair EA, Musselman JK, and Heidelberger P (eds). *Proceedings of the 1989 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 4-6 December, pp 1112-1120.

Kuzdrall PJ, Kwak NK, and Schmitz HH (1981). Simulating space requirements and scheduling policies in a hospital surgical suite. *Simulation* **27**:163-171.

Kwak N, Kuzdrall P, and Schmitz H (1975). Simulating the use of space in a hospital surgical suite. *Simulation* **24**:5:147-152.

Lambo E (1983). An optimization-simulation model of a rural health center in Nigeria. *Interfaces* **13**:3:29-35.

Law AM and Kelton WD (1991). *Simulation Modeling & Analysis*. McGraw-Hill, New York, New York, USA.

Lehaney B and Hlupic V (1995). Simulation modeling for resource allocation and planning in the health sector. *Journal of the Royal Society of Health* **115**:6:382-385.

Lehaney B and Paul RJ (1994). Using soft systems methodology to develop a simulation of out-patient services. *Journal of the Royal Society for Health* **114**:248-251.

Lehaney B and Paul RJ (1996). The use of soft systems methodology in the development of a simulation of outpatient services at Watford General Hospital. *Journal of the Operational Research Society* **47**:864-870.

Lennon J (1992). Simulation in the design and planning of emergency departments. In: *1992 SCS Western Multiconference: Simulation in education for business, management, and MIS*. Society for Computer Simulation, Newport Beach, California, USA, January, pp 93-97.

Levy JL, Watford BA, and Owen VT (1989). Simulation analysis of an outpatient services facility. *Journal of the Society for Health Systems* 1:2:35-49.

Lewis, PAW and Shedler, GS (1979). Simulation of Nonhomogeneous Poisson Process by Thinning. *Nav. Res. Logist*. Quart., **26**: 403-413.

Lim T, Uyeno D, and Vertinsky I (1975). Hospital admissions systems: A simulation approach. *Simulation and Games* **6**:2:188-201.

Liyanage L and Gale M (1995). Quality improvement for the Campbelltown hospital emergency service. In: *1995 IEEE International Conference on Systems, Man, and Cybernetics*. Institute of

Electrical and Electronic Engineers, Vancouver, British Columbia, Canada, 22-25 October, **13**: pp 1997-2002.

Lowery JC (1992). Simulation of a hospital's surgical suite and critical care area. In: Swain JJ, Goldsman D, Crain RC, and Wilson JR (eds). *Proceedings of the 1992 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, Arlington, Virginia, USA, 13-16 December, pp 1071-1078.

Lowery JC (1993). Multi-hospital validation of critical care simulation model. In: Evans, GW, Mollaghasemi M, Russell EC, and Biles WE (eds). *Proceedings of the 1993 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Los Angeles, California, USA, 12-15 December, pp 1207-1215.

Lowery JC (1994). Barriers to implementing simulation in health care. In: Tew JD, Manivannan S, Sadowski DA, and Seila AF (eds). *Proceedings of the 1994 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Lake Buena Vista, Florida, USA, 11-14 December, pp 888-875.

Lowery JC (1996). Introduction to simulation in health care. In: Charnes JM, Morrice DM, Brunner DT, Swain JJ (eds). *Proceedings of the* 1*996 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Coronado, California, USA, 8-11 December, pp 78-84.

Lowery JC and Martin JB (1992). Design and validation of a critical care simulation model. *Journal of the Society for Health Systems* **3**:3:15-36.

Magerlein JM and Martin JB (1976). Surgical demand scheduling: A review. *Health Services Research* **11**:53-68.

Mahachek A (1992). An introduction to patient flow simulation for health-care managers. *Journal of the Society for Health Systems* **3**:3:73-81.

Mahachek AR and Knabe TL (1984). Computer simulation of patient flow in obstetrical/gynecology clinics. *Simulation* **30**:95-101.

Marsh J (1979). Simulation as a decision aid: A management perspective. *Simuletter* **10**:5:47-49.

McGuire F (1994). Using simulation to reduce length of stay in emergency departments. In: Tew JD, Manivannan S, Sadowski DA, and Seila AF (eds). *Proceedings of the 1994 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Lake Buena Vista, Florida, USA, 11-14 December, pp 861-867.

McHugh ML (1989). Computer simulation as a method for selecting nurse staffing levels in hospitals. In: MacNair EA, Musselman JK, and Heidelberger P (eds). *Proceedings of the 1989 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 4-6 December, pp 1121-1129.

Meier L, Sigal E, and Vitale FR (1985). The use of simulation model for planning ambulatory surgery. In: Gantz DT, Blasis GC, and Solomon SL (eds). *Proceedings of the 1985 Winter Simulation.* Institute of Electrical and Electronics Engineers, San Francisco, California, USA, 11-13 December, pp 558-563.

Mukherjee AK (1991). A simulation model for management of operations in the pharmacy of a hospital. *Simulation* **56**:2:91-103.

Murphy DR and Sigal E (1985). Evaluating surgical block scheduling using computer simulation. In: Gantz DT, Blasis GC, and Solomon SL (eds). *Proceedings of the 1985 Winter Simulation.* Institute of Electrical and Electronics Engineers, San Francisco, California, USA, 11-13 December, pp 551-557.

O'Kane PC (1981). A simulation model of a diagnostic radiology department. *European Journal of Operational Research* **6**:38-45.

Olson E and Dux LE (1994). Computer model targets best route for expanding hospital surgicenter. *Industrial Engineering* **26**:9:24-6.

Pardue J and Cognetta A (1995). A systems analysis and model of real-time skin cancer treatment. In: Mediros DJ, Watson EF, Carson JS, and Manivannan MS (eds). *Proceedings 1998 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 13-16 December, pp 1054-1061.

Paul RJ (1989). Visual simulation: Seeing is believing. *Impacts of Recent Computer Advances on Operations Research* **9**:422-432.

Paul RJ and Kuljis J (1995). A generic simulation package for organizing outpatient clinics. In: Alexopoulos C, Kang K, Lilegdon WR, and Goldsman D (eds). *Proceedings of the 1995 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 3-6 December, pp 1043-1047.

Rakich JS, Kuzdrall PJ, Klafehn KA, and Krigline AG (1991). Simulation in the hospital setting: Implications for managerial decision making and management development. *Journal of Management Development* **10**:4:31-37.

Rising EJ, Baron R, and Averill B (1973). A systems analysis of a university-health-service outpatient clinic. *Operations Research* **21**:5:1030-1047.

Ritondo M and Freedman RW (1993). The effects of procedure scheduling on emergency room throughput: A simulation study. In: Anderson JG and Katzper M (eds). *1993 SCS Western Multiconference on Simulation: Simulation in the Health Sciences and Services*. Society for Computer Simulation, La Jolla, California, USA, 17-20 January, pp 8-11.

Romanin-Jacur G and Facchin P (1987). Optimal planning of a pediatric semi-intensive care unit via simulation. *European Journal of Operational Research* **29**:192-198.

Sargent RG (1992). Validation and verification of simulation In: Swain JJ, Goldsman D, Crain RC, and Wilson JR (eds). *Proceedings of the 1992 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, Arlington, Virginia, USA, 13-16 December, pp 104-114.

Sargent RG (1998). Verification and Validation of Simulation Models In: Mediros DJ, Watson EF, Carson JS, and Manivannan MS (eds). *Proceedings 1998 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 13-16 December, pp 121-130.

Smith EA and Warner HR (1971). Simulation of a multiphasic screening procedure for hospital admissions. *Simulation* **17**:2:57-64.

Smith SR, Schroer BJ, and Shannon RE (1979). Scheduling of patients and resources for ambulatory health care. In: Highland HJ, Spiegel MG, and Shannon R (eds). *Proceedings of the 1979 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, San Diego, California, USA, 3-5 December, pp 553-562.

Smith-Daniels VL, Schweikhart SB, and Smith-Daniels DE (1988). Capacity management in health care services: Review and future research directions. *Decision Sciences* **19**:889-918.

Stafford EF Jr (1976). A General Simulation Model for Multifacility Outpatient Clinics. M.S. thesis, Pennsylvania State University, Pennsylvania.

Stafford EF Jr (1978). Simulation vs. mathematical analysis for systems modeling: A comparison of techniques. In: *Proceedings of the 1978 Summer Computer Simulation Conference*. AFIPS Press, Los Angeles, California, USA, 24-26 July, pp 153-159.

Steward D and Standridge CR (1996). A veterinary practice simulator based on the integration of expert system and process modeling. *Simulation* **66**:143-159.

Swisher JR (1999). Evaluation of the design of a family practice healthcare clinic using discrete-event simulation. M.S. thesis, Virginia Polytechnic Institute and State University, Virginia, USA.

Swisher JR, Jun BJ, Jacobson SH, and Balci O (1997). Simulation of the Queston physician network. In: Andradottir S, Healy KJ, Withers DE, and Nelson BL (eds). *Proceedings of the 1997 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Atlanta, Georgia, USA, 7-10 December, pp 1146-1154.

Valinsky D (1975). Simulation. In: Shuman LJ, Speas RD, and Young JP (eds). *Operations Research in Healthcare: A Critical Analysis*. The Johns Hopkins University Press: Baltimore, Maryland, USA.

Vassilacopoulos, G (1985). A simulation model for bed allocation to hospital inpatient departments. *Simulation* **45**:5:233-241.

Vemuri S (1984). Simulated analysis of patient waiting time in an outpatient pharmacy. *American Journal of Hospital Pharmacy* **41**:1127-1130.

Walter SD (1973). A comparison of appointment schedules in hospital radiology department. *British Journal of Preventive and Social Medicine* **27**:160-167.

Whitner, RB and Balci, O (1989). Guidelines for selecting and using simulation model verification techniques. In: MacNair EA, Musselman JK, and Heidelberger P (eds). *Proceedings of the 1989 Winter Simulation Conference.* Institute of Electrical and Electronics Engineers, Washington DC, USA, 4-6 December, pp 559-568.

Williams SV (1983). How many intensive care beds are enough? *Critical Care Medicine* **11**:6:412-416.

Williams WJ, Covert RP, and Steele JD (1967). Simulation modeling of a teaching hospital clinic. *Hospitals* **41**:21:71-75.

Wilson JCT (1981). Implementation of computer simulation projects in health care. *Journal of the Operational Research Society* **32**:825-832.

Wilt A and Goddin D (1989). Health care case study: Simulating staffing needs and work flow in an outpatient diagnostic center. *Industrial Engineering* **21**:5:22-26.

Wright MB (1987). The application of a surgical bed simulation model. *European Journal of Operational Research* **32**:26-32.

Yücesan E (1993). Randomization Tests for Initialization Bias in Simulation Output. *Naval Research Logistics* **40**:643-663.

Zilm F, Arch D, and Hollis RB (1983). An application of simulation modeling to surgical intensive care bed need analysis in a university hospital. *Hospital & Health Services Administration* **28**:5:82-101.

Appendix A: Patient Input Data from Domain Experts

Appendix A.1 Initial Patient Category Input Data

	Patient Categories									
	Cat 1a	**Cat 1b**	**Cat 1c**	**Cat 2**	**Cat 2PV**	**Cat 3**	**Cat 3PV**	**Cat 4a**	**Cat 4b**	**Cat 5**
Probability of Occurrence:	0.1	0.1	0.02	0.3	0.2*	0.28	0.2*	0.04	0.08	0.08
Fraction of Total Patients:	0.0896	0.0896	0.0179	0.2688	0.0538	0.2509	0.0502	0.0358	0.0717	0.0717 ^see below
Scheduling Lead Time in days: Min	0	2	2	2	2	4	4	5	2	5
Scheduling Lead Time in days: Mode	1	3	3	5	5	5	5	10	5	8
Scheduling Lead Time in days: Max	3	5	4	7	7	7	7	15	7	10
Probability of No Shows:	0	0	0	0	0	0	0	0	0	0
Revenue Generation Distribution: Min	$ 23	$ 23	$ 23	$ 38	$ 38	$ 54		$ 79	$ 79	$ 125
Revenue Generation Distribution: Mode										
Revenue Generation Distribution: Max	$ 35	$ 35	$ 35	$ 50	$ 50	$ 70		$ 92	$ 92	$ 140
Cost Distribution: Min				$ 10	$ 10	$ 14		$ 20	$ 20	$ 31
Cost Distribution: Mode	$ 6	$ 6	$ 6							
Cost Distribution: Max				$ 13	$ 13	$ 18		$ 23	$ 24	$ 37
Patient Time Slot Scheduling Rule:	100all	100all	100all	50all, 25am, 25pm	100all	100all	100am	100pm	100all	20am, 80pm ^see below

^s-specific time, am-sequential morning, pm-sequential afternoon, all-sequential all day

*Probability of previsit (given the patient is a Cat2 patient)

^These values were calculated by using the following equation:

Fraction of total patients for category X = P(CatX)/(sum of normal visit category's probability of occurrence + P(Cat2PV)*P(Cat2) +P(Cat3PV)*P(Cat3))

Probability of having a **Registration**:	1	1	1	1	1	1	1	1	1	1
with a clerical:	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
with a clerical and min time of:	1	2	3	3	3	3	3	5	5	5
with a clerical and mode time of:	2	3	4	4	4	4	4	7	7	8
with a clerical and max time of:	3	4	5	5	5	5	5	10	10	10
Average registration time (analytically)	2.0	3.0	4.0	4.0	4.0	4.0	4.0	7.3	7.3	7.7
Probability of having a **Check-In**:	1	1	0	1	0.2	1	0.2	1	1	1
with a Medical Assistant:	75%	15%	0%	65%	10%	65%	10%	30%	30%	15%
with a Med Assist and min time:	2	2	2	2	2	2	2	2	2	2
with a Med Assist and mode time:	5	5	5	5	5	5	5	5	5	6
with a Med Assist and max time:	10	10	10	10	10	10	10	10	10	10
with a Nurse:	20%	80%	0%	30%	90%	30%	90%	65%	65%	75%
with a Nurse and min time:	2	2	2	2	2	2	2	2	2	2
with a Nurse and mode time:	5	5	5	5	5	5	5	5	5	6
with a Nurse and max time:	10	10	10	10	10	10	10	10	10	10
with a NP/PA:	5%	5%	0%	5%	0%	5%	0%	5%	5%	10%
with a NP/PA and min time:	2	2	2	2	2	2	2	2	2	2
with a NP/PA and mode time:	5	5	5	5	5	5	5	5	5	6
with a NP/PA and max time:	10	10	10	10	10	10	10	10	10	10
with a Physician:	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
with a Physician and min time:	0	0	0	0	0	0	0	0	0	0
with a Physician and mode time:	0	0	0	0	0	0	0	0	0	0
with a Physician and max time:	0	0	0	0	0	0	0	0	0	0
Average check-in time (analytically)	5.7	5.7	0.0	5.7	1.1	5.7	1.1	5.7	5.7	6.0

Probability of having a **Pre-Exam**:	0	0	0	0.05	1	0.95	1	0.8	0.8	0.5
with a Technician:	0%	0%	0%	0%	0%	0%	0%	20%	20%	10%
with a Technician and min time:	0	0	0	0	0	0	0	8	8	8
with a Technician and mode time:	0	0	0	0	0	0	0	10	10	12
with a Technician and max time:	0	0	0	0	0	0	0	15	15	15
with a Medical Assistant:	0%	0%	0%	80%	80%	30%	80%	20%	20%	15%
with a Med Assist and min time:	0	0	0	3	3	3	3	8	8	8
with a Med Assist and mode time:	0	0	0	5	5	5	5	10	10	12
with a Med Assist and max time:	0	0	0	20	20	20	20	15	15	15
with a Nurse:	0%	0%	0%	10%	20%	60%	20%	50%	50%	50%
with a Nurse and min time:	0	0	0	3	3	3	3	8	8	8
with a Nurse and mode time:	0	0	0	4	4	4	4	10	10	12
with a Nurse and max time:	0	0	0	10	10	10	10	15	15	15
with a NP/PA:	0%	0%	0%	10%	0%	10%	0%	10%	10%	20%
with a NP/PA and min time:	0	0	0	3	0	3	0	8	8	8
with a NP/PA and mode time:	0	0	0	4	0	4	0	10	10	12
with a NP/PA and max time:	0	0	0	10	0	10	0	15	15	15
with a Physician:	0%	0%	0%	0%	0%	0%	0%	0%	0%	5%
with a Physician and min time:	0	0	0	0	0	0	0	0	0	8
with a Physician and mode time:	0	0	0	0	0	0	0	0	0	12
with a Physician and max time:	0	0	0	0	0	0	0	0	0	15
Average pre-exam time (analytically)	0.0	0.0	0.0	0.4	8.6	6.4	8.6	8.8	8.8	5.8

Probability of having an **Exam:**	0	0.3	0	1	0	1	0	1	1	1
with a Medical Assistant:	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
with a Med Assist and min time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and mode time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and max time:	0	0	0	0	0	0	0	0	0	0
with a Nurse:	0%	85%	0%	0%	0%	0%	0%	0%	0%	0%
with a Nurse and min time:	0	5	0	0	0	0	0	0	0	0
with a Nurse and mode time:	0	7	0	0	0	0	0	0	0	0
with a Nurse and max time:	0	8	0	0	0	0	0	0	0	0
with a NP/PA:	0%	15%	0%	40%	0%	80%	0%	60%	60%	40%
with a NP/PA and min time:	0	5	0	6	0	10	0	18	18	20
with a NP/PA and mode time:	0	7	0	10	0	12	0	20	20	25
with a NP/PA and max time:	0	8	0	12	0	15	0	25	25	30
with a Physician:	0%	0%	0%	60%	0%	20%	0%	40%	40%	60%
with a Physician and min time:	0	0	0	6	0	10	0	18	18	20
with a Physician and mode time:	0	0	0	8	0	12	0	20	20	25
with a Physician and max time:	0	0	0	12	0	15	0	25	25	30
Average examination time (analytically)	0.0	2.0	0.0	8.9	0.0	12.3	0.0	21.0	21.0	25.0

Probability of having a **Post Exam**:	0	0	0.5	0.8	0	1	0	0.2	0.2	0.8
with a Technician:	0%	0%	5%	0%	0%	0%	0%	20%	20%	20%
with a Technician and min time:	0	0	3	0	0	0	0	8	8	8
with a Technician and mode time:	0	0	4	0	0	0	0	10	10	10
with a Technician and max time:	0	0	5	0	0	0	0	12	12	12

with a Medical Assistant:	0%	0%	5%	80%	0%	0%	0%	25%	25%	25%
with a Med Assist and min time:	0	0	3	3	0	0	0	8	8	8
with a Med Assist and mode time:	0	0	4	5	0	0	0	10	10	10
with a Med Assist and max time:	0	0	5	20	0	0	0	12	12	12
with a Nurse:	0%	0%	80%	20%	0%	70%	0%	35%	35%	35%
with a Nurse and min time:	0	0	5	3	0	3	0	5	5	8
with a Nurse and mode time:	0	0	8	4	0	5	0	6	6	10
with a Nurse and max time:	0	0	10	10	0	20	0	8	8	12
with a NP/PA:	0%	0%	10%	0%	0%	30%	0%	15%	15%	15%
with a NP/PA and min time:	0	0	8	0	0	3	0	5	5	8
with a NP/PA and mode time:	0	0	12	0	0	4	0	6	6	10
with a NP/PA and max time:	0	0	15	0	0	10	0	8	8	12
with a Physician:	0%	0%	0%	0%	0%	0%	0%	5%	5%	5%
with a Physician and min time:	0	0	0	0	0	0	0	3	3	8
with a Physician and mode time:	0	0	0	0	0	0	0	5	5	10
with a Physician and max time:	0	0	0	0	0	0	0	6	6	12
Average post-exam time (analytically)	0.0	0.0	3.9	6.9	0.0	8.2	0.0	1.6	1.6	8.0
Probability of having a **Exit Interview**:	0	0	1	1	0	1	0	1	1	1
with a Medical Assistant:	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
with a Med Assist and min time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and mode time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and max time:	0	0	0	0	0	0	0	0	0	0
with a Nurse:	0%	0%	55%	25%	0%	25%	0%	20%	20%	0%
with a Nurse and min time:	0	0	2	3	0	3	0	5	5	0
with a Nurse and mode time:	0	0	3	4	0	4	0	7	7	0
with a Nurse and max time:	0	0	5	5	0	5	0	10	10	0
with a NP/PA:	0%	0%	30%	15%	0%	15%	0%	15%	15%	20%
with a NP/PA and min time:	0	0	2	3	0	3	0	5	5	8
with a NP/PA and mode time:	0	0	3	4	0	4	0	7	7	10
with a NP/PA and max time:	0	0	5	5	0	5	0	10	10	12
with a Physician:	0%	0%	10%	60%	0%	60%	0%	65%	65%	80%
with a Physician and min time:	0	0	2	3	0	3	0	8	8	5
with a Physician and mode time:	0	0	3	5	0	5	0	10	10	15
with a Physician and max time:	0	0	5	8	0	8	0	15	15	20
Average exit interview time (analytically)	0	0.0	3.2	4.8	0.0	4.8	0.0	9.7	9.7	12.7
Probability of having a **Check-Out**:	1	1	1	1	1	1	1	1	1	1
with a clerical:	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
with a clerical and min time of:	2	2	2	4	4	3	2	4	4	2
with a clerical and mode time of:	3	3	3	5	5	4	4	5	5	4
with a clerical and max time of:	4	4	4	6	6	5	5	6	6	5
Average registration time (analytically)	3.0	3.0	3.0	5.0	5.0	4.0	3.7	5.0	5.0	3.7

	Cat 1a	Cat 1b	Cat 1c	Cat 2	Cat 2PV	Cat 3	Cat 3PV	Cat 4a	Cat 4b	Cat 5
Average Total Time:	**10.7**	**13.7**	**14.0**	**35.7**	**18.7**	**45.5**	**17.4**	**59.1**	**59.1**	**68.8**
Average after 1/3 to 2/3 rule:	6.9	8.6	9.3	17.9	12.2	20.5	10.9	27.9	27.9	30.5

Appendix A.2 Revised Patient Category Input Data

	Patient Categories									
	Cat 1a	**Cat 1b**	**Cat 1c**	**Cat 2**	**Cat 2PV**	**Cat 3**	**Cat 3PV**	**Cat 4a**	**Cat 4b**	**Cat 5**
Probability of Occurrence:	0.1	0.1	0.02	0.3	0.2*	0.28	0.2*	0.04	0.08	0.08
Fraction of Total Patients:	0.0896	0.0896	0.0179	0.2688	0.0538	0.2509	0.0502	0.0358	0.0717	0.0717 ˜see below
Scheduling Lead Time in days: Min	0	2	2	2	2	4	4	5	2	5
Scheduling Lead Time in days: Mode	1	3	3	5	5	5	5	10	5	8
Scheduling Lead Time in days: Max	3	5	4	7	7	7	7	15	7	10
Probability of No Shows:	0.05	0.07	0.05	0.2	0.2	0.2	0.2	0.33	0.05	0.05
Revenue Generation Distribution: Min	$ 23	$ 23	$ 23	$ 38	$ 38	$ 54		$ 79	$ 79	$ 125
Revenue Generation Distribution: Mode										
Revenue Generation Distribution: Max	$ 35	$ 35	$ 35	$ 50	$ 50	$ 70		$ 92	$ 92	$ 140
Cost Distribution: Min				$ 10	$ 10	$ 14		$ 20	$ 20	$ 31
Cost Distribution: Mode	$ 6	$ 6	$ 6							
Cost Distribution: Max				$ 13	$ 13	$ 18		$ 23	$ 24	$ 37
Patient Time Slot Scheduling Rule:	100all	100all	100all	50all, 25am, 25pm	100all	100all	100am	100pm	100all	20am, 80pm ˜see below

˜s-specific time, am-sequential morning, pm-sequential afternoon, all-sequential all day

*Probability of previsit (given the patient is a Cat2 patient)

˜These values were calculated by using the following equation:

Fraction of total patients for category X = P(CatX)/(sum of normal visit category's probability of occurrence + P(Cat2PV)*P(Cat2) +P(Cat3PV)*P(Cat3))

Probability of having a **Registration**:	1	1	1	1	1	1	1	1	1	1
with a clerical:	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
with a clerical and min time of:	1	2	3	3	3	3	3	5	5	5
with a clerical and mode time of:	2	3	4	4	4	4	4	7	7	8
with a clerical and max time of:	3	4	5	5	5	5	5	10	10	10
Average registration time (analytically)	2.0	3.0	4.0	4.0	4.0	4.0	4.0	7.3	7.3	7.7

Probability of having a **Check-In**:	1	1	0	1	0.2	1	0.2	1	1	1
with a Medical Assistant:	75%	15%	0%	65%	10%	65%	10%	30%	30%	15%
with a Med Assist and min time:	2	2	2	2	2	2	2	2	2	2
with a Med Assist and mode time:	4	4	4	4	4	4	4	4	4	5
with a Med Assist and max time:	10	10	10	10	10	10	10	10	10	10
with a Nurse:	20%	80%	0%	30%	90%	30%	90%	65%	65%	75%
with a Nurse and min time:	2	2	2	2	2	2	2	2	2	2
with a Nurse and mode time:	4	4	4	4	4	4	4	4	4	5
with a Nurse and max time:	10	10	10	10	10	10	10	10	10	10
with a NP/PA:	5%	5%	0%	5%	0%	5%	0%	5%	5%	10%
with a NP/PA and min time:	2	2	2	2	2	2	2	2	2	2
with a NP/PA and mode time:	4	4	4	4	4	4	4	4	4	5
with a NP/PA and max time:	10	10	10	10	10	10	10	10	10	10
with a Physician:	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
with a Physician and min time:	0	0	0	0	0	0	0	0	0	0
with a Physician and mode time:	0	0	0	0	0	0	0	0	0	0
with a Physician and max time:	0	0	0	0	0	0	0	0	0	0
Average check-in time (analytically)	5.3	5.3	0.0	5.3	1.1	5.3	1.1	5.3	5.3	5.7

Probability of having a **Pre-Exam**:	0	0	0	0.05	1	0.95	1	0.8	0.8	0.5
with a Technician:	0%	0%	0%	0%	0%	0%	0%	20%	20%	10%
with a Technician and min time:	0	0	0	0	0	0	0	6	6	6
with a Technician and mode time:	0	0	0	0	0	0	0	8	8	8
with a Technician and max time:	0	0	0	0	0	0	0	12	12	12
with a Medical Assistant:	0%	0%	0%	80%	80%	30%	80%	20%	20%	10%
with a Med Assist and min time:	0	0	0	3	3	3	3	6	6	6
with a Med Assist and mode time:	0	0	0	5	5	5	5	8	8	8
with a Med Assist and max time:	0	0	0	15	15	15	15	12	12	12
with a Nurse:	0%	0%	0%	10%	20%	60%	20%	50%	50%	50%
with a Nurse and min time:	0	0	0	3	3	3	3	6	6	6
with a Nurse and mode time:	0	0	0	4	4	4	4	8	8	8
with a Nurse and max time:	0	0	0	10	10	10	10	12	12	12
with a NP/PA:	0%	0%	0%	10%	0%	10%	0%	10%	10%	20%
with a NP/PA and min time:	0	0	0	3	0	3	0	6	6	6
with a NP/PA and mode time:	0	0	0	4	0	4	0	8	8	8
with a NP/PA and max time:	0	0	0	10	0	10	0	12	12	12
with a Physician:	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%
with a Physician and min time:	0	0	0	0	0	0	0	0	0	6
with a Physician and mode time:	0	0	0	0	0	0	0	0	0	8
with a Physician and max time:	0	0	0	0	0	0	0	0	0	12
Average pre-exam time (analytically)	0.0	0.0	0.0	0.4	7.3	6.0	7.3	6.9	6.9	4.3

	Cat 1a	**Cat 1b**	**Cat 1c**	**Cat 2**	**Cat 2PV**	**Cat 3**	**Cat 3PV**	**Cat 4a**	**Cat 4b**	**Cat 5**
Probability of having an **Exam:**	0	0.3	0	1	0	1	0	1	1	1
with a Medical Assistant:	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
with a Med Assist and min time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and mode time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and max time:	0	0	0	0	0	0	0	0	0	0
with a Nurse:	0%	85%	0%	0%	0%	0%	0%	0%	0%	0%
with a Nurse and min time:	0	5	0	0	0	0	0	0	0	0
with a Nurse and mode time:	0	7	0	0	0	0	0	0	0	0
with a Nurse and max time:	0	8	0	0	0	0	0	0	0	0
with a NP/PA:	0%	15%	0%	20%	0%	60%	0%	50%	50%	20%
with a NP/PA and min time:	0	5	0	4	0	6	0	8	8	10
with a NP/PA and mode time:	0	7	0	6	0	8	0	10	10	15
with a NP/PA and max time:	0	8	0	10	0	11	0	20	20	30
with a Physician:	0%	0%	0%	80%	0%	40%	0%	50%	50%	80%
with a Physician and min time:	0	0	0	4	0	6	0	8	8	10
with a Physician and mode time:	0	0	0	6	0	8	0	10	10	15
with a Physician and max time:	0	0	0	10	0	11	0	20	20	30
Average examination time (analytically)	0.0	2.0	0.0	6.7	0.0	8.3	0.0	12.7	12.7	18.3

Probability of having a **Post Exam**:	0	0	0.5	0.8	0	1	0	0.2	0.2	0.8
with a Technician:	0%	0%	5%	0%	0%	0%	0%	20%	20%	20%
with a Technician and min time:	0	0	3	0	0	0	0	6	6	6
with a Technician and mode time:	0	0	4	0	0	0	0	7	7	7
with a Technician and max time:	0	0	5	0	0	0	0	10	10	10

with a Medical Assistant:	0%	0%	5%	80%	0%	0%	0%	20%	20%	20%
with a Med Assist and min time:	0	0	3	3	0	0	0	6	6	6
with a Med Assist and mode time:	0	0	4	5	0	0	0	7	7	7
with a Med Assist and max time:	0	0	5	15	0	0	0	10	10	10
with a Nurse:	0%	0%	80%	20%	0%	70%	0%	30%	30%	30%
with a Nurse and min time:	0	0	5	3	0	3	0	5	5	6
with a Nurse and mode time:	0	0	8	4	0	5	0	6	6	7
with a Nurse and max time:	0	0	10	10	0	15	0	8	8	10
with a NP/PA:	0%	0%	10%	0%	0%	30%	0%	15%	15%	15%
with a NP/PA and min time:	0	0	8	0	0	3	0	5	5	6
with a NP/PA and mode time:	0	0	10	0	0	4	0	6	6	7
with a NP/PA and max time:	0	0	15	0	0	10	0	8	8	10
with a Physician:	0%	0%	0%	0%	0%	0%	0%	15%	15%	15%
with a Physician and min time:	0	0	0	0	0	0	0	3	3	6
with a Physician and mode time:	0	0	0	0	0	0	0	4	4	7
with a Physician and max time:	0	0	0	0	0	0	0	6	6	10
Average post-exam time (analytically)	0.0	0.0	3.8	5.8	0.0	7.1	0.0	1.3	1.3	6.1
Probability of having a **Exit Interview**:	0	0	1	1	0	1	0	1	1	1
with a Medical Assistant:	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
with a Med Assist and min time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and mode time:	0	0	0	0	0	0	0	0	0	0
with a Med Assist and max time:	0	0	0	0	0	0	0	0	0	0
with a Nurse:	0%	0%	50%	5%	0%	10%	0%	5%	5%	5%
with a Nurse and min time:	0	0	2	3	0	3	0	3	3	0
with a Nurse and mode time:	0	0	3	4	0	4	0	5	5	0
with a Nurse and max time:	0	0	5	5	0	5	0	8	8	0
with a NP/PA:	0%	0%	25%	10%	0%	10%	0%	5%	5%	5%
with a NP/PA and min time:	0	0	2	3	0	3	0	3	3	4
with a NP/PA and mode time:	0	0	3	4	0	4	0	5	5	6
with a NP/PA and max time:	0	0	5	5	0	5	0	8	8	9
with a Physician:	0%	0%	25%	85%	0%	80%	0%	90%	90%	90%
with a Physician and min time:	0	0	2	3	0	3	0	5	5	5
with a Physician and mode time:	0	0	3	5	0	5	0	7	7	9
with a Physician and max time:	0	0	5	8	0	8	0	10	10	15
Average exit interview time (analytically)	0.0	0.0	3.3	5.1	0.0	5.1	0.0	7.1	7.1	9.0
Probability of having a **Check-Out**:	1	1	1	1	1	1	1	1	1	1
with a clerical:	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
with a clerical and min time of:	2	2	2	4	4	3	2	4	4	2
with a clerical and mode time of:	3	3	3	5	5	4	4	5	5	4
with a clerical and max time of:	4	4	4	6	6	5	5	6	6	5
Average registration time (analytically)	3.0	3.0	3.0	5.0	5.0	4.0	3.7	5.0	5.0	3.7

	Cat 1a	Cat 1b	Cat 1c	Cat 2	Cat 2PV	Cat 3	Cat 3PV	Cat 4a	Cat 4b	Cat 5
Average Total Time:	**10.3**	**13.3**	**14.2**	**32.3**	**17.3**	**39.8**	**16.0**	**45.7**	**45.7**	**54.8**
Average after 1/3 to 2/3 rule:	6.8	8.4	9.4	16.8	11.8	18.6	10.4	23.5	23.5	25.8

Appendix B: Queston Simulation Output Files

File Name	Daily Output Measure Description
CLNC1	Overtime of the clinic
CLNC2	Number of companions
CLNC3	Number of patients scheduled into a physician and clinic time slots (physician appointments are patients treated by a physician sometime during the visit, whereas, a clinic appointment patients are treated by a medical staff members but not the physician)
CLNC4	Number of prescheduled patients in a physician and clinic slot
CLNC5	Number of no shows for both the physician or clinic scheduled patients
CLNC6	Number of walk-in patients taking a physician or clinic slot
CLNC7	Total number of patients admitted in the morning and in the afternoon
CLNC8	Number of clinic appointment patients admitted in the morning and in the afternoon
INFO1	Time the operators are busy during clinic hours only
IWA1	Maximum number of patients waiting in the internal waiting are
IWA2	Distribution of the number of patients waiting in the internal waiting area.
OFFC1_N	Total busy time for physician N
OFFC2_N	Total lunch time for physician N
OFFC3	Time distribution for the number of physicians busy
OFFC4	Elapsed time before the first patient of the day is treated by a physician
OP1_N	Total and fraction of the time operator N is busy
OP2_N	Total number of calls handled by operator N and total calls resulting in a scheduled patients
P1	Total number of walk-in patients
P2	Total number of patients admitted for each patient category
P3	Total LOS for each patient category
P3_1	ALOS for each patient category
P5	Total waiting time in the seat for each patient category
P5_1	Average waiting time in the seat for each patient category
P6	Total waiting time in the waiting room buffer for each patient category
P6_1	Average waiting time in the waiting room buffer for each patient category
P7	Total waiting time in an examination room for each patient category

P7_1	Average waiting time in an examination room for each patient category
P8	Total waiting time in the internal waiting room for each patient category
P8_1	Average waiting time in the internal waiting room for each patient category
P9	Total waiting time in the check-in room for each patient category
P10	Average waiting time in the check-in room for each patient category
P21_1	Average registration time for each patient category
P22	Total accumulated time in the check-in room for each patient category
P22_1	Average check-in time for each patient category
P23	Total accumulated time in the examination room for each patient category
P23_1	ALOS in the examination room for each patient category
P24	Total accumulated examination time for each patient category
P24_1	Average accumulated examination time for each patient category
P25	Total accumulated post-examination time for each patient category
P25_1	Average accumulated post-examination time for each patient category
P26	Total accumulated exit interview time for each patient category
P26_1	Average accumulated exit interview time for each patient category
P27	Total accumulated check-out time for each patient category
P27_1	Average accumulated check-out time for each patient category
P31	Total waiting time to get into the check-in room after registration for each patient category
P31_1	Average waiting time to get into the check-in room after registration for each patient category
QUEUE1	Total number of calls received by the Queston Information Center
QUEUE2	Total number of calls placed on hold (time >0 in the call queue)
QUEUE3	Total hold time for all calls placed on hold
QUEUE4	Average waiting time for all calls placed on hold
QUEUE5	Maximum waiting time for all calls placed on hold
QUEUE6	Time distribution of calls put on hold
REG1	Time distribution for the number of registration window busy.
REG1_N	Total and fraction of the time registration window N is busy
REG2_N	Number of registration and checkouts handled by registration window N
REG3_N	Distribution of time registration N is busy
RM1_1_N	Total and fraction of the time check-in room N is busy
RM1_2_N	Total and fraction of the time specialty room N is busy

RM1_3_N	Total and fraction of the time examination room N is busy
RM2_1_N	Number of patients handled by check-in room N
RM2_2_N	Number of patients handled by specialty room N
RM2_3_N	Number of patients handled by examination room N
RM3_1_N	Utilization of check-in room N
RM3_2_N	Utilization of specialty room N
RM3_3_N	Utilization of examination room N
RMAREA1	Time distribution for the number of check-in rooms busy.
RMAREA2	Time distribution for the number of specialty rooms busy.
RMAREA3	Time distribution for the number of examination rooms busy.
STF1_1_N	Total and fraction of the time medical assistant N is busy
STF1_2_N	Total and fraction of the time nurse N is busy
STF1_3_N	Total and fraction of the time physician assistant N is busy

Appendix C: Time Study Data

Appendix C.1 Christiansburg, Virginia Clinic Patient Visit Data from Day One

Day 1

Patient	Scheduled time	Actual Arrival Time	Departure time	Total Duration	Sex	Companion	Comments
1	8:15:00 AM	- data not collected -			F	0	
2	8:30:00 AM	CANCELLATION			F	0	
3	WALK-IN	- data not collected -			F	0	
4	8:45:00 AM	- data not collected -			M	0	
5	9:00:00 AM	8:52:00 AM	9:58:00 AM	1:06:00	F	0	
6	9:00:00 AM	8:52:00 AM	9:58:00 AM	1:06:00	M	0	accompanied prior patient
7	9:15:00 AM	9:11:00 AM	9:55:59 AM	0:44:59	F	0	child
8	WALK-IN	9:11:00 AM	9:55:59 AM	0:44:59	M	0	accompanied prior patient
9	WALK-IN	9:19:00 AM	10:04:06 AM	0:45:06	F	0	
10	9:30:00 AM	9:27:00 AM	10:22:15 AM	0:55:15	M	0	
11	9:45:00 AM	9:46:00 AM	10:30:45 AM	0:44:45	F	1	
12	10:00:00 AM	9:52:00 AM	10:33:43 AM	0:41:43	M	0	
13	10:15:00 AM	10:13:00 AM	11:12:05 AM	0:59:05	M	0	
14	10:30:00 AM	9:59:00 AM	11:04:22 AM	1:05:22	F	0	
15	WALK-IN	9:59:00 AM	11:04:22 AM	1:05:22	F	0	accompanying child treated
16	WALK-IN	10:44:00 AM	11:03:09 AM	0:19:09	F	0	
17	11:00:00 AM	NO-SHOW			F	0	
18	WALK-IN	11:11:00 AM	11:22:00 AM	0:11:00	M	0	seeking medication for wife
19	11:15:00 AM	11:10:00 AM	11:31:50 AM	0:21:50	M	1	baby
20	WALK-IN	11:24:00 AM	11:58:00 AM	0:34:00	F	0	
21	WALK-IN	11:38:00 AM	11:45:00 AM	0:07:00	F	1	billing question only
22	11:45:00 AM	11:54:00 AM	12:20:37 PM	0:26:37	M	1	
23	12:15:00 PM	12:05:00 PM	12:36:00 PM	0:31:00	M	0	
24	2:00:00 PM	1:54:00 PM	2:30:00 PM	0:36:00	F	2	baby
25	WALK-IN	1:59:00 PM	2:37:00 PM	0:38:00	F	0	
26	WALK-IN	2:00:00 PM	2:37:25 PM	0:37:25	F	0	
27	2:15:00 PM	1:57:00 PM	2:53:54 PM	0:56:54	F	0	
28	2:30:00 PM	2:19:00 PM	3:09:00 PM	0:50:00	F	2	baby
29	2:45:00 PM	2:22:00 PM	3:20:46 PM	0:58:46	M	1	child
30	3:00:00 PM	NO-SHOW			F	0	
31	3:15:00 PM	NO-SHOW			M	0	
32	3:30:00 PM	2:56:00 PM	4:18:15 PM	1:22:15	M	0	
33	WALK-IN	3:27:00 PM	3:41:36 PM	0:14:36	F	0	referral questions
34	WALK-IN	3:27:00 PM	4:03:02 PM	0:36:02	M	0	billing ques. and BP check
35	4:00:00 PM	3:32:00 PM	4:46:50 PM	1:14:50	F	3	
36	4:15:00 PM	4:05:00 PM	4:58:03 PM	0:53:03	F	0	

Appendix C.1 Christiansburg, Virginia Clinic Patient Visit Data from Day One (cont.)

Day 1 C - Clerical (Sharon), M - Medical Assistant (Pam), N - Nurse (Kim), P - Physician (Dr. Da

Patient	Regist.	Check-in	with	Pre-Exam	with	Exam	with	Post-Exam	with	Exit-Interv	with	Check-out	with
1													
2													
3													
4													
5	0	60	N	67	N	191	P	0		455	P	70	M
6	0	60	N	67	N	191	P	0		0		133	M
7	12	43	N	208	M/N/P	0		0		0		141	M
8	12	43	N	208	M/N/P	0		0		0		141	M
9	0	159	M	57	N	0		0		0		125	M
10	33	14	N	180	N	900	P	0		0		540	M
11	0	49	N	93	N	735	P	0		0		343	M
12	0	24	N	185	N	440	P	0		0		72	M
13	0	45	N	113	N	450	P	0		85	P	194	M
14	35	86	N	173	N	458	P	0		0		250	M
15	35	86	N	173	N	458	P	0		0		250	M
16	0	240	N	0		0		0		0		120	M
17													
18	0	0		435	P	0		0		0		190	M
19	0	173	N	0		410	P	0		0		67	M
20	240	60	N	90	N	600	P	0		0		143	M
21	360	0		0		0		0		0		0	
22	0	80	N	55	N	645	P	0		0		159	M
23	0	93	N	87	N	235	P	0		0		130	M
24	0	73	N	287	N	972	P	0		32	P	121	M
25	120	60	N	194	N	571	P	0		0		120	M
26	0	44	N	252	N	0		0		0		120	M
27	0	100	N	137	N	613	P	0		51	P	206	M
28	0	67	M	351	N	128	P	0		0		302	M
29	0	55	N	98	N	501	P	0		0		72	M
30													
31													
32	71	37	N	179	N	651	P	540	N	670	P	190	M
33	772	0		0		0		0		0		0	
34	754	0		133	M	0		0		0		0	
35	0	33	N	181	N	1188	P	0		0		27	M
36	0	70	N	87	N	411	P	0		0		279	M

Appendix C.2 Christiansburg, Virginia Clinic Patient Visit Data from Day Two

Day 2

Patient	Scheduled time	Actual Arrival Time	Departure time	Total Duration	Sex	Companion	Comments
1	8:30:00 AM	8:29:00 AM	9:01:20 AM	0:32:20	F	0	
2	9:00:00 AM	8:50:55 AM	9:31:18 AM	0:40:23	F	0	
3	8:45:00 AM	8:40:00 AM	9:34:15 AM	0:54:15	F	0	
4	9:15:00 AM	8:54:55 AM	9:55:20 AM	1:00:25	M	0	elderly
5	9:30:00 AM	9:23:00 AM	10:05:02 AM	0:42:02	F	0	
6	9:45:00 AM	9:49:02 AM	10:44:35 AM	0:55:33	M	0	
7	10:00:00 AM	10:07:18 AM	10:58:15 AM	0:50:57	F	0	
8	10:15:00 AM	10:25:50 AM	11:16:15 AM	0:50:25	M	0	
9	10:45:00 AM	10:46:00 AM	11:27:30 AM	0:41:30	M	1	
10	10:45:00 AM	10:34:12 AM	11:31:52 AM	0:57:40	F	0	
11	11:00:00 AM	11:00:38 AM	11:43:00 AM	0:42:22	F	1	
12	11:15:00 AM	11:01:53 AM	11:50:53 AM	0:49:00	M	0	
13	WALK-IN	10:59:48 AM	11:55:15 AM	0:55:27	F	0	

C - Clerical (Sharon), M - Medical Assistant (Pam), N - Nurse (Kim), P - Physician (Dr. Da

Patient	Regist.	Check-in	with	Pre-Exam	with	Exam	with	Post-Exam	with	Exit-Interv	with	Check-out	with
1	0	110	M	110	M	395	P	0		0		88	C
2	0	45	M	140	M	520	P	0		0		148	C
3	0	70	M	165	M	1085	P	0		0		849	C
4	0	35	M	125	M	837	P	0		82	P	36	C
5	0	20	M	155	M	595	P	0		0		77	C
6	0	39	M	271	M	780	P	480	M	0		221	C
7	0	42	M	147	M	710	P	190	M	0		290	C
8	0	40	M	168	M	600	P	0		210	P	171	C
9	0	48	M	670	M	0		0		0		120	C
10	0	49	M	281	M	565	P	0		0		243	C
11	0	63	M	450	M	0		0		0		344	C
12	0	0		265	M	650	P	0		0		637	C
13	0	61	M	223	M	407	P	0		0		120	C

Appendix D: Experimentation Results

See Table 7.1 for the Experiment Descriptions

Output Measures:		Experiments I	II	III	IV	V	VI	VII
Overtime of the Clinic	M	82.6	76.8	76.6	47.7	38.4	35.5	42.7
	T	37.2	49.8	47.6	17.8	12.4	14.2	13.6
	W	12.3	47.9	36.6	12.5	13.1	9.6	4.2
	H	43.5	68.1	65.2	42	38.1	36.2	44.7
	F	95.6	68.4	67.6	40.3	27.5	25.3	68.1
	Avg	54.3	62.2	58.7	32.1	25.9	24.2	34.7
Number of patients in the morning and in the afternoon		47/19	36/30	same				18/22
Number of clinic appointments in the am and in the pm.		?	?	12am/7p	same			6/4
Maximum number of patients in the internal waiting area		?	22	22	27	18	15	14
Max number of people in the waiting room buffer		31	14	9	4	2	0	0
								not available
Fraction of the time physician #1 is busy		0.73	0.70	0.71	0.71	0.72	0.72	0.70
								40.10
Total lunch time for physician #1		0.04	11.92	13.5	29.9	36.8	38	
								89.0
Distribution of the time 0, 1, or 2 physicians are busy	0	43.5	51.8	46.4	33.7	26.4	26.5	402.0
	1	209.2	199.2	200.0	158.6	146.1	138.8	na
	2	265.1	281.6	282.9	303.6	315.3	320.2	na
								na
Average patient length of stay for category:	1	64.9	48.9	48.1	24.0	24.4	24.7	27.1
	2	65.8	53.4	52.4	34.4	31.0	29.6	35.1
	3	65.4	43.2	45.9	33.7	28.0	30.6	24.1
	4	142.9	121.7	118.5	85.3	76.9	75.9	94.6
	4P	62.4	49.5	48.3	32.6	33.1	27.3	30.5
	5	151.6	131.6	127.0	98.8	79.3	76.9	101.2
	5P	55.1	44.9	44.3	31.7	33.0	27.0	22.6
	6	142.2	120.8	110.0	81.7	70.6	73.1	46.0
	7	165.8	153.4	142.8	101.2	88.4	86.8	93.5
	8	179.3	158.4	152.5	117.7	107.1	105.4	110.9
	Avg	133.0	114.2	110.1	80.5	69.8	68.6	81.6
Distribution of time 0,1,or 2 check-in rooms are busy	0	65.6	86.6	84.6	202.3	242.1	277.1	312.4
	1	61.8	53.8	56.7	116.6	112.6	102.3	91.0
	2	275.0	280.7	279.6	85.3	48.9	23.5	na
Distribution of time 0 or 1 specialty room is busy	0	293.3	299.0	297.3	276.8	264.4	261.4	na
	1	31.9	30.7	30.8	29.3	29.3	29.3	na
Distribution of time 0, 1, 2, 3, 4, 5, or 6 exam rooms are	0	9.1	23.9	25.9	46.4	55.4	58.7	257.4
	1	29.5	33.5	34.3	41.6	45.0	45.3	16.5
	2	25.9	34.2	34.0	39.0	46.5	45.4	
	3	33.2	41.0	36.6	39.9	45.6	48.5	94.1
	4	39.9	49.5	44.9	41.1	46.2	46.8	68.9
	5	60.4	63.1	60.4	51.8	51.1	51.8	70.0
	6	319.9	288.2	292.6	230.1	192.1	183.4	253.2
								na
Distribution of time 0, 1, 2, ... nurses are busy	0	130.9	141.7	138.4	139.9	172.4	186.9	na
	1	133.9	142.6	137.2	87.9	105.3	128.7	na
	2	209.4	209.7	212.2	221.3	165.3	124.7	
	3							270.2
								182.6

Output Measures:		Experiments						
		I	II	III	IV	V	VI	VII
Distribution of time 0, 1, 2, ... medical assistants are b	0	129.2	151.7	152.5	170.4	189.3	199.7	na
	1	164.9	169.9	159.7	111.9	125.9	130.8	na
	2	107.4	104.7	109.1	79.2	71.4	75.5	na
	3	60.0	58.9	59.4	86.2	50.4	35.1	
								292.6
Distribution of time 0, 1, 2, ... physician assistants are	0	183.1	203.0	200.5	195.2	194.3	166.1	123.0
	1	173.2	170.3	167.4	121.0	115.2	89.7	36.3
	2	107.1	110.8	112.5	127.0	124.3	182.1	na
	3							na
Fraction of the time medical assistant #... is busy	1					0.274	0.254	198.9
	2					0.279	0.257	258.0
	3					0.281	0.260	na
								na
Fraction of the time nurse #... is busy	1					0.425	0.377	na
	2					0.433	0.378	
								0.2
Fraction of the time physician assistant #... is busy	1					0.359	0.450	0.2
	2					0.363	0.447	na
Average Waiting Time in the Clinic	1	54.2	38.2	37.4	17.1	17.5	17.8	20.2
	2	52.1	39.7	38.7	25.8	22.4	21.0	26.5
	3	51.3	29.1	31.8	24.3	18.6	21.2	14.7
	4	107.0	85.8	82.6	67.3	58.9	57.9	76.6
	4P	43.7	30.8	29.6	20.4	20.9	15.1	18.3
	5	105.9	85.9	81.3	78.2	58.7	56.3	80.6
	5P	37.7	27.5	26.9	20.8	22.1	16.1	11.7
	6	82.3	60.9	50.1	53.5	42.4	44.9	17.8
	7	105.9	93.5	82.9	73.0	60.2	58.6	65.3
	8	110.0	89.1	83.2	87.0	76.4	74.7	80.2
	Avg	94.0	75.2	71.1	61.8	51.1	49.9	62.9
Average Waiting Time only in the Internal Waiting Area	1					0.0	0.0	0.0
	2					1.6	1.4	4.1
	3					4.4	3.2	4.7
	4					17.9	16.7	28.6
	4P					2.4	2.0	3.4
	5					17.9	17.2	35.6
	5P					1.6	1.3	3.1
	6					6.7	6.2	5.5
	7					18.2	17.2	28.1
	8					14.2	14.3	28.8
	Avg					13.5	12.8	23.8
Average Waiting Time Only in the Exam Room	1					0.0	0.0	0.0
	2					1.9	0.9	0.8
	3					3.5	6.5	2.7
	4					26.8	27.2	30.1
	4P					1.4	0.7	0.6
	5					25.9	24.2	27.5
	5P					1.8	0.4	0.4
	6					27.1	30.5	19.2
	7					28.0	27.0	23.2
	8					51.4	50.1	39.1
	Avg					23.0	22.6	22.6

Appendix D: Experimentation Results (continued)

See Table 7.1 for the Experiment Descriptions

Output Measures:		Experiments							
		VIII	IX	X	XI	XII	XIII	XIV	XV
Overtime of the Clinic	M	39.1	68.9	63.3	36.9	14.9	4.2	4.6	3.2
	T	39.9	77.1	68.5	25.2	12.5	0.0	0.0	0.0
	W	25.2	33.3	34.0	16.7	9.9	4.5	6.0	4.4
	H	51.6	43.4	75.6	17.1	10.2	0.0	0.0	0.0
	F	31.1	67.2	92.8	25.7	3.3	8.6	10.6	8.8
	Total	37.4	58.0	66.8	24.3	10.2	3.5	4.2	3.3
Number of patients in the am and in the pm		68/59	28/39	27/40	27/39	27/40	29/37	29/37	29/37
Number of clinic appointments in the am and in the pm		23/14	9/10	9/10	9/10	10/12	10/11	10/11	10/11
Maximum number of patients in the internal waiting a		22	27	24	26	16		17	15
Max number of people in the waiting room buffer		44	0	0	1	7	7	4	1
Max number of seats taken				24		17	18	17	
Fraction of the time physician #1 is busy		0.68	0.68	0.70	0.60	0.54	0.56	0.56	0.56
Total lunch time for physician #1		32.00	52.58	48.16	55.78	56.84	51.64	51.40	51.48
Distribution of the time 0, 1, or 2 physicians are	0	16.6	56.2	44.3	66.9	79.0	64.9	63.7	64.3
	1	53.6	146.9	147.3	145.3	147.8	138.1	144.2	139.0
	2	80.5	332.8	346.4	271.1	237.3	238.4	235.2	237.8
	3	160.3	na	na	na	na	na	na	na
	4	201.0	na	na	na	na	na	na	na
Average patient length of stay for category:	1	56.9	21.5	20.1	25.3	26.4	25.6	24.5	25.5
	2	54.7	25.4	24.4	27.9	26.2	25.9	25.9	26.3
	3	60.0	34.0	31.6	30.5	30.2	31.9	35.1	33.2
	4	92.2	85.4	97.8	66.8	62.6	61.7	62.2	60.2
	4P	49.3	26.1	27.7	28.6	26.2	27.5	27.8	27.8
	5	87.5	90.8	109.8	69.2	62.9	60.3	64.9	60.8
	5P	48.3	21.3	21.8	23.3	20.5	21.5	22.3	21.7
	6	76.4	64.2	78.7	53.4	49.3	48.5	48.4	46.5
	7	104.3	100.9	115.8	74.2	65.5	65.6	68.8	65.0
	8	115.3	115.9	132.7	94.3	88.5	82.2	85.2	83.6
	Avg	85.1	76.3	88.2	61.0	56.5	55.0	56.8	54.7
Distribution of time 0,1,or 2 check-in rooms are	0	217.6	285.7	287.6	301.2	297.7	295.4	284.7	292.8
	1	136.0	102.9	101.6	96.8	97.5	96.7	102.3	98.6
	2	51.4	23.8	22.5	17.3	17.1	17.1	22.2	18.2
Distribution of time 0 or 1 specialty room is bus	0	322.8	278.4	302.8	266.0	270.7	230.4	231.1	230.4
	1	55.2	30.7	29.3	22.5	15.8	16.2	16.2	16.2
Distribution of time 0, 1, 2, 3, 4, 5, or 6 exam ro	0	43.8	98.6	83.6	112.0	122.6	106.8	105.3	76.5
	1	32.2	46.1	38.2	38.0	43.4	42.6	37.7	(2) 49.7
	2	(4) 31.6	43.4	34.4	46.4	47.7	41.2	39.2	(4) 50.8
	3	(7) 41.3	44.5	39.6	49.7	50.6	46.4	41.6	(6) 52.2
	4	(9) 29.8	42.4	37.2	47.0	44.4	47.7	44.6	(8) 57.7
	5	(11) 34.2	48.0	50.2	49.4	46.3	47.1	50.1	(10) 62.9
	6	(12) 95.2	208.0	253.8	141.4	109.0	110.3	124.9	21.2
Distribution of time 0, 1, 2, ... nurses are busy	0	130.4	235.2	240.4	224.1	214.9		169.4	191.6
	1	92.7	141.2	141.7	125.5	118.3		100.3	110.6
	2	90.7	118.5	118.5	104.5	104.4		157.3	123.5
	3	79.7	na		na	na	na	na	na
	4	81.8	na		na	na	na	na	na

Output Measures:		Experiments							
		VIII	IX	X	XI	XII	XIII	XIV	XV
Distribution of time 0, 1, 2, ... medical assistan	0	132.2	241.1	243.1	214.0	208.3		234.6	191.6
	1	118.3	137.9	140.1	131.5	125.6		188.3	110.6
	2	104.4	74.2	75.0	74.8	75.5		na	123.5
	3	72.4	34.5	31.5	29.8	29.1		na	na
	4	40.5	na		na	na	na	na	na
Distribution of time 0, 1, 2, ... physician assista	0	134.4	211.3	216.9	223.3	219.1		174.9	212.0
	1	79.3	97.4	106.9	95.3	84.4		62.9	127.5
	2	68.6	173.5	169.0	133.5	131.0		178.7	84.9
	3	57.0	na		na	na	na	na	na
	4	115.1	na		na	na	na	na	na
Fraction of the time medical assistant #... is bus	1								
	2								
	3								
Average Waiting Time in the Clinic	1	50.0	14.6	13.2	18.6	19.6	18.8	17.8	18.8
	2	46.2	16.9	15.8	19.5	17.7	17.5	17.4	17.9
	3	50.6	24.6	22.2	21.1	20.8	22.5	25.7	23.8
	4	74.2	67.4	79.8	50.1	45.8	44.9	45.4	43.4
	4P	37.0	13.9	15.5	16.8	14.4	15.7	16.1	16.0
	5	66.9	70.2	89.2	50.6	44.3	41.8	46.4	42.2
	5P	37.3	10.4	10.9	12.9	10.0	11.1	11.8	11.3
	6	48.2	36.0	50.5	29.9	25.9	25.1	24.9	23.0
	7	76.1	72.7	87.6	50.7	42.0	42.2	45.4	41.6
	8	84.6	85.5	102.1	68.5	62.7	56.4	59.4	57.8
	Avg	66.4	57.6	69.5	44.2	39.7	38.1	40.0	37.9
	8	84.6	85.2	102.1	68.5	62.7	56.4	59.4	57.8
Average Waiting Time only in the Internal Waitin	1	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0
	2	0.5	2.0	2.8	1.3	0.3	0.6	0.8	0.5
	3	1.8	8.4	8.2	5.4	2.3	2.4	3.1	2.4
	4	8.2	25.3	35.6	12.9	8.1	6.2	8.1	6.0
	4P	0.5	3.2	6.0	1.7	1.1	1.3	1.6	1.2
	5	7.9	34.0	48.1	15.1	9.4	7.4	9.5	6.7
	5P	0.1	0.6	1.7	0.4	0.2	0.5	0.9	0.7
	6	2.0	5.9	11.2	3.6	2.3	1.3	2.0	1.3
	7	7.5	29.2	40.9	13.8	7.7	6.9	8.2	6.1
	8	6.0	22.4	32.3	9.4	6.6	5.0	6.3	4.5
	Avg	5.9	21.8	30.9	10.3	6.4	5.1	6.5	4.7
Average Waiting Time Only in the Exam Room	1	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0
	2	0.4	0.6	0.6	0.5	0.3	0.5	0.8	0.6
	3	8.1	8.3	7.2	4.3	3.4	5.1	8.9	6.2
	4	27.5	31.3	26.3	16.9	15.6	16.7	16.0	16.0
	4P	0.4	0.5	0.5	0.5	0.4	0.4	1.0	0.5
	5	19.2	24.2	23.7	16.3	14.3	14.2	17.2	15.1
	5P	0.5	0.3	0.5	0.4	0.4	0.3	1.0	0.4
	6	22.0	25.4	21.6	9.8	9.4	11.5	11.3	10.1
	7	29.0	31.5	23.4	15.6	12.7	13.5	15.8	14.1
	8	50.6	51.4	36.3	29.0	23.7	20.8	23.2	23.1
	Avg	21.718	24.7	20.9	14.1	12.4	12.8	14.1	13.1

Vita

Jong (Brian) Jun was born in Seoul, South Korea to Tuk Ki and Yong Ja Jun on June 1, 1967. Brian received his Bachelor of Science degree in Aerospace Engineering from Virginia Polytechnic Institute and State University in 1990. In May, 1999 he will receive his Master of Science in Industrial and Systems Engineering with a concentration in Operations Research from Virginia Polytechnic Institute and State University. While completing his degree, Brian worked both as an operations research analyst and as a management engineer at Biological & Popular Culture, Inc., and as an officer at First USA Bank. He is a member of Alpha Phi Mu, Alpha Phi Omega, IIE, and INFORMS.