Handbook of Research on Advances in Health Informatics and Electronic Healthcare Applications: Global Adoption and Impact of Information Communication Technologies

Khalil Khoumbati
University of Sindh, Pakistan

Yogesh K. Dwivedi
Swansea University, UK

Aradhana Srivastava
Participatory Research in Asia (PRIA), India

Banita Lal
Nottingham Trent University, UK

Medical Information Science Reference

MEDICAL INFORMATION SCIENCE REFERENCE

Hershey · New York

Director of Editorial Content: Kristin Klinger
Senior Managing Editor: Jamie Snavely
Assistant Managing Editor: Michael Brehm
Publishing Assistant: Sean Woznicki
Typesetter: Michael Brehm
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Medical Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com/reference

Library of Congress Cataloging-in-Publication Data

Handbook of research on advances in health informatics and electronic healthcare applications : global adoption and impact of information communication technologies / Khalil Khoumbati ... [et al.], editors.
p. ; cm.
Includes bibliographical references and index.
Summary: "This book presents a comprehensive resource elucidating the adoption and usage of health informatics"--Provided by publisher.
ISBN 978-1-60566-030-1 (h/c)
1. Medical informatics--Handbooks, manuals, etc. I. Khoumbati, Khalil, 1965-
[DNLM: 1. Medical Informatics--trends. W 26.5 H2353 2010]
R858.H322 2010
651.5'04261--dc22
2009001900

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 16
Speeding Up Decision Support:
Investigating the Distributed Simulation of a Healthcare Supply Chain

Navonil Mustafee
Warwick Business School, UK

Simon J.E. Taylor
Brunel University, UK

Korina Katsaliaki
Middlesex University, UK

Sally Brailsford
University of Southampton, UK

ABSTRACT

Discrete-Event Simulation (DES) is a decision support technique that allows stakeholders to conduct experiments with models that represent real-world systems of interest. Its use in healthcare is comparatively new. Healthcare needs have grown and healthcare organisations become larger, more complex and more costly. There has never been a greater need for carefully informed decisions and policy. DES is valuable as it can provide evidence of how to cope with these complex health problems. However, the size of a healthcare system can lead to large models that can take an extremely long time to simulate. In this chapter the authors investigate how a technique called distributed simulation allows us to use multiple computers to speed up this simulation. Based on a case study of the UK National Blood Service they demonstrate the effectiveness of this technique and argue that it is a vital technique in healthcare informatics with respect to supporting decision making in large healthcare systems.

INTRODUCTION

Computer simulation, or just simulation, is a decision support technique that allows stakeholders to conduct experiments with models that represent real-world systems of interest (Pidd, 2004a). It has been widely used for many years in domains such as manufacturing, logistics and telecommunication.. However, its use in healthcare is comparatively new. It is only really during the last decade that the application of simulation in health care has grown

DOI: 10.4018/978-1-60566-030-1.ch016

substantially (Fone et al, 2003). Healthcare needs have also grown in the same period and healthcare organizations become larger, more complex and more costly. There has never been a greater need for carefully informed decisions and policy. Computer simulation is valuable as it can provide evidence of how to cope with these complex health problems. It can be used as an alternative to "learning by doing" or empirical research (Royston, 1999). Furthermore, if carried out correctly, simulation modelling gives stakeholders the opportunity to participate in model develop and, hopefully, gain deeper understanding of the problems that they face. As a result, decision-makers and stakeholders can gain a new perspective on the relationships between the available resources, the level of the system's performance and the overall quality of the healthcare provision.

Many successful studies have been reported using simulation to address health care system problems (Jun et al, 1999; Cooper et al 2007). Of these, four simulation approaches have been used. These are Monte-Carlo Simulation, Agent-Based Simulation, System Dynamics and Discrete-Event Simulation. Monte-Carlo Simulation has its roots in World War Two and is a simulation technique that uses a sequence of random numbers according to probabilities assumed to be associated with a source of uncertainty, for example, stock prices, interest rates, commodity prices, etc. (Rubinstein, 1981). In healthcare, Monte-Carlo Simulation has been used to evaluate the cost-effectiveness of competing technologies or healthcare strategies that require the description of patient pathways over extended time horizons. It is the main approach to modelling used in economic evaluations in health care interventions when there is a need to increase the number of states in the model to overcome the homogeneity assumptions inherent in Markov models and decision trees (Barton et al., 2004). Agent-Based Simulation is a computational technique for modelling the actions and interactions of autonomous individuals in a network, called agents, with a view to assessing their effects on the system as a whole but not independently. It is a technique used since the mid-1990s to solve a variety of financial, business and technology problems. Its application in the healthcare sector is not yet widespread but is has been used to study problems such as the spread of epidemics (Bagni et al., 2002). System Dynamics comes from Industrial Engineering in the 1950's and is a modeling approach that takes a holistic view of the problem. In healthcare, Systems Dynamics is used to model health systems from a more integrated or top-level approach. This simulation technique can assist the design of healthcare policies by examining how the fundamental structure might influence the progressive behaviour of a system. It takes into consideration factors such as the time variation both of tangible elements, such as waiting times and health care costs, as well as intangible, such as patient anxiety and the effects of various pressures on purchasing decisions (Taylor and Lane, 1998). In Discrete-Event Simulation, a technique that emerged in the UK in the late 1950's, systems are modeled in greater detail than with Systems Dynamics and with more complex temporal dependencies than with Monte-Carlo Simulation. It involves the modelling of a system as it progresses through time and is particularly useful for modelling queuing systems (Robinson, 1994). Discrete-Event Simulation is therefore particularly well-suited to tackle problems in healthcare where, for example, resources are scarce and patients arrive at irregular times (for example in A&E departments). Some of the applications of Discrete Event Simulation is therefore to forecast the impact of changes in patient flow, to examine resource needs (either in physical capacity of beds and equipment or in staffing), to manage patient scheduling and admissions or to investigate the complex relationships among the different model variables (for example, rate of arrivals or time spent in the system). Discrete-Event Simulation therefore allows decision makers (namely, health policy makers, administrators and hospital managers) to effectively assess the efficiency of existing

health care delivery systems, to improve system performance or design, and to plan new ones. An extensive taxonomy of discrete event simulation studies in healthcare over the past twenty years is presented in Jun et al. (1999) and Fone et al. (2003). It has been shown that Discrete-Event Simulation can create significantly more insight than Monte-Carlo Simulation in areas such as health economics (Eldabi et al, 2000).

To recap, healthcare systems are becoming larger and more complex. Of our computer simulation approaches, Discrete-Event Simulation (DES) is the most suitable technique to capture these complexities for to support meaningful decision making as it allows appropriate levels of detail and dynamic, stochastic behaviour to captured. However, a knock-on effect of this is that DES models are becoming larger, more complex, and significantly more computationally demanding. In some cases, the time taken to run the simulation of a single scenario can take over a day. These long run-times can make the whole simulation project untenable. However, research has shown that it is possible to reduce these run-times by using a technique called *distributed simulation* that utilises multiple computers to share the processing load. This is, however, not at all easy! This chapter therefore reports on this recent, exciting advance in healthcare informatics that could widely benefit many large-scale modelling projects and, ultimately, broaden the application of this technique.

The chapter is structured as follows. In section 2 we present more details on DES. Section 3 gives an overview of *distributed simulation*, the approach to sharing the processing load of large models over several computers. Section 4 reports on our case study, the National Blood Service simulation. Section 5 presents some results from our case study that compares our work on the performance of the simulation running a single computer system against a simulation running on multiple computers using distributed simulation techniques. Section 6 discusses the implications of these results. Section 7 discusses some important future trends emerging from our demonstration of the benefits of distributed simulation. Section 8 draws the chapter to a close.

DISCRETE EVENT SIMULATION

A Discrete-Event Simulation (DES) model encompasses a number of important concepts, namely entities, state, events and logical relationships, which define the overall behaviour of the abstract representation of the real system being studied. Entities are the elements of the system that can be individually identified and processed (e.g., patients, orders, documents, etc.). These "flow" through the system requiring resources (e.g., nurses, beds, etc.) in order to perform activities (e.g., prognoses, operations, etc.). Waiting lines (queues) (e.g., reception areas, waiting rooms, clinics, etc.) are where entities wait for needed resources to become available or for events to take place (e.g., hospital admissions, doctor assessment, etc.). Logical relationships link the different entities together and make them behave in a certain way. While a complete introduction to DES is outside the scope of this paper, excellent background literature includes Pidd (, 2004a), Law (2007) and many papers from the Winter Simulation Conference series (www.wintersim.org).

The process of building DES models usually involves some form of software. The software can either be a high level programming language or a data-driven software system in which the model is specified using user-defined and default data items. These computer packages are described as Visual Interactive Modelling Systems (VIMS) (Pidd, 2004a), or Simulators (Law, 2007). Most of these have evolved into "black box" or "shrink wrapped" software with user interfaces that are familiar to users of Microsoft Windows™ packages such as Office™. We therefore refer to these as Commercial Off-The-Shelf (COTS) Simulation Packages (CSPs). Examples include AnyLogic™

(XJ Technologies), Arena™ (Rockwell Automation), Flexsim™ (Flexsim Software Products, Inc), Simul8™ (Simul8 Corporation) and Witness™ (Lanner group). Moreover, some CSPs are specifically aimed at the health care industry, such as MedModel™ (ProModel Corporation) and Arena™ (Rockwell Automation) with a health care template.

The CSPs allow the users to create and experiment with simulations visually and interactively. They are easy and intuitive to use and are popular among modellers. They enable relatively unskilled users to develop useful simulations in a short period of time without the need for detailed computer programming (Pidd and Cassel, 2000). Furthermore, the introduction of visually oriented graphical outputs not only aids in the verification and validation of models and results (Gipps, 1986), but also in the communication of results to decision makers. In addition, once the model is set and enhanced with user-friendly mechanisms for running different scenarios, these systems can be easily be reused by end-users alone. The number of health care organisations and government agencies using these advanced simulation packages has grown (Jun et al., 1999).

These CSPs, although suitable for most systems that are modelled in industry (which includes the healthcare sector), may lack the capability to simulate large and complex models (Pidd, 2004b). Moreover, "there remain systems that cannot be sensibly simulated in this way, either because the application logic is too detailed or obscure for the simulators, or for other reasons such as the need to run an extremely fast or large simulation" (Pidd and Cassel, 2000). Therefore, there are still occasions that may justify the development of simulation programs from scratch using general-purpose programming languages (such as Java or C++). However, since these bespoke simulation programs are usually tailor-made to investigate specific problems (i.e. lack flexibility) and because they usually lack in visual interaction and animation capabilities (i.e. CSPs have evolved over the years to give quite sophisticated capabilities that bespoke solutions typically cannot afford), they may not appeal to healthcare managers.

Nevertheless, as healthcare systems become more complex to manage there is a greater demand for quantifiable evidence to assist decision making. Within the healthcare sector, only a limited number of simulation models have been developed to analyse complex multi-facility healthcare delivery systems (Jun et al., 1999). Most simulation models report on individual or local in scale units of healthcare facilities or problems in general. There is an emerging need for the development of more powerful high speed distributed simulations which will could facilitate the creation of complex, but tractable, models of large integrated systems, with the results implemented more easily and frequently (Baezner et al., 1990). This book chapter therefore focuses on an approach to *distributed simulation* that is making the high speed simulation of large and complex healthcare systems using CSPs more possible.

Distributed simulation refers to the execution of a DES comprising two or more models, each of which runs on a separate processor or computer (the distinction is made as some computers have multiple processors). However, the CSPs do not have inbuilt support for distributed simulation. *The objective of this chapter is therefore to propose and demonstrate a solution which enables the execution of large and complex healthcare models, developed in CSPs via the use of distributed simulation.* We demonstrate via a case study how this approach has been used to support the National Blood Service (NBS) supply chain simulation in the Southampton area of the UK. This is arguably the first attempt to create a distributed simulation in healthcare. This research assumes added significance because a CSP (Simul8™ from Simul8 Corporation) has been used to create and execute the NBS models. This is unlike the majority of distributed simulations, in domains such as the military, where the models themselves are coded using a general purpose programming language.

We now introduce the technique of distributed simulation.

DISTRIBUTED SIMULATION

This section will outline the motivations of using distributed simulation and highlight its application areas. It will discuss the theory behind it and present an overview of different middleware being used to executing such simulations, in particular the High Level Architecture (HLA), which is increasingly becoming the *de facto* standard for distributed simulation.

Definition

In a distributed simulation, a large computer model is executed over several processors. These processors can be a part of a multiprocessor computer or may belong to multiple PCs that are connected over a network. Parallel Discrete Event Simulation (PDES) usually refers to the execution of such distributed DES on parallel and distributed machines (Page and Nance, 1994).

In the context of PDES, Fujimoto (2001) distinguishes between parallel and distributed simulation based on the frequency of interactions between processors during the simulation execution. A parallel simulation is defined as running a simulation on a tightly coupled computer with multiple central processing units (CPUs) where the communication between the CPUs can be very frequent (e.g., thousands of times per second). A distributed simulation, on the other hand, is defined as executing simulations on multiple processors over loosely coupled systems (e.g., a network of PCs) where the interactions take more time (e.g., milliseconds or more) and occur less often. Sometimes the terms parallel simulation and distributed simulation are used interchangeably (Reynolds, 1988). Fujimoto (2003) uses the term distributed simulation to refer to both the parallel and distributed variants of PDES. The rationale presented is that, although historically, the terms "distributed simulation" and "parallel simulation" referred to geographically distributed simulations and simulations on tightly coupled parallel computers respectively, new distributed computing paradigms like clusters of workstations and grid computing has made this distinction less obvious. This research takes a similar view and therefore does not distinguish between the parallel and distributed variants of PDES. We will therefore use "*distributed simulation*" to refer to the execution of simulations on both multiprocessor machines and over a network of PCs.

Motivations

Some of the reasons for using distributed simulation are as follows (Fujimoto, 1999a; Fujimoto, 2003).

- Distributed simulation can facilitate model reuse by "hooking together" existing simulations into a single simulation environment. It is usually far more economical to link existing simulations to create distributed simulation environments than to create new models within the context of a single tool or piece of software.
- A large simulation may have memory and processing requirements that cannot be provided by a single system. Distributing the simulation execution across multiple machines may allow the memory and processors of many computer systems to be utilized. Thus, distributed simulation may enable large simulations to be executed that could not be executed on a single computer. This may also lead to the distributed model running faster than the single "standalone" alternative.
- Executing simulations on a set of geographically distributed computers facilitates wider user participation in the simulation experiments. This also alleviates the

cost and time that is normally associated with bringing participants to one physical place in, for example a joint training exercise or decision making in a supply chain.

Application Areas

The current and potential application areas for distributed simulation are presented in Table 1 (Fujimoto, 1999b).

Although the table lists only some of the application areas of distributed simulation, the fact that CSP-based simulation has not been identified as either a current or potential distributed simulation application area may seem to suggest that there is very little work done in the area of CSP-based distributed simulation.

Distributed Simulation Theory

A simulation has to process events in increasing timestamp order (the "timestamp" of an event is the time at which the event is scheduled to occur). Failure to do so will result in *causality errors*. A causality error occurs when a simulation has processed an event with timestamp T1 and subsequently receives another event with timestamp T2, wherein T1 > T2. Since the execution of the event with timestamp T1 may have changed the state variables that will be used by the event with timestamp T2, this would amount to simulating a system in which the future could affect the past (Fujimoto, 1990). For a serial simulator that has only one event list and one logical clock it is trivial to avoid causality errors. In the case of distributed simulation, the avoidance of causality is a lot more difficult because it has to deal with multiple event lists and multiple logical clocks that are assigned to various processors. The reason for this is explained below.

The system being modelled (e.g., a hospital) may be composed of a number of physical processes (e.g., clinics and operating theatres within the hospital). In a distributed simulation, each physical process is usually mapped to a logical simulation process running on a separate machine. All the interactions between the physical processes (e.g., transfer of patients from clinic to the operation theatre) are modelled as messages that are exchanged between their corresponding logical

Table 1. Application areas of distributed simulation

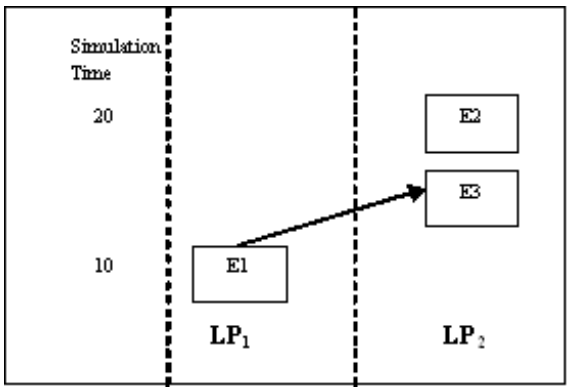
Applications	Type of Simulation
Military applications	Analytical war game simulations are performed to evaluate different strategies for war. These simulations are typically composed of individual models that represent different military divisions and use algorithms for synchronisation of the models (discussed in section 3.3). Another application of distributed simulation in the military is for training, and Test and Evaluation (T&E). These are conducted in Distributed Virtual Environments (DVE) where both humans (human-in-the-loop) and devices (hardware-in-the-loop) take part in the simulation.
Telecommunication networks	Analytical PDES have been used widely to evaluate networking hardware, software, protocols and services in the telecommunication industry.
Social interactions and business collaborations	Distributed Virtual Environments allow people to interact socially and to develop business collaborations on the Internet. Note: This was identified as a potential application area of distributed simulation in 1999, but today it has become a reality with popular Internet-based 3-D social networks like Second Life (Linden Research, 2007).
Medical application (potential area)	Computer generated virtual environments have been created both for doctors (to practice surgical techniques) and for patients (to treat various phobias). However, most of this work is currently limited to non-distributed virtual environments.
Transportation (potential area)	PDES (multiprocessors) can reduce the time taken to experiment with different strategies for responding to unexpected events like congestion resulting from weather conditions, etc. This will help take decisions faster.

processes. Each message will have a timestamp associated with it.

In figure 1, the simulation represents a physical system that has two physical processes, say, PP1 and PP2. Logical simulation processes LP1 and LP2 model the two physical processes. The logical processes have their own simulation executive, simulation clock and an event list. During simulation initialisation the event lists of both LP1 and LP2 are populated with the events E1 and E2 respectively. The timestamps for E1 and E2 are 10 and 20 respectively. It will be possible for LP1 to process event E1 without any causality error since the timestamp of E1 is less than the timestamp of E2. However, LP2 will not be able to execute event E2 at time 20 because causality error may then occur. The reason for this is that execution of E1 might schedule another event E3 for LP2 at time 15. In such a case, if LP2 had been allowed to execute E2 at simulated time 20 then it would have resulted in a causality error because the timestamp of E3 is less than the timestamp of E2. Different synchronisation protocols exist for distributed simulation that prevent or correct such causality errors.

Synchronisation protocols are one of the most important research areas of distributed simulation. They can be broadly divided into conservative (pessimistic) protocols and optimistic protocols.

Figure 1. Execution of events in a distributed simulation (Fujimoto, 1990)



In a conservative protocol a processor is never allowed to process an event out of order; whereas in an optimistic protocol a processor is allowed to process an event out of order, provided it can revert back to its previous state in the case of a causality error (Nicol and Heidelberger, 1996). Chandy and Misra (1979) created one of the first pessimistic approaches that implements the conservative synchronisation protocol. An optimistic synchronisation protocol like Virtual Time, and its implementation called the Time Warp mechanism, executes events without considering the event time ordering (Jefferson, 1985). It has to save its state frequently so that a rollback to a previous state can occur when an event with a time stamp less than the current simulation time is received. Today, synchronisation protocols are usually implemented through supporting software termed distributed simulation middleware. These are discussed next.

Distributed Simulation Middleware

A distributed simulation middleware is a software component that implements the distributed simulation algorithms to achieve synchronisation between individual running simulations (or LPs). Middleware such as HLA-RTI (IEEE 1516, 2000), FAMAS (Boer, 2005), GRIDS (Taylor et al., 2002) and CSPE-CMB (Mustafee, 2004), can be used to facilitate distributed execution of CSP-based simulations. Distributed simulation protocols such as Aggregate Level Simulation Protocol (ALSP) (Fischer et al., 1994) and Distributed Interactive Simulation (DIS) (Miller and Thorpe, 1995) have been used widely in defence training simulations. However, there has been no reported application of these technologies to CSP-based simulations. As such they fall outside the scope of this book chapter.

The our healthcare case study uses the HLA-RTI middleware to couple together models created using the CSP Simul8™ and executed on different computers. As such, the next section of

this chapter discusses the HLA-RTI middleware for distributed simulation.

The High Level Architecture

The High Level Architecture (HLA) (IEEE 1516, 2000) was originally proposed to support distributed simulation between existing and new simulations within the U.S Department of Defense (DoD). This came from the need to reduce the cost of training military personnel by reusing computer simulations linked via a network. HLA is now a IEEE standard. In the HLA, a distributed simulation is called a *federation*, and each individual simulation is referred to as a *federate*. A HLA Runtime Infrastructure (HLA-RTI) is a distributed simulation middleware, conforming to the HLA standards, that provides facilities to enable federates to interact with one another, as well as to control and manage the simulation.

The HLA is composed of four parts: a set of compliance rules, the Federate Interface Specification (FIS), the Object Model Template (OMT), and the Federate Development Process (FEDEP). The rules are a set of ten basic conventions that define the responsibilities of both federates and the federation in the context of their relationship with the HLA-RTI. The FIS is an application interface standard which defines how federates interact within the federation. The FIS standard is implemented by the HLA-RTI. The HLA-RTI, thus, forms a base into which existing simulations (federates) can be "plugged into" to form a large distributed simulation (Fujimoto and Weatherly, 1996). There are several implementations of HLA-RTI available, for example, DMSO HLA-RTI and Pitch pRTI (Karlsson and Olsson, 2001). The OMT provides a common presentation format for HLA federates. FEDEP defines the recommended practice processes and procedures that should be followed by users of the HLA to develop and execute their federations.

For models created using CSPs to interoperate using the HLA standard, some of the FIS-defined interfaces have to be implemented. The FIS organises the communication between federates and the HLA-RTI into six different management groups. These are:

- **Federation management:** HLA-RTI calls for the creation and deletion of a federation, the joining and resigning of federates from the federation, etc.
- **Declaration management:** These pertain to the publication and subscription of messages between federates.
- **Object management:** Calls that relate to the sending and receiving of messages to and from federates.
- **Ownership management:** Calls for transfer of an object and attribute ownership.
- **Time management:** These provide synchronisation services.
- **Data distribution:** For efficient routing of data between federates.

Mustafee and Taylor (2006a) have shown that a HLA-based CSP interoperability solution is possible by using services defined in at least four of these six management groups, viz., federation management, declaration management, object management and time management.

The time management component of the HLA supports interoperability among federates that use different time management mechanisms. These include federates executing simulations using both conservative and optimistic synchronisation protocols (Fujimoto and Weatherly, 1996). However, almost all research in CSP interoperability using the HLA standard is concerned with conservative synchronisation. For example, HLA-RTI has been used with CSPs AnyLogic™ (Borshchev et al., 2002), AutoSched™ (Gan et al., 2005) and Witness™ (Taylor et al., 2003). However, these individual research projects developed different and incompatible approaches to using CSPs together with HLA standard for distributed simulation.

Building on the lessons learnt from these work,

a standardization effort, described in Taylor et al. (2006), specifically addressing the problems of HLA-based distributed simulation and CSPs began in 2002. This has led to the development of a suite of CSP Interoperability (CSPI) standards under the Simulation Interoperability Standards Organization's (SISO) CSPI Product Development Group (CSPI PDG). The CSPI PDG's standards are intended to provide guidance on how specific requirements of HLA-based distributed simulation can be supported with CSPs.

This section has presented a detailed discussion on distributed simulation and the HLA-RTI middleware. In the next section the authors discuss the case study, the National UK Blood Service distributed simulation, wherein the HLA-RTI middleware has been used to execute a large and complex healthcare simulation model that was created using the CSP Simul8™.

THE HEALTHCARE SIMULATION CASE STUDY

Our blood supply chain study was carried out with the collaboration of the UK National Blood Service (NBS) and its main concern has been the analysis of policies for managing the blood inventory system in typical UK hospitals supplied by regional blood centres (Katsaliaki and Brailsford, 2007; Katsaliaki et al., 2007). The model was built using the CSP Simul8™. In order to overcome time execution problems of this large and complex healthcare model the researchers had to interface different copies of Simul8™, each executing part of the model in separate computers, with the HLA-RTI middleware for distributed simulation.

Background to the Blood Supply Chain

The NBS consists of 15 Process, Testing and Issuing (PTI) Centres which together serve 316 hospitals across England and North Wales. Each PTI Centre serves around 20 hospitals. Our case study was performed with the Southampton PTI Centre.

The NBS schedule collections of whole blood from voluntary donors in local, community venues or places of employment. The blood is transported back to the nearest PTI Centre where it is tested for ABO and Rhesus grouping and infectious diseases such as HIV. A unit (450ml) of whole blood is then processed into around 115 different products, of which the main three are red blood cells (RBC), platelets and plasma. RBC have usually a shelf life of 35 days and platelets of 5 days. Plasma can be frozen and stored for up to a year. In this study we consider only RBC and platelets which together comprise 85% of issues and are the chief source of wastage and shortages.

Blood products are stored in the PTI Centre's blood bank until they are requested by the hospitals served by that Centre. There are mainly three types of delivery. Routine scheduled deliveries are usually made on a daily basis on milk runs and are free of charge. The NBS also makes additional deliveries to an individual hospital in response to specific requests by charging a small fee. These are: emergency deliveries, which are prioritised on receipt for immediate dispatch and transportation and ad-hoc deliveries, which are additional to routine deliveries. There is also a nationally coordinated scheme for transferring excess stock between PTI Centres.

The blood remains in the hospital bank until it is cross-matched (tested for compatibility) for a named patient after a doctor's request. Individual doctors are responsible for the quantity of blood products ordered for each patient in the hospital. It is common place for doctors to over-order to be on the safe side although there is some guidance from the Maximum Surgical Blood Ordering Schedule (MSBOS) which specifies how much blood is required for a given operation.

After cross-matching, blood is then placed in the "assigned inventory" for that patient for some time until the transfusion and for some "safety" time

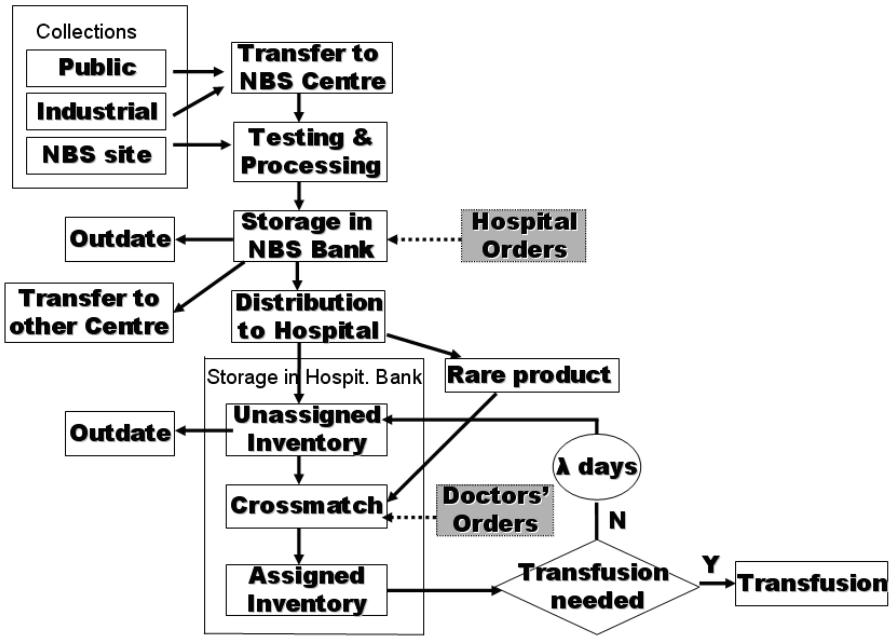
after if not used at that instant. If unused, it returns to "unassigned inventory" and is cross-matched to another patient. Blood units are usually sorted in a FIFO order (not age-based) in the hospital bank and issued accordingly. Ideally the oldest units should be issued first but this is not always the case since doctors prefer to use fresher blood, and moreover sorting the stock according to its age, especially in a big blood bank, is a painstaking procedure to which not all the hospitals devote the necessary time. In practice, only just above half of the cross-matched blood is actually transfused, and on average a unit will be cross-matched around three times before it is used or outdated. This clearly represents a huge potential for savings since the cost of a single unit of RBC is around £132 and for platelets is £214. Moreover, patients should ideally be given blood of the same type (a blood component is a set of eight products specified by the ABO and Rhesus group) but "mismatching" is possible in emergencies; for example, O-negative blood can be given to anybody. However, this practice is prohibited and is perceived as poor quality service.

Overall, there are two mechanisms for placing orders for blood; doctors who place orders to the hospital blood bank for their patients and hospital blood bank managers who place orders to the NBS Centre for stock replenishment. There are also two stocking processes; one at the central blood bank and one at the hospital bank; in the latter, stock can be either assigned or unassigned. Figure 2 illustrates all these processes in a flowchart.

The Standalone NBS Model

The blood supply system described above is undoubtedly a stochastic system with variable demand for blood (even for elective surgery) depending on the number of patients, type of operations and the occurrence of complications requiring extra transfusions. The supply is variable too since it relies on volunteers showing up to donate. Organisational issues also arise from the fact that the NBS manages the supply side and the hospitals manage the demand side of the logistics chain. As discussed earlier, DES was

Figure 2. Lifecycle of a blood unit (Katsaliaki, 2008)

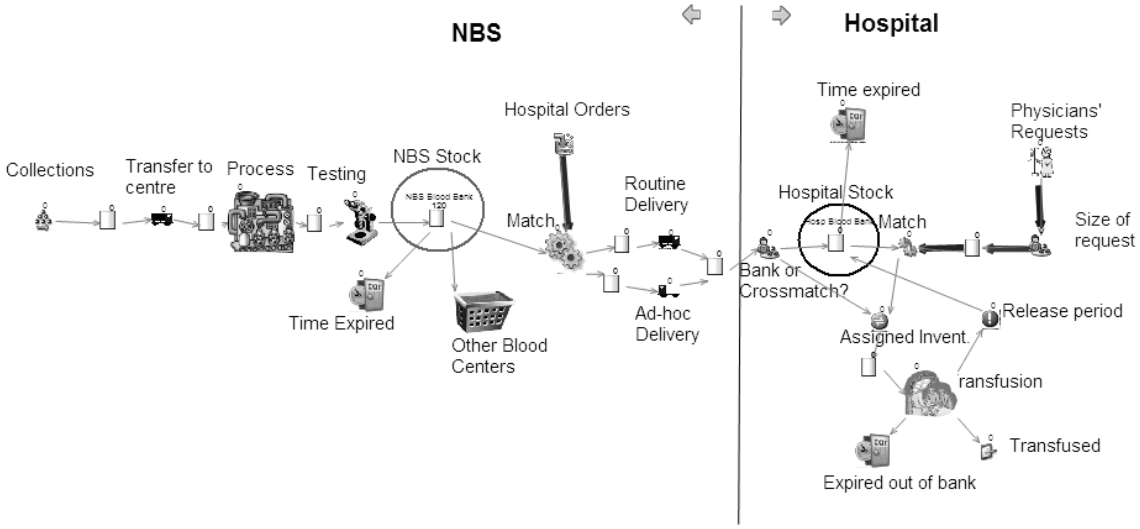


chosen to investigate the problems of this supply chain as complex stochastic multi-product, multi echelon perishable inventory problems have been shown to be intractable by analytic techniques (Donselaar et al, 2006; Goyal and Giri, 2001) or other simulation methods. DES was also the technique which the majority of the researchers have adopted to tackle parts of this problem since some decades ago.

The model was built using the CSP Simul8™. The supply chain model is very large and complex, and as such requires extensive data. Up to nineteen months' data were acquired for the years 2003 and 2004 by the NBS information system (PULSE) providing details about collections, processed and issued units, stock holding and discards. This gave details of the products supplied to each hospital, by date, time, delivery type, quantity and blood group. Questionnaires were also sent to the hospitals supplied by the Southampton centre, and interviews conducted with NBS staff and hospital blood bank managers. There are two main categories of entities in the model; items and orders. Items are the individual blood units (RBC and platelets) delivered from the NBS Centre to the hospitals in a one-way direction, since returns of products are not allowed. Items are also delivered internally from the hospital bank to the patient changing their stage in the system. In the latter returns within the hospital bank are allowed. Orders are placed both by the doctors to the hospital blood bank and by the hospitals' blood bank managers to the NBS Centre for blood products, and represent the backwards flow of information. Requests are matched with items according to their characteristics (attributes) as in a Kanban system and delivered as appropriate.

While the model runs, data are reported in an Excel file, such as the day and time of placing an order with the Centre, the type of order (routine, ad-hoc or emergency), the requested product and the amount by blood group. The model time units are minutes, and the remaining shelf-life of blood products is counted in minutes. However, the hospitals' blood bank stock for placing orders to the NBS is checked only every hour. Moreover, the decision to run the model in minutes was enforced by the fact that many processes, such as physician requests and delivery times, could be better approximated in small units of time. In

Figure 3. Screenshot of a simplified version of the Simul8 model showing one hospital



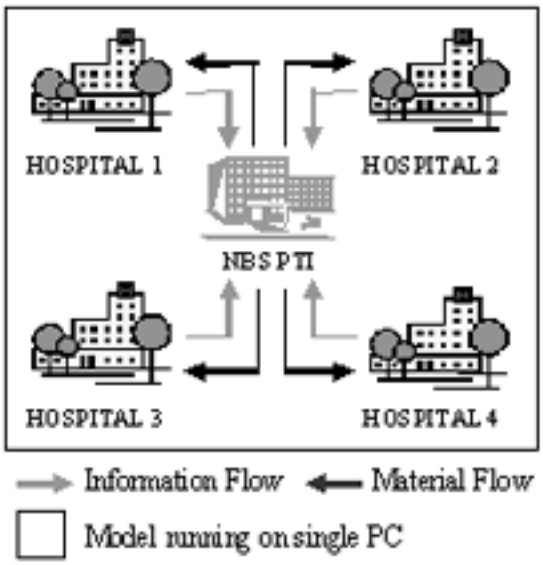
addition, an attempt to run the model in hours did not significantly accelerate the overall running time.

The basic version of the model contains the processes of the NBS Centre, from collection of whole blood to delivery of blood products, and the processes within a single medium-volume hospital. The model captures all the main processes of the supply chain. Figure 3 shows a simplified illustration of this simulation model. The black arrows represent blood units flow and the thick blue arrows represent information flow, i.e. orders.

The expanded version of this model incorporates the Centre's supply of multiple hospitals. Figure 4 shows an example of the relationships between the NBS supply centre and hospitals it serves, which in the "conventional" approach is simulated on a single computer. Note that this shows four hospitals. Ideally, there should be up to twenty!

In our "conventional" or standalone single computer approach the execution times increased dramatically when the number of hospitals increased in the model (see section 5). We now describe our distributed simulation approach to this problem.

Figure 4. NBS conventional model with the NBS PTI and four hospitals

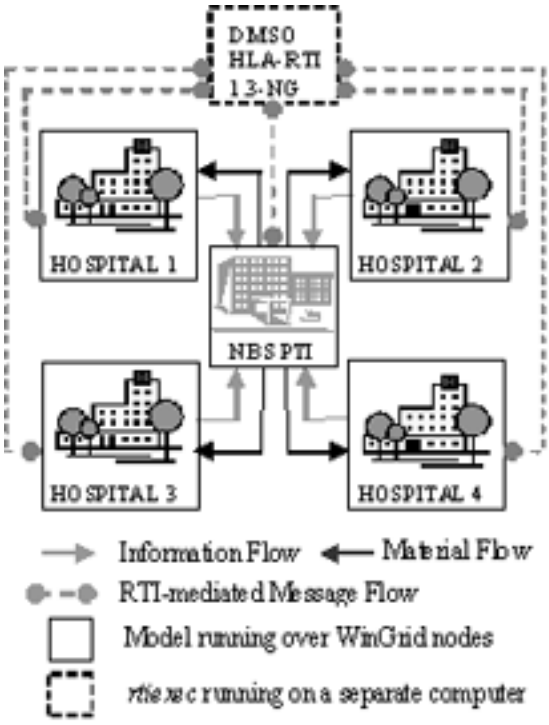


The Distributed NBS Model

In our distributed simulation of the NBS supply chain, we (1) divided up the conventional NBS model into different model elements, (2) used the HLA-RTI middleware and (3) interfaced different copies of CSP Simul8™ with the HLA-RTI. The version of the HLA-RTI we have used for our research is the DMSO RTI 1.3NG *(Rtiexec.exe)*. Figure 5 shows the logical relationship between the different parts of the distributed NBS model.

The model decomposition creates individual models of the Southampton PTI and hospitals (in this example four different hospitals were used). These models run in separate copies of Simul8™. Together they form federates that interact by time-stamped messages that represent the interaction

Figure 5. NBS distributed model with NBS PTI and four hospitals



of one model part with another (e.g., when an entity leaves one part of a model and arrives at another). These are mapped onto HLA interactions. The complete model, constituted of distributed federates, form our NBS supply chain federation. Note that in this work, Simul8™, like most of the other CSPs, does not provide inbuilt support for distributed simulation. The modifications made to the Simul8™ CSP are described in Mustafee and Taylor (2006).

In this investigation, to make our approach as easy to use and support as possible, interaction between the models/Simul8 and the HLA-RTI is via an Excel file. For example, entities representing orders are written into the file by Simul8 during the execution of hospital models. The HLA-RTI then correctly transfers this information to the NBS model by means of HLA interactions. This approach was adopted as it did not require special modifications to the CSP and modellers are typically skilled to use the CSP with Excel (the HLA-RTI/Excel link is very simple). The incoming orders from each hospital are collected into their corresponding queues in the NBS model and the orders are matched with the available stock of blood. The resulting matched units are written into an Excel spreadsheet in the NBS federate. This information is then sent to the different hospital models in a similar manner. As mentioned above, the decision to implement the distributed supply chain in this manner was motivated by issues of end user transparency and ease of implementation.

EXPERIMENTS AND RESULTS

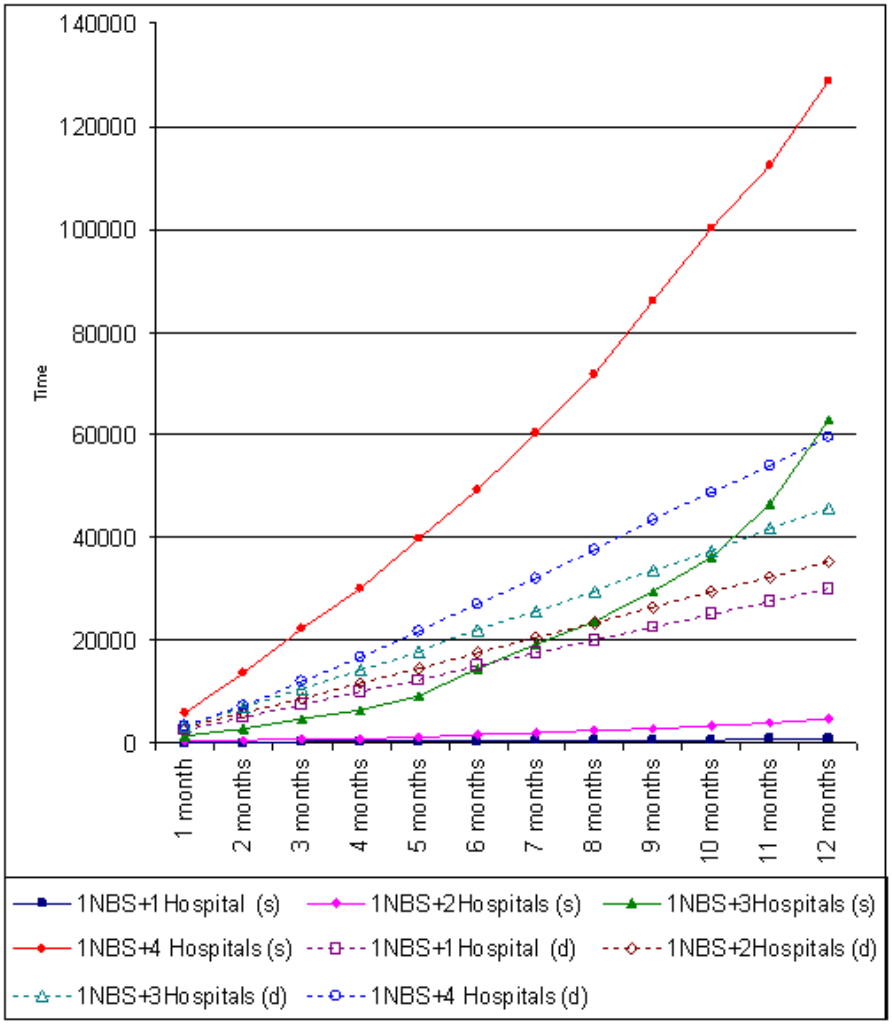
To investigate the distributed approach against the conventional approach, four scenarios were investigated. These were one NBS supply centre serving one, two, three and four hospitals respectively. The hospitals which were added to the models were all of the same size. For instance, physician requests were around 1000 blood units for each hospital per month, with each hospital diverging by a small percentage (≤6%) from the mean. Before experimentation commenced, the outputs for the conventional and distributed models were compared to check that the same results for a year's run was produced. This was done to validate the minor modifications to link Simul8/Excel/HLA-RTI in the distributed model did not artificially increase/decrease the workload. All experiments were conducted on Dell Inspiron laptop computers running Microsoft Windows XP operating system with 1.7GHz processors and 1GB RAM connected through a 100Mbps CISCO switch. The same computer specifications were used to guarantee consistency in runtimes. The results of the execution times for each of the models are based on the average of 5 runs (selected as an isolated network was used – low variance).

Figure 6 shows the execution time in seconds for both conventional and distributed approaches as the NBS simulation progress, month by month. The results show that the conventional model with one hospital took approximately 14 minutes to run for a whole simulated year. The run time rose to 78 minutes when the model ran with two hospitals and to approximately 17.5 hours with three hospitals. The addition of the fourth hospital increases the execution time to 35.8 hours. The distributed model with one NBS supply centre and one hospital ran in approximately 8.5 hours, with two hospitals in 9.8 hours, with three hospitals in 12.7 hours and with four hospitals in 16.5 hours.

DISCUSSION

The NBS simulation is representative of a large model. Looking at the results as the system size grows, runtimes increase rapidly from the trivial (for one hospital) to the extravagant (for only four hospitals!). From the results it is apparent that the versions with one or two hospitals are less time consuming to run using the conventional approach. Conversely, when a third and fourth

Figure 6. Runtimes of conventional (s) and distributed method (d) for one NBS PTI centre with one to four hospitals



hospital are added then the distributed method bests the runtime of the conventional approach. There also appears to be an exponential escalation of the runtime in the conventional version while increasing the number of hospitals in the model. This is quite a contrast to the substantially smaller and smoother rise in the runtime in the distributed method.

The enormous number of entities in the system, each of which carries many attributes, increases the computation time of the conventional model exponentially even though there is no exponential element in the functions of the model. This exponential increase appears to result from a combination of two different factors. Firstly, the massive amount of information generated by the model cannot be accommodated in the random access memory (RAM) alone, and hence the operating system has to keep swapping information to the hard disk. The part of the hard disk which is kept aside for use as swap space is called virtual memory. It takes much more time for the processor to recall information from the virtual memory when compared to RAM. Thus, as the models get bigger,

more information is generated resulting in more swaps between the RAM and the virtual memory, thereby contributing to an increase of execution time. Secondly, the behaviour of the system being modelled is such that all entities (blood units) in the system have a limited shelf life. This behaviour is modelled in the NBS simulation by continually scheduling events that decrease the shelf life of each entity by the minute. This results in more computations as the number of entities flowing through the system increases. Thus, the increase in runtime appears to be primarily due to a large event list caused by a combination of the volume of entities and the "counting down" of the shelf life. The large event list in turn causes swapping between RAM and virtual memory which further causes long runtimes. Our results suggest that the distributed approach allows the processing and memory demands made by large event lists to be shared over several computers.

What does this mean overall? In this particularly large scale simulation, our distributed simulation effectively halved the time taken to perform a single of the NBS simulation with four hospitals. The trend appear to indicate that this performance gain will become better with more hospitals. The *magnitude* of the problem means that runtimes will never be trivial. However, our approach to using distributed simulation, one that is generalizable to many different CSPs and large scale simulations, means that one may expect large runtimes to be reduced. As mentioned previously, this work is important as it is the first demonstration that an effective distributed simulation technique *can* help in the reduction of unrealistic runtimes and thus make possible decision support systems for large scale problems that were hitherto unfeasible.

FUTURE TRENDS

To introduce this problem we commented that CSPs, although suitable for most simulations that are modelled in industry, may be unable to simulate large and complex models (Pidd, 2004b), as was the case with our NBS simulation. Arguably, one reason for this is, the larger the model, the greater the processing power and memory required to simulate the model. Simulation is a computationally intensive technology that has benefitted from increasing processor speeds made possible through advances in computer science; and with ever increasing processing speeds, the CSPs, in future, will possibly provide features that may not presently seem possible (for example, dramatic decrease in model runtime, execution of increasingly large and complex models, etc.) (Hollocks, 2006).

However, it is also true that with more processing power available the simulation user may tend to develop even larger and more complicated models simply because it is possible to do so (Robinson, 2005). This, in turn, may again mean that standalone CSPs will not be able to support execution of some user models because of their sheer size and complexity. We have demonstrated that distributed simulation can be used to help reduce the impact of size and complexity in terms of reducing runtimes. Our approach involved interfacing to the CSP via an Excel spreadsheet. However, if it was possible to interface directly to the CSP, then further performance gains could be made. Thus, CSPs that implement synchronisation algorithms or which allow interfacing with distributed simulation middleware like HLA-RTI, could make possible the simulation of large complex systems that are currently beyond the capability of many CSPs and thus beyond the reach of health care management.

CONCLUSION

This chapter has described an investigation into the decision support of large healthcare systems. The chapter has presented Discrete-Event Simulation (DES) as an approach to decision support but suffers from long runtimes when large systems

are simulated. The chapter also introduce COTS Simulation Packages (CSPs) as the main tools used for DES. We then introduced distributed simulation as a possible technique for sharing the processing load of large healthcare systems simulations. A case study comparing conventional and distributed approaches to simulating the supply chain of blood from a National Blood Service Centre to hospitals with the simulation package Simul8™ was then presented. Our results show that it is possible to use multiple computers to reduce the runtime of large simulations using distributed simulation. Further performance gains could be made if this technique and its associated technologies are directly integrated into the CSPs used for simulation. We therefore argue that Discrete-Event Simulation, supported by CSPs with integrated distributed simulation technology, could make possible a range of large scale decision support tools for healthcare that are currently not possible. As healthcare systems are growing, and the need for carefully considered decisions increase, the lack of such technology could place a significant barrier to effective healthcare in the future. We hope that our demonstration shows that this barrier could be overcome and that it is entirely possible that healthcare systems, and healthcare informatics as a field, could widely benefit from this complex, but simply realised, technique.

REFERENCES

Baezner, D., Lomow, G., & Unger, B. W. (1990). Sim++: The transition to distributed simulation. In D. Nicol (Ed)., *Proceedings of the 1990 SCS Western Multiconference on Simulation: Distributed Simulation,* January 17-19 (pp. 211-218). San Diego, CA: Society for Computer Simulation.

Bagni, R., Berchi, R., & Cariello, P. (2002). A comparison of simulation models applied to epidemics. *Journal of Artificial Societies and Social Simulation*, *5*(3).

Barton, P., Bryan, S., & Robinson, S. (2004). Modelling in the economic evaluation of health care: selecting the appropriate approach. *Journal of Health Services Research & Policy*, *9*(2), 110–118. doi:10.1258/135581904322987535

Boer, C. A. (2005). *Distributed simulation in industry*. PhD thesis, Erasmus Research Institute of Management (ERIM), Erasmus University Rotterdam, The Netherlands. Retrieved February 4th, 2008 from https://ep.eur.nl/handle/1765/6925.

Borshchev, A., Karpov, Y., & Kharitonov, V. (2002). Distributed simulation of hybrid systems with AnyLogic and HLA. *Future Generation Computer Systems*, *18*(6), 829–839. doi:10.1016/S0167-739X(02)00055-9

Chandy, K.M. & Misra, J. (1979). *Distributed Simulation: A Case Study in Design and Verification of Distributed Programs, 5*(5), 440-452.

Cooper, K., Davies, R., & Brailsford, S. (2007). Choice of modelling technique for evaluating health care interventions. *The Journal of the Operational Research Society*, *58*, 168–176.

Eldabi, T., Paul, R. J., & Taylor, S. J. E. (2000). Simulating economic factors in adjuvant breast cancer treatment. *The Journal of the Operational Research Society*, *51*(4), 465–475.

Fischer, M. C., Adams, A., & Miller, G. (1994). Aggregate level simulation protocol (ALSP) - training for the future. In *Proceedings of the 1994 Military Operations Research Symposium*. Military Operations Research Society (MORS), USA. Retrieved February 15th, 2008 from http://ms.ie.org/alsp/biblio/mors_94_fischer/mors_94_fischer.html

Fone, D., Hollinghurst, S., Temple, M., & Round, A. (2003). Systematic review of the use and value of computer simulation modelling in population health and healthcare delivery. *Journal of Public Health Medicine*, *25*(4), 325–335. doi:10.1093/pubmed/fdg075

Fujimoto, R. M. (1990). Parallel discrete event simulation. *Communications of the ACM*, *33*(10), 30–53. doi:10.1145/84537.84545

Fujimoto, R. M. (1999a). Parallel and distributed simulation. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, & G. W. Evans, (Eds.), *Proceedings of the 31st Winter Simulation Conference*, (pp. 122– 131). New York: ACM Press.

Fujimoto, R. M. (1999b). *Parallel and distributed simulation systems*. New York: John Wiley & Sons.

Fujimoto, R. M. (2001). Parallel and distributed simulation systems. In B. A. Peters, J. S. Smith, D. J. Medeiros, & M. W. Rohrer (eds.), *Proceedings of the 33rd Winter Simulation Conference*, (pp. 147-157). Washington, DC: IEEE Computer Society.

Fujimoto, R. M. (2003). Distributed simulation systems. In S. Chick, P. J. Sánchez, D. Ferrin, & D. J. Morrice (Eds.) *Proceedings of the 35th Winter Simulation Conference*, (pp. 124-134). Winter Simulation Conference, USA.

Fujimoto, R. M., & Weatherly, R. M. (1996). Time management in the DoD high level architecture. In *Proceedings of the 10th Workshop on Parallel and Distributed Simulation Workshop* (pp. 60-67). Washington, DC: IEEE Computer Society.

Gan, B. P., Lendermann, P., Low, M. Y. H., Turner, S. J., Wang, X., & Taylor, S. J. E. (2005). Interoperating Autosched AP using the high level architecture. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, & J. A. Joines (Eds.) *Proceedings of the 37th Winter Simulation Conference*, (pp. 394-401). Winter Simulation Conference, USA.

Gipps, P. J. (1986). The role of computer graphics in validating simulation models. *Mathematics and Computers in Simulation*, *28*, 285–289. doi:10.1016/0378-4754(86)90049-2

Goyal, S., & Giri, B. (2001). Recent trends in modeling of deteriorating inventory. *European Journal of Operational Research*, *134*(1), 1–16. doi:10.1016/S0377-2217(00)00248-4

Hollocks, B. W. (2006). Forty years of discrete-event simulation - a personal reflection. *The Journal of the Operational Research Society*, *57*(12), 1383–1399. doi:10.1057/palgrave.jors.2602128

IEEE. 1516 (2000). *IEEE standard for modelling and simulation (M&S) high level architecture (HLA)*. New York: Institute of Electrical and Electronics Engineers.

Jefferson, D. R. (1985). Virtual Time. *ACM Transactions on Programming Languages and Systems*, *7*(3), 404–425. doi:10.1145/3916.3988

Jun, J. B., Jacobson, S. H., & Swisher, J. R. (1999). Application of discrete-event simulation in health care clinics: A survey. *The Journal of the Operational Research Society*, *50*, 109–123.

Karlsson, M., & Olsson, L. (2001). pRTI 1516 - rationale and design. In *Proceedings of the 2001 Fall Simulation Interoperability Workshop* (01F-SIW-038). Orlando, FL: Simulation Interoperability Standards Organization.

Katsaliaki, K. (2008). Cost effective practices in the blood service sector. *Health Policy.*

Katsaliaki, K., & Brailsford, S. B. (2007). Using Simulation to Improve the U.K. Blood Supply Chain. *The Journal of the Operational Research Society*, *58*, 219–227.

Katsaliaki, K., Mustafee, N., Taylor, S.J.E., Brailsford, S. (2007). Comparing Conventional and Distributed Approaches to Simulation in a Complex Supply-Chain Health System. *Journal of the Operation Research Society*.

Law, A. M. (2007). *Simulation Modelling & Analysis*. New York: McGraw-Hill International Edition.

Linden Research. (2007). *Second Life*. Retrieved February 9th, 2008 from http://secondlife.com/.

Miller, D. C., & Thorpe, J. A. (1995). SIMNET: the advent of simulator networking. *Proceedings of the IEEE*, *83*(8), 1114–1123. doi:10.1109/5.400452

Mustafee, N. (2004). *Performance evaluation of interoperability methods for distributed simulation*. MSc. thesis, Department of Information Systems, Computing and Mathematics, Brunel University, UK.

Mustafee, N., & Taylor, S. J. E. (2006). Investigating distributed simulation with COTS simulation packages: experiences with Simul8 and the HLA. In J. Garnett, S. Brailsford, S. Robinson, & S. Taylor, (Eds.) *Proceedings of the 2006 Operational Research Society Simulation Workshop (SW06),* (pp. 33-42). Birmingham, UK: Operational Research Society.

Nicol, D., & Heidelberger, P. (1996). Parallel Execution for Serial Simulators. *ACM Transactions on Modeling and Computer Simulation*, *6*(3), 210–242. doi:10.1145/235025.235031

Page, E. H., & Nance, R. E. (1994). Parallel discrete event simulation: a modeling methodological perspective. In *Proceedings of the 8th Workshop on Parallel and Distributed Simulation*, (pp. 88-93). New York: ACM Press.

Pidd, M. (2004a). *Computer simulation in management science* (5th Ed.). Chichester, UK: John Wiley & Sons.

Pidd, M. (2004b). Simulation worldviews: so what? In R. G. Ingalls, M. D. Rossetti, J. S. Smith, & B. A. Peters, (eds.) *Proceedings of the 36th Winter Simulation Conference*, (pp. 288-292). Winter Simulation Conference, USA.

Pidd, M., & Cassel, R. A. (2000). Using Java to develop discrete event simulations. *The Journal of the Operational Research Society*, *51*, 405–412.

Reynolds, P. F. (1988). A spectrum of options for parallel simulation. In M. Abrams, P. Haigh, and J. Comfort (Eds.), *Proceedings of the 20th Winter Simulation Conference*, (pp. 325 –332). New York: ACM Press.

Robinson, S. (1994). *Successful Simulation: A Practical Approach to Simulation Projects*, Maidenhead, UK: McGraw-Hill.

Robinson, S. (2005a). Discrete-event simulation: from the pioneers to the present, what next? *The Journal of the Operational Research Society*, *56*(6), 619–629. doi:10.1057/palgrave.jors.2601864

Royston, G. (1999). Commentary: trials versus models in appraising screening programmes. *British Medical Journal*, *318*, 360–361.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method.* New York: John Wiley & Sons.

Taylor, K., & Lane, D. (1998). Simulation applied to health services: opportunities for applying the system dynamics approach. *Journal of Health Services Research & Policy*, *3*, 226–232.

Taylor, S. J. E., Sharpe, J., & Ladbrook, J. (2003). Time management issues in COTS distributed simulation: a case study. In S. Chick, P. J. Sánchez, D. Ferrin, & D. J. Morrice, (Eds.) *Proceedings of the 35th Winter Simulation Conference*, (pp. 838-846). Winter Simulation Conference, USA.

Taylor, S. J. E., Sudra, R., Janahan, T., Tan, G., & Ladbrook, J. (2002). GRIDS-SCF: An infrastructure for distributed supply chain simulation. *Simulation*, *78*(5), 312–320. doi:10.1177/0037549702078005553

Taylor, S. J. E., Wang, X., Turner, S. J., & Low, M. Y. H. (2006). Integrating heterogeneous distributed COTS discrete-event simulation packages: an emerging standards-based approach. *IEEE Transactions on Systems . Man and Cybernetics: Part A*, *36*(1), 109–122. doi:10.1109/TSMCA.2005.859167

van Donselaar, K. H., van Woensel, T., Broekmeulen, R. A. C. M., & Fransoo, J. C. (2006). Inventory control of perishables in supermarkets. *International Journal of Production Economics*, *104*(2), 462–472. doi:10.1016/j.ijpe.2004.10.019

KEY TERMS AND DEFINITIONS

COTS: Commercial, Off-The-Shelf (COTS). This term is used to refer to software applications that can be purchased from software vendors.

CSP: COTS Simulation Package (CSP). In this thesis the term CSP is used to refer to simulation packages for both Discrete-Event Simulation (DES) and Monte Carlo Simulation (MCS).

HLA: The High Level Architecture (HLA) is an IEEE standard for distributed simulation.

HLA-RTI: The High Level Architecture-Run Time Infrastructure (HLA-RTI) is distributed simulation middleware that implements the interface specifications outlined by the HLA standard.

Rtiexec: rtiexec.exe is the HLA-RTI middleware program.

Discrete Event Simulation (DES): DES is an approach to modelling using interconnected blocks to represent interaction between specific processes and is run on a computer using mathematical models. The latter are stochastic, that is they involve input generated according to probability distributions. A discrete model assumes that the state of the system changes only at specific times, often referred to as events.