INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 00 (2011) 1–19
DOI: 10.1111/j.1475-3995.2011.00838.x

Motivations and barriers in using distributed supply chain simulation

Navonil Mustafee[a], Simon Taylor[b], Korina Katsaliaki[c], Yogesh Dwivedi[a] and Michael Williams[a]

[a] *School of Business and Economics, College of Business, Economics and Law, Swansea University, Swansea, Wales, United Kingdom*
[b] *Department of Information Systems and Computing, Brunel University, Middlesex, United Kingdom*
[c] *School of Economics and Business Administration, International Hellenic University, Thessaloniki, Greece, E-mail: n.mustafee@swansea.ac.uk [Mustafee]; simon.taylor@brunel.ac.uk [Taylor]; k.katsaliaki@ihu.edu.gr [Katsaliaki]; y.k.dwivedi@swansea.ac.uk [Dwivedi]; m.d.williams@swansea.ac.uk [Williams]*

Received 02 February 2010; received in revised form 20 October 2011; accepted 22 October 2011

Abstract

Discrete event simulation (DES) is a technique that is used by analysts to take informed decisions regarding an existing or proposed system of interest. DES models typically represent the processes associated with various business units. However, in the case of supply chains more than one business unit may need to be modelled as different organisations may be responsible for various supply chain operations such as manufacturing, transport and logistics, distribution, warehouse operations, etc. Organisations can be protective about their internal processes and can have concerns regarding data/information security and privacy. Thus it could be argued that creating a single DES supply chain model representing the various inter-organisational processes is usually not an option since this will run counter to organisational privacy. Further, issues such as data transfer, model composability and execution speed may also make a single model approach problematic. A potential solution could be to create several distinct and well-defined DES models, each modelling the processes associated with one specific supply chain business unit, linked together over the internet. We refer to this possible distributed approach as Distributed Supply Chain Simulation (DSCS). Although this approach holds great promise, there are technical barriers in using DSCS. The paper discusses the benefits and barriers of a distributed approach and then, using a healthcare DSCS, the technological feasibility is demonstrated. In conclusion, the paper argues that adopting a standardised approach to DSCS will remove a major barrier to its use.

Keywords: supply chain simulation; distributed supply chain simulation; distributed simulation; healthcare; high-level architecture; standards.

1. Introduction

Supply chains, by their very nature, are usually complex as they entail all the processes from procurement and manufacturing to sales and support (Stevens, 1989). Moreover, modern supply chain management approaches favour a global, holistic view in which the individual echelons share information and trust each other, rather than simply trying to optimise their own local processes independently of its neighbours (Chapman and Corso, 2005). Most of these multi-echelon and complex supply chains can benefit from Operational Research (OR) techniques. One such OR technique is "simulation"; it is recognised as the second most widely used technique after "modelling" in the field of Operations Management (Pannirselvam et al., 1999). Discrete event simulation (DES) is one such simulation technique that can be used to model supply chain simulations (SCSs).

SCS helps organisations determine the strategies that have the potential to provide the most flexible and profitable operating environment (Huang et al., 2003). SCS differs from the conventional types of DES (e.g. traditional manufacturing simulation) because it spans far beyond the confines of a single manufacturing site and its goal is to improve the financial position of an entire enterprise or a group of trading partners (Bagchi et al., 1998). Furthermore, the conventional model of supply chains, comprising only a single enterprise with multiple facilities and distribution centres, have been replaced by supply chains that have crossed organisational boundaries (Gan et al., 2000). Thus, whilst in traditional supply chains all operations (e.g. procurement, manufacturing, transport, inventory, distribution) were usually performed by a single entity, in "modern" supply chains this responsibility is shared amongst various organisations that come together towards realisation of the supply chain.

In the context of modelling SCS, a supply chain can be modelled using a single DES model. This model is typically created using a DES Commercial-off-the-shelf Simulation Package (CSP) such as Witness™ (Lanner Group), Simul8™ (Simul8 Corporation), AnyLogic™ (XJ Technologies) and Arena™ (Rockwell Automation). It will normally contain the logic of all the processes associated with the supply chain. In this case, there is usually no requirement for data/process secrecy because a conventional supply chain is characterised by the existence of a single organisation. However, for supply chains that consist of multiple organisations, it may not be possible to build a single model. This may be primarily due to strict process/data privacy requirements that may be enforced by such organisations (Mertins et al., 2005; Li et al., 2010). However, there may be other practical problems that prevent a single model from being developed without a high degree of inconvenience. The alternative is to build separate DES models reflecting each business unit in the supply chain. These separate SCS models can be linked together over a computer network such as the internet using specialist networking software to realise a *Distributed Supply Chain Simulation* (DSCS). Although this approach of executing SCS models permits data-hiding and yet enables the simulation of the entire supply chain, there are considerable technological barriers in implementing this solution. Despite these obstacles, there are significant potential benefits of this approach. This paper therefore aims to introduce the supply chain community to these distributed simulation concepts and its potential benefits (namely, ensuring data/model privacy, avoiding problems associated with model composability, enabling faster execution), and to make them aware of related research in this field. Furthermore, this paper demonstrates the technological feasibility of the CSP-based DSCS approach through a feasibility study in healthcare SCS and argues that the standardisation of

this technology is a key step forward in enabling its widespread use. Although the feasibility study focuses specifically on the speed-up factor, it also illustrates how data privacy is maintained and how issues of model composability can be avoided by keeping the individual supply chain components of the models separate.

The rest of the paper is organised as follows. Section 2 provides the motivation for combining distributed simulation and SCS to form DSCS. It presents an overview of distributed simulation and the need for synchronisation of simulation time across computers (Section 2.1). This section also includes a short introduction to the IEEE 1516 standard for distributed simulation (IEEE 2010), which is increasingly becoming the de-facto standard for such simulations. Section 3 reviews related work in this area. The healthcare supply chain feasibility study is presented in Section 4. The primary objective of the feasibility study is to present a proof-of-concept that distributed simulation is a viable technology to speed up simulation execution (demonstrated in Section 6). The secondary objective of the study is the demonstration of the technological feasibility of our CSP-based DSCS approach (demonstrated in Section 5). Section 7 then summarises the position of the paper and argues for a need to develop a standardised approach to CSP-based distributed simulation.

2. Distributed simulation and SCS

Figure 1 illustrates a possible supply chain scenario where DSCS could be applied and shows three organisations (X, Y and Z), each engaging in a specific activity. Here there may be concerns regarding information security since each company may not wish to reveal its data and internal

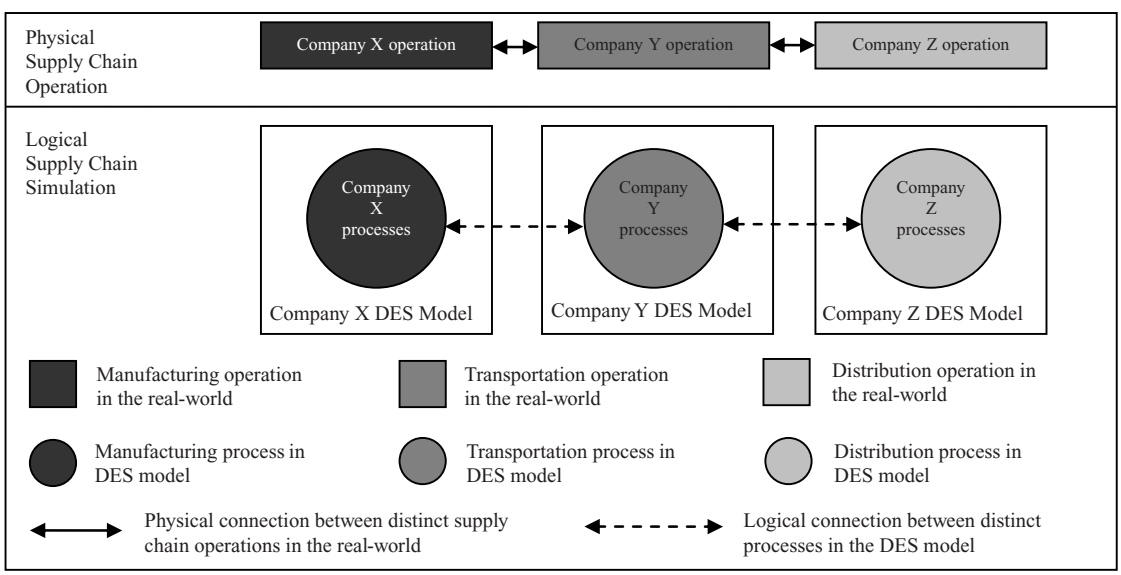

Fig. 1. Modern supply chain with organisations X, Y and Z involved in manufacturing, transportation and distribution operations, respectively. The logical simulation processes representing these operations are contained in three different DES simulations, each representative of the physical operation associated with a specific organisation.

processes to another company that it is happy to work with. If this supply chain was represented as a single model then these "secrets" would be revealed as they would be specified explicitly in the model. In addition to *privacy*, further problems include the following:

- *Data transfer/access problems*. Companies may be "open" to each other (i.e. happy to share data and internal processes). A single model will reside on a single computer in a particular place (say, organisation X). That model will need data drawn from Y and Z. However, databases can be large and time consuming to copy (even when accessed over the internet). Also, arguably, data when copied are instantly out of date. Running a model using copies of organisation data can therefore be time consuming and inaccurate.
- *Model composability problems*. If each of our organisations had previously developed models, these models cannot simply be "cut and pasted" into the same single model. Variable name clashes, global variables and different validation assumptions are three examples of the many problems with this approach. Further, if an organisation needs to update its model, it has to update the single model. How do we make sure that every organisation has the correct version of the single model? What if the update causes problems in another part of the single model owned by another organisation? Additionally, models developed in different CSPs are simply not compatible. One cannot transfer a model developed in one CSP into another without significant effort.
- *Execution time*. Large models will most likely develop large event lists that must be processed and updated each time an event is executed. This can take a considerable amount of time. Worse, the processing capacity of even a high specification PC may not be enough to physically cope as the actual CSP may have an upper limit on the event list size.

In the above cases, an alternative approach is needed. Here we create separate DES models for processes representative of each organisation. Linking the models together over a network such as the internet using *distributed simulation* technologies and techniques creates a DSCS. This allows the models to be executed separately and privately by companies X, Y and Z, respectively, to simulate organisation-specific processes while accessing local data, and avoiding many model composability issues (although execution speed may still be a problem as shown by our feasibility study).

2.1. Distributed simulation

Distributed simulation can be defined as the distribution of the execution of a single run of a simulation program across multiple processors (Fujimoto, 2000). Distributed simulation software (sometimes called *middleware*) is quite complex and implements well-known distributed simulation time management algorithms to achieve synchronisation between individual running simulations (Fujimoto, 1990). The time management algorithms are required for the prevention of causality errors. Causality errors happen as a result of a failure to process simulation events in an increasing timestamp order. More specifically, 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 time stamp $T1$ will have normally changed the state variables that will be used by the event with timestamp $T2$, this would amount to simulating a

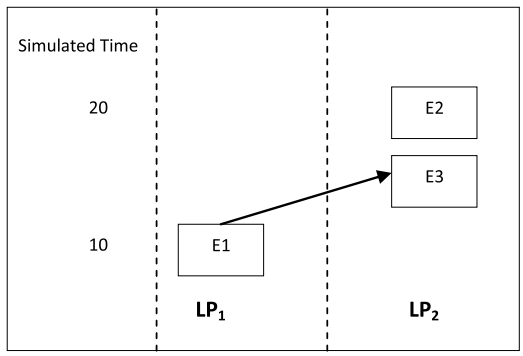

Fig. 2. Execution of events in a distributed simulation (adapted from Fujimoto, 1990).

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 fairly easy 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 may be composed of a number of physical processes. In a distributed simulation, each physical process is usually mapped to a logical simulation process running on a separate machine. In the context of supply chains, the physical processes may characterise the activities of manufacturing organisations or they may represent processes associated with storage, transport and logistics. All the interactions between the physical processes (e.g. material movement from one supply chain component to the other) are modelled as messages that are exchanged between their corresponding logical processes. Each message will have a time stamp associated with it.

In Fig. 2, the simulation represents a physical system that has two physical processes, say, PP_1 and PP_2. Logical simulation processes LP_1 and LP_2 model the two physical processes. Each of these logical processes has their own simulation engine, simulation clock and an event list. During simulation initialisation, the event lists of both LP_1 and LP_2 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 LP_1 to process event $E1$ without any causality error since the timestamp of $E1 <$ timestamp of $E2$. But LP_2 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 LP_2 at time 15. In such a case, if LP_2 had been allowed to execute $E2$ at simulated time 20 then it would have resulted in a causality error because the time stamp of $E3 <$ the time stamp of $E2$. Different synchronisation protocols are proposed for distributed simulation that prevents or corrects such causality errors.

The current standard to support this is the IEEE 1516 High Level Architecture (HLA) (IEEE 2010). This came from the need of the US Department of Defense to reduce the cost of training military personnel by reusing computer simulations linked via a communication network such as the internet. In HLA terminology, a distributed simulation is called a *federation*, and each individual simulator (in our case the combination of a CSP and its model) is referred to as a *federate*. The HLA *Federate Interface Specification* (FIS) defines distributed simulation software termed a *Runtime Infrastructure* (RTI). A distributed simulation is therefore a federation composed of many federates interacting over a communication network via RTI software (Fujimoto and Weatherly, 1996). There

are several RTIs available including the DMSO HLA-RTI (US DoD M&S Office, 1999) and the Pitch pRTI (Karlsson and Olsson, 2001). In DSCS, a model and its CSP software is a federate and the set of these federates is a DSCS federation. All the interactions between the models (e.g. transport of goods from the manufacturer to the wholesalers) are represented as messages that are exchanged between the federates via the RTI over the communication network in such a way that time is managed correctly.

3. Related work

There are several DES studies on distributed simulation of supply chains. The two primary motivations of these studies are (a) distributed simulation as an enabler of large and complex supply chain models; and (b) distributed simulation as an enabler of inter-organisational supply chain models (the need for privacy across supply chains is a factor commonly cited in the papers that belong to this category). The review of literature presented in this section is thus grouped under the two aforementioned categories. The third section presents existing literature on distributed simulation using CSPs and the HLA.

3.1. Distributed simulation as an enabler of large and complex supply chain models

Linn et al. (2002) describe a successful two-machine implementation of a distributed simulation model for an international transportation system in a supply chain network operation. Rabe and Jäkel (2003) analysed the requirements for distributed simulation in production and logistics. Lee and Wysk (2004) present a development of a top-down mapping mechanism for modelling and coordinating a federation of distributed DES models representing intra supply chain entities using an Enterprise Resource Planning system as the federation coordinator. The study by Bandinelli and Orsoni (2005) illustrates the design and use of a distributed simulation system for the assessment of competing outsourcing strategies in the context of large-scale manufacturing. Bandinelli et al. (2006) present an overview of standards, models and/or architectures, describing the technological choices of DSCS and propose how and when a distributed SCS framework is to be used.

Chong et al. (2006) developed a distributed simulation model that can be used to study a complex supply chain. They fine-tune the execution speed of the model, and then use the model to investigate how the frequency of inventory updates and demand changes affect the on-time delivery performance of the entire supply chain. Tammineni and Venkateswaran (2007) propose an Advanced Look-ahead Based Approach (ALBA), a hybrid conservative approach for time synchronisation that allows the models to run as fast as possible to the nearest interaction event. This is achieved using an improved supply chain domain specific look-ahead algorithm that handles multiple types of interactions.

In the semiconductor sector, Chong et al. (2004) describe how a distributed simulation test bed enables a very detailed SCS to study a customer-demand-driven semiconductor supply chain. Turner et al. (2000) describe their experiences on employing the HLA to support reusability and interoperability of this application area. Their experiments show that by fine-tuning the integration of the application with the HLA-RTI, considerable performance improvements can be achieved.

3.2. Distributed simulation as an enabler of inter-organisational supply chain models

Mertins et al. (2005) discuss the advantages of distributed simulation to assist DES models in analysing the behaviour of supply chains, especially those in which several enterprises are involved. This work highlights the fact that integrating local models of the supply chain into one complete model is time consuming and error prone. It argues that distributed simulation offers a solution to this problem and, furthermore, provides encapsulation, if supply chain partners do not wish to publish details of their node to other partners. Justifying that a distributed approach could be successful in modelling supply chains across multiple businesses where some of the information about the inner workings of each organisation may be hidden from other supply chain members, McLean and Riddick (2000) attempt to integrate distributed manufacturing simulation systems with each other, with other manufacturing software applications and with manufacturing data repositories. More recently, Li et al. (2010) present a distributed simulation framework to facilitate collaborative cluster supply chains simulation. The proposed integration framework constructs a cross-chain simulation while hiding model details within the enterprises.

Jain et al. (2007) present a distributed simulation based approach for supply chain interoperability testing. Simulations are used to represent real-life organisations to serve as sources and consumers of dynamic data. The data can be encapsulated according to the standard under consideration and exchanged with other organisations directly or through selected applications for testing. Furthermore, Iannone et al. (2007) propose an efficient architecture (SYNCHRO) that is able to synchronise, simply and securely, simulation models that are located in different geographical areas.

Hongyu et al. (2010) propose an HLA-distributed simulation method (WS-HLA) that combines Web Service technologies in order to support analyzing bullwhip effect and information sharing in the supply chain. They built a model of the Beer Game to verify the feasibility of the WS-HLA-based simulation method. Also, Taejong et al. (2009) proposed an SCS framework through a combination of Parallel and Distributed Simulation (PADS) and Web services technology. In this framework, PADS provides the infrastructure for SCS execution while Web services technology makes it possible to coordinate the SCS model.

3.3. Distributed simulation using CSPs and the HLA

There have been several attempts to create distributed simulations of supply chain using the Higher Level Architecture (IEEE 1516 2010). The first major work in the application of this primarily military technology to the civilian domain was done by Straßburger (2001). Another notable contribution in this area is the work by Hibino et al. (2002), wherein an HLA-based distributed simulation system was used to evaluate a very large manufacturing system by synchronising several simulators. Various other strategies have been investigated since then in several supply chain application areas. Individual research projects developed different, but incompatible approaches to the use of the HLA supporting distributed simulation with specific CSPs: AnyLogic™ (Borshchev et al., 2002), AutoSched™ (Gan et al., 2005), Witness™ (Taylor et al., 2005); and simulation languages MOD-SIM III™ (Johnson, 1999), DEVS (Al-Zoubi and Wainer, 2008) and SLX™ (Straßburger et al., 2007).

In recent years, attempts have been made to unify the above approaches into a single standard that is based on the HLA standard (Taylor et al., 2006). This has led to the development of a suite of CSP-distributed simulation standards under the Simulation Interoperability Standards Organization (SISO), led by the COTS Simulation Package Interoperability Product Development Group (CSPI PDG). The CSPI PDG standards are intended to provide guidance on how specific requirements of HLA-based distributed simulation can be supported with CSPs. Examples of research based on these standards include Wang et al. (2006) who study possible implementations; Taylor et al. (2005) who investigate the use of distributed simulation in engine manufacturing; Gan et al. (2005) and Lendermann et al. (2007) who investigate the use of distributed simulation in semiconductor manufacturing supply chains.

4. The distributed National Blood Service (NBS) model: a feasibility study

To illustrate the benefits of DSCS and to outline the technological barriers, this section presents a DSCS feasibility study. This study involves the UK National Blood Service supply chain simulated model and its realisation as a DSCS using the CSP Simul8^{TM} and the DMSO HLA-RTI (US DoD M&S Office, 1999) distributed simulation software. It is to be noted here that our feasibility study borrows the "conventional", one-computer NBS blood supply chain model that was previously developed as a separate piece of research by one of the co-authors of this paper (Katsaliaki and Brailsford, 2007). By "conventional" we mean a one-computer simulation that can be executed without the requirement of distributed simulation. Although the concept of using distributed simulation could have been applied to any arbitrary large and complex simulation model (perhaps, an imaginary model created by us), for our feasibility study we decided to use the conventional NBS supply chain model since it was a validated and verified model and was based on a case study that had inputs from the Southampton NBS PTI Centre. Even though we use the same simulation model for our distributed simulation experiments (albeit we divide the singular NBS model into several sub-models for individual execution), there are several key differences between the work published in Katsaliaki and Brailsford (2007) and the present work. These differences are presented in Table 1.

Table 1
Key differences between the conventional NBS case study (Katsaliaki and Brailsford, 2007) and the distributed NBS feasibility study presented in this paper

Conventional NBS case study: Katsaliaki and Brailsford (2007)	Distributed NBS feasibility study: Work presented in this paper
Focus: The development of the NBS model in order to aid decision making.	*Focus:* The application of distributed simulation principles and techniques to the conventional NBS model in order to execute the model faster.
Objective: Better blood ordering strategies that could be used by NBS to reduce wastage (operations research objective)	*Objective:* Proof of concept that distributed simulation is a variable technology to speed up simulation execution (faster execution objective)
Contribution: Application of simulation in modelling perishable healthcare products (i.e. those products that have limited shelf lives)	*Contribution:* Investigating the motivations (faster execution, data hiding) and the barriers to distributed supply chain simulation.

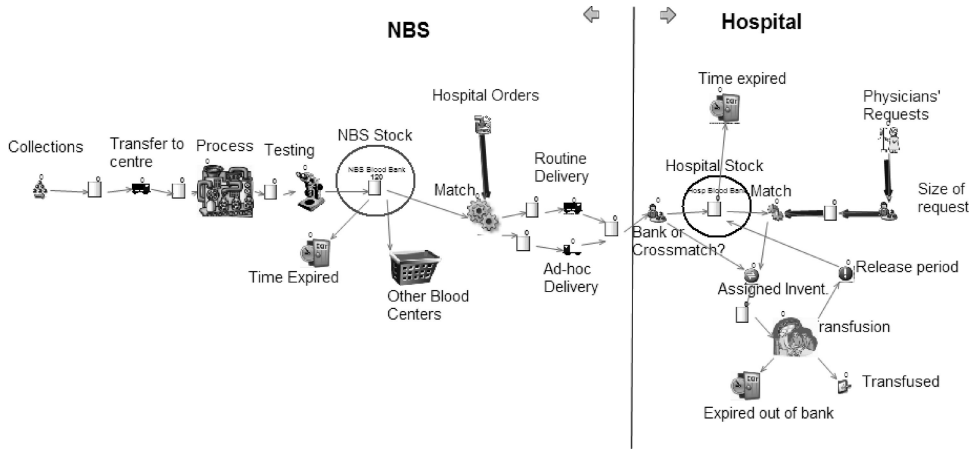

Fig. 3. Simplified model of the NBS supply chain with NBS PTI (left) and one hospital.

4.1. The conventional NBS SCS

The NBS is a part of the UK National Health Service Blood and Transplant (NHSBT) organisation. The NBS infrastructure consists of 15 Process, Testing and Issuing (PTI) centres that together serve 316 hospitals across England and North Wales. Each PTI Centre thus serves around 20 hospitals. The NBS is responsible for collecting blood through voluntary donations, classifying it by ABO and Rhesus grouping, testing the blood for infectious diseases such as HIV and processing the blood into around 115 different products (the main ones being red blood cells (RBC), platelets and plasma). The NBS stores the stockpile, transfers excess stock between different NBS centres and issues the different blood products to the hospitals according to demand. The conventional model, developed using the CSP Simul8^{TM}, contains the processes of the NBS PTI Centre, from the collection of blood to the delivery of blood products, and the processes of a hospital. The model captures physicians' requests for blood and the processes whereby the hospital blood bank checks its stock levels and places orders. The order entities and item entities are represented as information flow (hospital orders) and material flow (blood products), respectively. A single supply centre and hospital is shown in Fig. 3. Figure 4 shows a simplified diagram showing the relationship between four hospitals and one supply centre implemented as a single model.

4.2. The need for distributed NBS SCS

The problem with the conventional approach is threefold. Firstly, the data used are private and sensitive as it involves information related to clinical practice. Most data in healthcare systems cannot just be taken on demand and are subject to stringent and lengthy data protection checks. Admittedly in the UK, data sharing does take place between hospitals in Primary Care Trusts and, to some extent, Strategic Health Authorities (although these have recently been disbanded). However, the NHSBT and the NHS hospitals are effectively separate organisations and it cannot be assumed that data are freely shared. Further, to generalise the original work by Katsaliaki and

Brailsford (2007), i.e. a supply chain analysis tool usable by the many developed countries that have equivalent blood supply chains, then one cannot assume that privacy issues will be any different. Secondly, *the data are fixed*. The data sources are private *and not easily moved*. There is much data in the model that is used to model the demand for blood, the availability of blood products and the current stock of the blood units in the supply chain. These are updated frequently and centralising the data in a single model would make it difficult to ensure the data are up to date. Finally, *the execution time is extremely poor*. A single year took 14 minutes to run with a single supply centre and single hospital, 78 minutes with two hospitals, 17.5 hours with three hospitals and 35.8 hours with four hospitals (1.7 GHz processor desktop PC with 1 GB RAM). These results are discussed in detail in Section 6. Note that in terms of execution time it may be possible to simplify the modelling approach to increase the speed of the simulation (such as by sacrificing the detail at which blood product orders are placed and/or the shelf-life of blood products). However, the goal here is to understand wastage and ordering patterns, and a sacrifice in detail for the sake of performance may produce results faster but not at the required level of detail.

The DSCS version of the NBS model is shown in Fig. 5. The NBS models representing the supply centre and hospitals are executed on different computers locally to the organisations in the supply chain. Each CSP simulates the model of its element of the supply chain and interacts with other models as appropriate via a computer network and middleware described earlier in this paper. As shown in the figure, the NBS DSCS federation is composed of one *PTI federate* and *several hospital federates* interacting via an RTI and specially developed software called the CSP Controller Middleware. Figure 5 also shows the presence of a sixth *Manager* federate. This federate coordinates the execution of the NBS DSCS federation. The technical implementation is presented next.

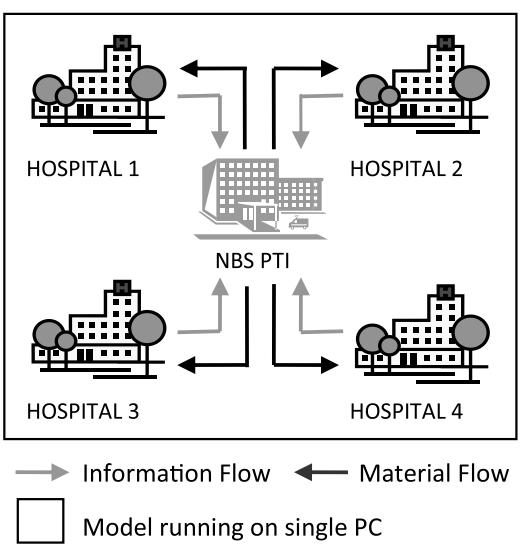

Fig. 4. Conventional SCS with NBS PTI and four hospitals being executed on the same computer.

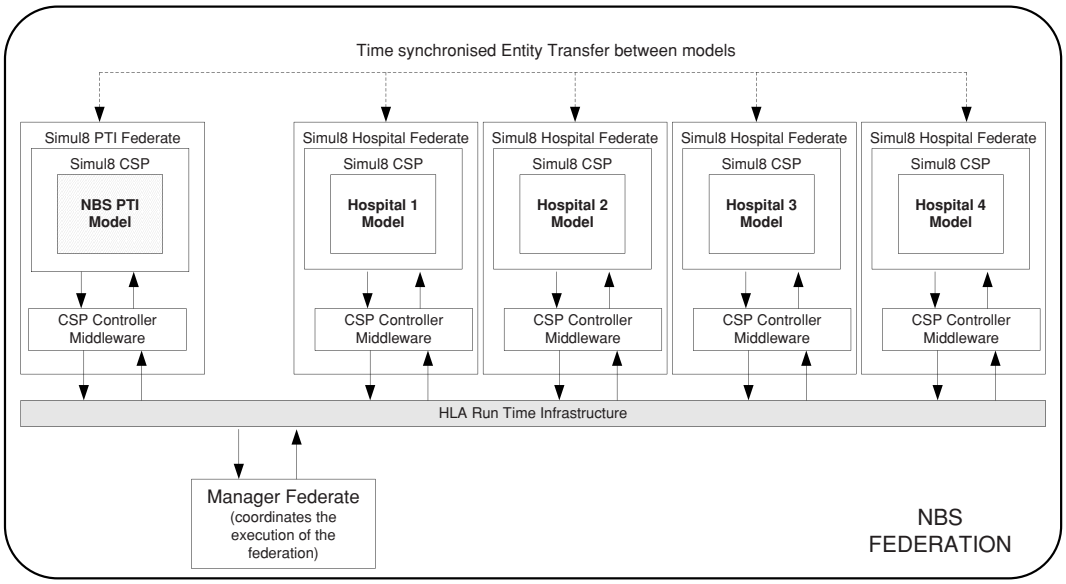

Fig. 5. The NBS distributed simulation federation comprising one NBS-PTI model, four NBS-Hospital models and one Manager Federate. The CSP controller middleware is the interface between the CSP Simul8^{TM} and HLA RTI.

5. Technical implementation of the distributed NBS SCS

Implementing distributed simulations is a complex process. We present here an overview of the technical implementation of the NBS DSCS feasibility study. This is intended to give guidance for anyone wanting to implement a DSCS using CSPs and follows approaches developed in Taylor et al. (2006), Mustafee and Taylor (2006) and Mustafee et al. (2009).

For models created using CSPs to inter-operate using the HLA standard, some of the FIS-defined interfaces have to be implemented. We have shown that an HLA-based CSP distributed simulation solution is possible by using services defined in four of these six management groups (Mustafee and Taylor, 2006). These are as follows:

- Federation Management: RTI calls for creation and deletion of federation; joining and resigning of federates from the federation and creation and realisation of synchronisation points.
- Declaration Management: Calls pertaining to publication and subscription of interactions.
- Object Management: Calls that relate to sending and receiving interactions.
- Time Management: RTI calls required to enable time constraint and time regulation and also to advance the federate simulation clock.

To link a CSP to an RTI, we have developed an approach to using an adaptor called the *CSP Controller Middleware (CCM)* (as modifying a CSP or RTI directly is often impossible). The CCM performs two specific tasks; it communicates with Simul8^{TM} through its COM interface and it interacts with RTI using the FIS-defined interfaces specification. Each of these two tasks is performed by two distinct components of the CSP controller middleware: the *Simul8^{TM} adapter* and the *RTI adapter*. These are described next.

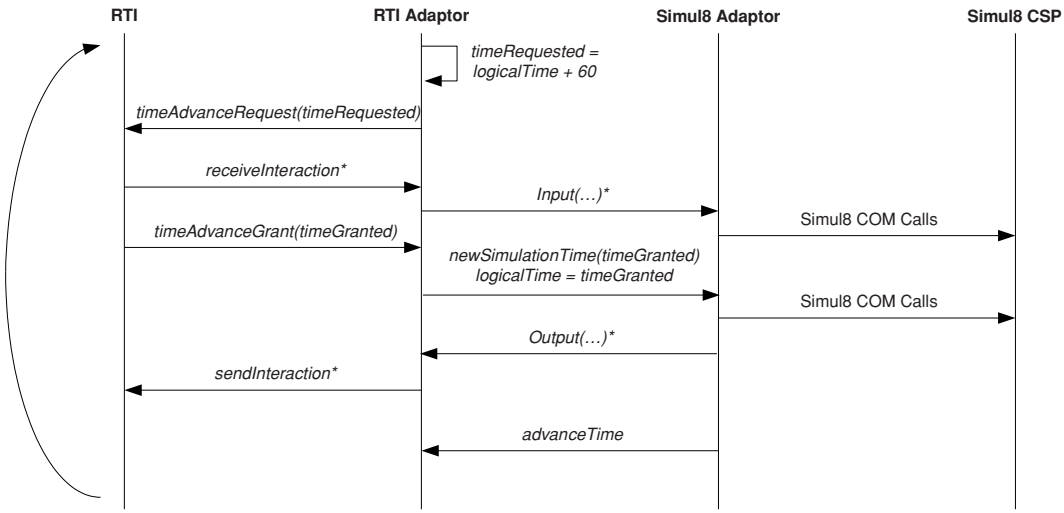

Fig. 6. CCM communication protocol showing the interactions between a federate (e.g. NBS PTI centre, hospitals, manager federate) and the HLA-RTI.

The Simul8^{TM} adapter defines methods such as OpenSim(modelFile), RunSimulation(time), getBloodOrdersFromHospital(hospital) and introduceEntitiesToHospital(hospital, bloodUnit) that are invoked by the RTI adapter to open a Simul8^{TM} modelFile, run the model to the time specified, get blood orders from hospital and to introduce entities into the hospital, respectively. These methods encapsulate both the application logic and the Simul8^{TM} COM method calls. For example, method getBloodOrdersFromHospital(hospital) has application logic that reads hospital order details being output by Simul8^{TM} into an Excel^{TM} file and method introduceEntitiesToHospital(hospital, bloodUnit) invokes Simul8^{TM} COM method ExecVL to set various bloodUnit parameters into the running hospital model and to schedule events. The Simul8^{TM} adapter also calls methods defined in the RTI adapter such as tellSimulationTimeEnd(time) and sendOrderToNBS(hospital, bloodOrder) to convey to the RTI adapter that Simul8^{TM} has completed processing a model till a defined "safe" time (see the discussion on causality error in Section 2.1; also refer to Fig. 2) and to transfer the bloodOrder collected from the hospital.

The RTI adapter methods contain application logic and invoke HLA-defined service calls. For example, the method *tellSimulationTimeEnd(time)* has application logic that sets the logical time of the federation to the *time* returned by the method call and *sendOrderToNBS(hospital, bloodOrder)* invokes HLA-defined method *sendInteraction* to pass the *bloodOrder* details from respective *hospital* federates to the NBS PTI federate in the form of HLA interactions. Thus, the CCM controls the time management of a CSP and controls the transfer of entities (in this case orders and blood units) in and out of the model and across the RTI (an example CCM communication protocol diagram is shown in Fig. 6).

To introduce the protocol presented in Fig. 6, we first clarify the FIS-defined *Time Advance Request* (TAR) service call. TAR is used to advance the simulation time of a federate; it is defined by the HLA standard and is implemented by the RTI. TAR is invoked with a time component that represents the logical time the federate wishes to move to. Invoking TAR will grant the simulation

federate the time requested *only when* the RTI determines that this would not lead to any causality error (refer to Fig. 2 and related discussion in Section 2.1). Until this new time is granted by the RTI, the simulation cannot proceed at the federate that made the TAR call.

As seen in Fig. 6, RTI adaptor invokes the TAR method call (*timeAdvanceRequest [timeRequested]*). This service call has a time argument (*timeRequested*) that specifies the simulation time to which the federate wants to move to. The CCM requests a time from the RTI that is always equal to its current logical time + 60 (*timeRequested = logicaltime+60*). This is because the NBS PTI centre and the hospitals exchange information at every 60 units of simulation time. The new time granted to the federate by the RTI is conveyed using the HLA TIME ADVANCE GRANT callback (*timeAdvanceGrant[timeGranted]*). This callback, invoked by the RTI on the federate RTI adapter, carries the time (*timeGranted*) that has been granted by the RTI and is a guarantee that there will be no external events (these are events from the other models running on separate computers) from the rest of the federation before this time. This new "safe" time is conveyed by the RTI adapter to the Simul8^{TM} adapter (*newSimulationTime[timeGranted]*) and the simulating federate processes the Simul8^{TM} model to this time. This may, in turn, generate other internal or external events. Subsequently, the logical time of the federate becomes equal to this new time (*logicalTime = timeGranted*) and the process of requesting time advancement using TAR starts all over again.

We now focus on how external events are sent across federates in our distributed NBS simulation. We use HLA interactions to achieve this. We can think of interactions as time-stamped messages that are sent by the RTI to individual federates. When a federate generates an external event the Simul8^{TM} adapter of CCM conveys this to the RTI adapter, which in turn invokes the HLA-defined service SEND INTERACTION (*sendInteraction**). Each interaction contains a time stamp and associated data. These interactions are sent to the RTI to be delivered to the respective federates in the causally correct order. On the receiving end, the RTI delivers the interactions to the RTI adapter though the RTI callback RECEIVE INTERACTION (*receiveInteraction**). The RTI adapter of the CCM then forwards the received data to the Simul8^{TM} adapter for introduction into the model. The data being exchanged in the federation relate to blood orders and deliveries. In both *sendInteraction** and *receiveInteraction**, the superscript "*" indicates that multiple interactions can be sent or received.

6. Speed-up analysis of the distributed NBS SCS

To investigate the performance of our NBS distributed simulation, we conducted experiments with four different scenarios. Each scenario represented one NBS PTI centre serving one, two, three or four hospitals, respectively. The name of the scenario reflected the number of hospitals that the NBS PTI catered for. For example, scenario 2Hospital would mean that two hospitals were being served by one NBS centre. In the distributed cases, scenario 2Hospital became three separate Simul8^{TM} models, each modelling either the NBS PTI centre, Hospital1 or Hospital2 and ran on three separate computers. In the standalone case, scenario 2Hospital meant that a single Simul8^{TM} model, running on a single PC, modelled the behaviour of the NBS centre and two hospitals.

Experiments were conducted on Dell Inspiron laptop computers running Microsoft Windows XP operating system with 1.73 GHz processors and 1 GB RAM with a medium specification desktop PC to host the RTI *rtiexec* software. These computers were connected through a 100 Mbps CISCO

switch and the RTI process (*rtiexec.exe*) was started on one of the computers. The results of the execution times for each of the scenarios were based on the average of five runs. 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 increased the execution time to 35.8 hours. Compared to this, the execution time for the distributed model was 7.2, 7.8, 10.3 and 15.5 hours for the 1Hospital, 2Hospital, 3Hospital and 4Hospital scenarios, respectively.

It is thus 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 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. Further, a more exhaustive analysis of the results reveals another significant feature. Every model for each method was monitored for its execution time per simulated month until the end of the run (1 year) as Fig. 7 shows. The graph clearly demonstrates that for the conventional method there is an upwards incremental trend in the runtime per added month. Especially for the model with one NBS supply centre and three hospitals, the monthly runtime rockets up from month 10 and over. For the model with four hospitals, this trend is apparent right from the first month (the reason for this is the enormous number of entities in the system, each of which carries many attributes, increases the computation time exponentially even though there is no exponential element in the functions of the model). The fluctuations in the runtimes between consecutive months are due to random variation.

These findings indicate that for the conventional method an expansion in model size will be accompanied by an exponential increase in both the total runtime and the time between iterations when the results are being collected. On the other hand, for the distributed method an increase in the number of hospitals (and therefore of computers) will be followed by a much smaller increase in total runtime, with no extensive increase in the time between iterations. Therefore, if more than two hospitals are added to any model, the distributed method would be a better platform in which to develop and run the simulation experiments. Overall, the distinctive trend that the two methods follow concerning runtimes seems to be continuous; in other words the more hospitals we add to the model, the more the differences in the runtimes between the two methods favour the distributed approach. The complete Southampton NBS supply chain model should include 16 hospitals. According to this feasibility study, it is clearly not feasible to run the conventional NBS simulation on a single PC, but the use of distributed simulation allows us the possibility of running the full model.

7. Conclusion

This paper has argued that the development of a single model of an SCS can bring with it issues of privacy, data transfer/access, model composability and execution time. DSCS has been introduced as a possible solution to the above as it avoids privacy, data and model problems and can introduce extra computing resources to reduce execution time. Some of the other key benefits of bringing

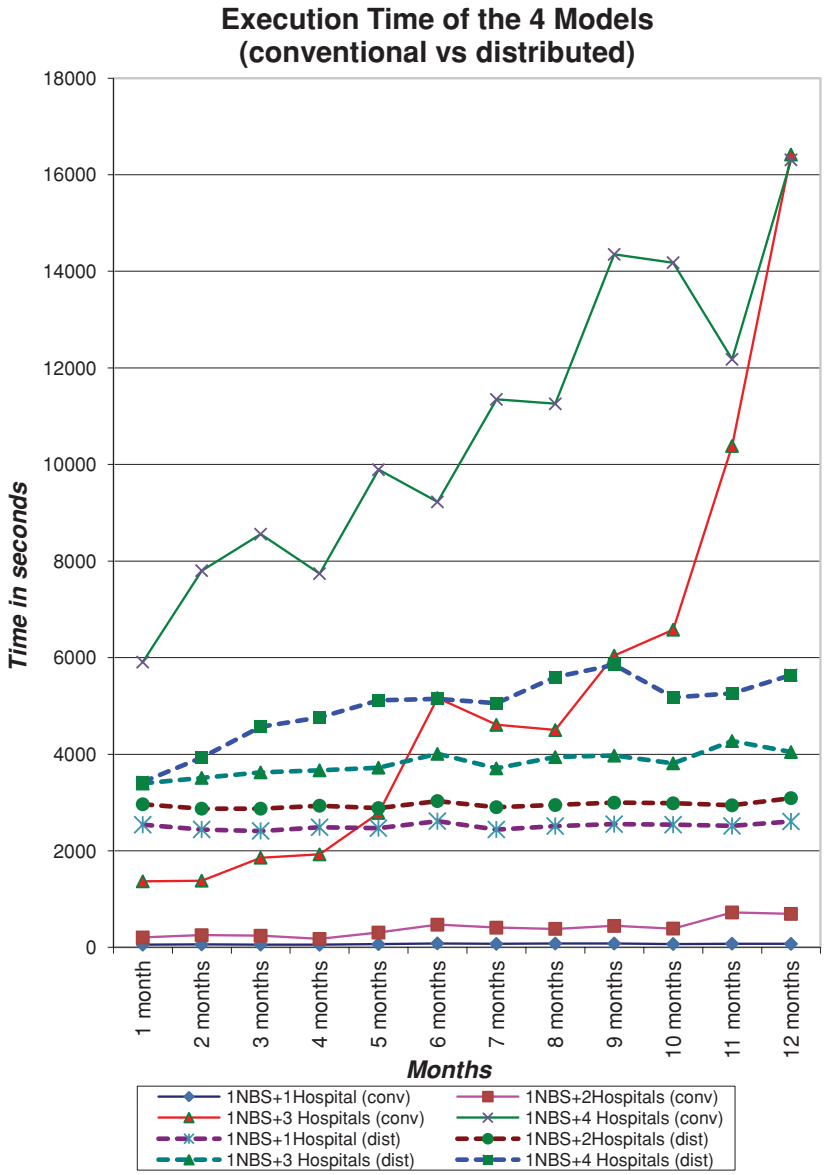

Fig. 7. Execution time per month of the four model scenarios (conventional and distributed).

together SCS and distributed simulation to form DSCS are the creation of supply chain models that access local up-to-date data, implement local changes efficiently and share the processing load of the model across the computers of the organisations. The modular nature of the individual models also means that these can be potentially "plugged" into different distributed models of other supply chains that an organisation might be part of as required (Lendermann, 2006; Boer et al., 2009).

However, there are two predominant barriers to implementing a DSCS solution. The first is that the present generation CSPs are not capable of executing distributed models directly as these are

set up for developing single models. Thus, the CSPs have to be interfaced with existing distributed simulation software by potentially costly software experts (i.e. a technical solution is not covered by the CSP licence fee). The second is that there is a very steep learning curve associated with implementing a CSP-HLA integration solution because it requires familiarity with distributed simulation theory, the HLA standard and HLA-based technology. Contemporary simulation vendors and consultancies rarely have this knowledge. These factors may be a reason why distributed simulation is widely and successfully used in the military but not in industry (Straßburger, et al., 2008).

The feasibility study presented in this paper has demonstrated the experimental realisation of a CSP-based DSCS. It is to be noted here that the primary purpose of using distributed simulation was to speed up the execution of the simulation (it had no other performance indicators). Although this may appear to be a trivial objective, the authors would like to point out that the "speed factor", in itself, is important since computing power can be a bottleneck to the development of simulation (Robinson, 2005). Also, and as highlighted by Pidd and Carvalho (2006), it is expected that there will be a steady increase in the number of large and complex simulations in the near future. Indeed, Taylor and Robinson (2006) have identified grid computing (which is also based on distributed computing principles) as a priority research area in the context of speeding up simulation execution. Although driven by the objective of decreasing the execution time, our feasibility study has also shown how data privacy is maintained and how problems of data transfer can be avoided through DSCS; it has illustrated how issues of model composability can be avoided by keeping the models separate. Finally, our distributed approach allows execution of certain large simulations that would not be possible using a conventional one computer simulation.

Although influenced by work described in the related work section, the technological implementation took a great deal of time and required expert knowledge and close collaboration with the vendor. However, we would like to emphasise that the time and effort that we expended in developing our solution could have been significantly reduced if there had been a standardised approach to implementing CSP-based distributed simulations. It is our hope that this paper will demonstrate the utility of DSCS and encourage interested stakeholders to get involved in standards development activities, such as that pioneered by the SISO COTS Simulation Package Interoperability Product Development Group (Taylor et al., 2007; SISO-STD-006-2010), to help reduce the learning curve of this arguably useful technology.

Acknowledgements

The authors would like to thank the Simul8 Corporation for their on-going support of this work. The authors would also like to thank Prof. Sally Brailsford for her contribution with the NBS case study.

References

Al-Zoubi, K., Wainer, G., 2008. Interfacing and coordination for a DEVS simulation protocol standard. In *Proceedings of the 12th International Symposium on Distributed Simulation and Real-Time Applications*, IEEE Computer Society, Washington, DC, 27–29 October, pp. 300–307.

Bagchi, S., Buckley, S.J., Ettl, M., Lin, G.Y., 1998. Experience using the IBM supply chain simulator. In *Proceedings of the 30th Winter Simulation Conference*, IEEE Computer Society Press, Los Alamitos, CA, 13–16 December, pp. 1387–1394.

Bandinelli, R., Orsoni, A., 2005. A DSS simulation model for outsourcing strategies in large-scale manufacturing. In *Proceedings of the 19th European Conference on Modelling and Simulation (ECMS)*, Riga, Latvia, 1–4 June, pp. 387–392.

Bandinelli, R., Rapaccini, M., Tucci, M., Visintin, F., 2006. Using simulation for supply chain analysis: reviewing and proposing distributed simulation frameworks. *Production Planning & Control* 17, 2, 167–175.

Boer, C.A., de Bruin, A., Verbraeck, A., 2009. A survey on distributed simulation in industry. *Journal of Simulation* 3, 1, 3–16.

Borshchev, A., Karpov, Y., Kharitonov, V., 2002. Distributed simulation of hybrid systems with AnyLogic and HLA. *Future Generation Computer Systems* 18, 6, 829–839.

Chapman, R.L., Corso, M., 2005. From continuous improvement to collaborative innovation: the next challenge in supply chain management. *Production Planning & Control* 16, 4, 339–344.

Chong, C.S., Lendermann, P., Gan, B.P., Duarte, B.M., Fowler, J.W., Callarman, T.E., 2004. Analysis of a customer demand driven semiconductor supply chain in a distributed simulation test bed. In *Proceedings of the 2004 Winter Simulation Conference*, Washington, DC, 5–8 December, pp. 1902–1909.

Chong, C.S., Lendermann, P., Gan, B.P., Duarte, B.M., Fowler, J.W., Callarman, T.E., 2006. Development and analysis of a customer-demand-driven semiconductor supply chain model using the high level architecture. *International Journal of Simulation & Process Modelling* 2, 3–4, 210–21.

Fujimoto, R.M., 1990. Parallel discrete event simulation. *Communications of the ACM* 33, 10, 30–53.

Fujimoto, R.M., 2000. *Parallel and Distributed Simulation Systems*. John Wiley & Sons, New York.

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*, IEEE Computer Society, Washington, DC, 22–24 May, pp. 60–67.

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 *Proceedings of the 37th Winter Simulation Conference*, Winter Simulation Conference, Orlando, FL, 4–7 December, pp. 394–401.

Gan, B.P., Liu, L., Jain, S., Turner, S.J., Cai, W., Hsu, W., 2000. Distributed supply chain simulation across enterprise boundaries. In *Proceedings of the 32nd Winter Simulation Conference*, Society for Computer Simulation International San Diego, CA, 10–13 December, pp. 1245–1251.

Hibino, H., Fukuda, Y., Yura, Y., Mitsuyuki, K., Kaneda, K., 2002. Manufacturing adapter of distributed simulation systems using HLA. In *Proceedings of the 34th Winter Simulation Conference*, IEEE Press, Piscataway, NJ, 8–11 December, pp. 1099–1107.

Hongyu, J., Xia, L., Yan, C., 2010. The research of a distributed simulation method of information sharing in supply chain. In *Proceedings of 2010 International Conference on Optoelectronics and Image Processing (ICOIP 2010)*, Haiko, Hainan, China, 11–12 November, pp. 596–599.

Huang, G.Q., Lau, J.S.K., Mak, K.L., 2003. The impacts of sharing production information on supply chain dynamics: a review of the literature. *International Journal of Production Research* 41, 7, 1483–1517.

Iannone, R., Miranda, S., Riemma, S., 2007. Supply chain distributed simulation: an efficient architecture for multi-model synchronization. *Simulation Modelling Practice and Theory* 15, 3, 221–236.

IEEE 1516 2010. *IEEE standard for modelling and simulation (M&S) high level architecture (HLA)*. Institute of Electrical and Electronics Engineers, New York.

Jain, S., Riddick, F., Craens, A., Kibira, D., 2007. Distributed simulation for interoperability testing along the supply chain. In *Proceedings of the 2007 Winter Simulation Conference*, Washington, DC, 9–12 December, pp. 1044–1052.

Johnson, G.D., 1999. Networked simulation with HLA and MODSIM III. In *Proceedings of the 31st Winter Simulation Conference*, ACM Press, New York, NY, 5–8 December, pp. 1065–1070.

Karlsson, M., Olsson, L., 2001. pRTI 1516—rationale and design. In *Proceedings of the 2001 Fall Simulation Interoperability Workshop*. 01F-SIW-038. Simulation Interoperability Standards Organization, Orlando, FL, 9–14 September.

Katsaliaki, K., Brailsford, S., 2007. Using simulation to improve the blood supply chain. *Journal of the Operational Research Society* 58, 2, 219–227.

Lee, S., Wysk, R.A., 2004. A top-down mapping and federated-based coordination for supply chain management systems using an ERP system. In *Proceedings of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics*, Orlando, FL, 18–21 July, pp. 325–330.

Lendermann, P., 2006. About the need for distributed simulation technology for the resolution of real-world manufacturing and logistics problems. In *Proceedings of the 38th Winter Simulation Conference*, Winter Simulation Conference, Monterrey, CA, 3–6 December, pp. 1119–1128.

Lendermann, P., Turner, S.J., Low, M.Y.H., Gan, B.P., Julka, N., Chan, L.P., Cai, W.T., Lee, L.H., Chew, E.P., Teng, S.Y., McGinnis, L.F., 2007. An integrated and adaptive decision-support framework for high-tech manufacturing and service networks. *Journal of Simulation* 1, 2, 69–79.

Li, J., Yuan, A., Wu, Q., 2010. A framework of simulation for cluster supply chain collaboration. In *Proceedings of the 2010 International Conference on Internet Technology and Applications*, Proceedings, art. no. 5566100, Wuhan, China, 21–23 August.

Linn, R.J., Chen, C.S., Lozan, J.A., 2002. Development of distributed simulation model for the transporter entity in a supply chain process. In *Proceedings of the 34th Winter Simulation Conference*, IEEE Press, Piscataway, NJ, 8–11 December, pp. 1319–1326.

McLean, C., Riddick, F., 2000. The IMS MISSION architecture for distributed manufacturing simulation. In *Proceedings of the 32nd Winter Simulation Conference*, Orlando, FL, 10–13 December, pp. 1539–1548.

Mertins, K., Rabe, M., Jäkel, F-W., 2005. Distributed modelling and simulation of supply chains. *International Journal of Computer Integrated Manufacturing* 18, 5, 342–349.

Mustafee, N., Taylor, S.J.E., 2006. Investigating distributed simulation with COTS simulation packages: experiences with Simul8 and the HLA. In *Proceedings of the 2006 Operational Research Society Simulation Workshop (SW06)*, Operational Research Society, Birmingham, UK, 28–29 March, pp. 33–42.

Mustafee, N., Taylor, S.J.E., Katsaliaki, K., Brailsford, S., 2009. Facilitating the analysis of a UK national blood service supply chain using distributed simulation. *SIMULATION: Transactions of the Society of Modelling and Simulation International* 85, 2, 113–128.

Pannirselvam, G.P., Ferguson, L.A., Ash, R.C., Siferd, S.P., 1999. Operations management research: an update for the 1990s. *Journal of Operations Management* 18, 1, 95–112.

Pidd, M., Carvalho, M.A., 2006. Simulation software: not the same yesterday, today or forever. *Journal of Simulation* 1, 1, 7–20.

Rabe, M., Jäkel, F.W., 2003. On standardization requirements for distributed simulation in production and logistics. *Building the Knowledge Economy*, IOS Press, Twente, The Netherlands, pp. 399–406.

Robinson, S., 2005. Discrete-event simulation: from the pioneers to the present, what next? *Journal of the Operational Research Society* 56, 6, 619–629.

SISO-STD-006-2010. 2010. *Standard for COTS Simulation Package Interoperability Reference Models*. Simulation Interoperability Standards Organization, Orlando, FL.

Stevens, G.C., 1989. Integrating the supply chain. *International Journal of Physical Distribution & Materials Management* 19, 8, 3–8.

Straßburger, S., 2001. *Distributed Simulation Based on the High Level Architecture in Civilian Application Domains*. Society for Computer Simulation International, Ghent, Belgium.

Straßburger, S., Schulze, T., Lemessi, M., 2007. Applying CSPI reference models for factory planning. In *Proceedings of the 39th Winter Simulation Conference*, IEEE Press, Piscataway, NJ, 9–12 December, pp. 603–609.

Straßburger, S., Schulze, T., Fujimoto, R., 2008. Future trends in distributed simulation and distributed virtual environments: results of a peer study. In *Proceedings of the 2008 Winter Simulation Conference*, Miami, FL, 7–10 December, pp. 777–785.

Taejong, Y., Kyungdoc, K., Sunyoung, S., Hyunbo, C., Yucesan, E., 2009. Applying web services technology to implement distributed simulation for supply chain modelling and analysis. In *Proceedings of the 2009 Winter Simulation Conference*, Austin, TX, 13–16 December, pp. 863–873.

Tammineni, S., Venkateswaran, J., 2007. Advanced Look-ahead Based Approach (ALBA) for distributed simulation of supply chains. In *Proceedings of the 2007 IEEE International Conference on Industrial Engineering and Engineering Management*, Singapore, 2–5 December, pp. 1838–1842.

Taylor, S.J.E., Bohli, L., Wang, X., Turner, S.J., Ladbrook, J., 2005. Investigating distributed simulation at the ford motor company. In *Proceedings of the 9th International Symposium on Distributed Simulation and Real-Time Applications*, IEEE Computer Society, Washington, DC, 10–12 October, pp. 139–147.

Taylor, S.J.E., Mustafee, N., Straßburger, S., Turner, S.J., Low, M.Y.H., Ladbrook, J., 2007. The SISO CSPI PDG standard for commercial off-the-shelf simulation package interoperability reference models. In *Proceedings of the 2007 Winter Simulation Conference*, Winter Simulation Conference, Piscataway, NJ, 9–12 December, pp. 594–602.

Taylor, S.J.E., Robinson, S., 2006. So where to next? A survey of the future for discrete-event simulation. *Journal of Simulation* 1, 1, 1–6.

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.

Turner, S.J., Cai, W., Gan, B.P., 2000. Adapting a supply-chain simulation for HLA. In *Proceedings of the 4th IEEE International Workshop on Distributed Simulation and Real-Time Applications (DS-RT'00)*, San Francisco, CA, 24–26 August, pp. 71–78.

US Department of Defense Modelling and Simulation Office. 1999. High level architecture run-time infrastructure RTI 1.3-next generation programmer's guide. US Department of Defence Modelling and Simulation Office, USA.

Wang, X., Turner, S.J., Low, M.Y.H., Taylor, S.J.E., 2006. COTS Simulation Package (CSP) interoperability—A solution to synchronous entity passing. In *Proceedings of the 20th Workshop on Principles of Advanced and Distributed Simulation*, Singapore. 24–26 May, pp. 201–210.