Distributed Simulation: State-of-the-Art and Potential for Operational Research

Simon J E Taylor
Modelling & Simulation Group
Department of Computer Science
Brunel University London
Uxbridge, Middx, UK UB8 3PH
simon.taylor@brunel.ac.uk
Corresponding author

In Operational Research conventional simulation practices typically focus on the conceptualization, development and use of a single model simulated on a single computer by a single analyst. Since the late 1970s the field of Distributed Simulation has led research into how to speed up simulation and how to compose large-scale simulations consisting of many reusable models running using distributed computers. There have been significant advances in the theories and technologies underpinning Distributed Simulation and there have been major successes in defence, computer systems design and smart urban environments. However, from an Operational Research perspective, Distributed Simulation has had little impact on mainstream research and practice. To argue the potential benefits of Distributed Simulation for Operational Research, this article gives an overview of Distributed Simulation approaches and technologies as well as discussing the state-of-the-art of Distributed Simulation applications. It will investigate the potential advantages of Distributed Simulation for Operational Research and present a possible sustainable future, based on experiences from e-Science, that will help Operational Research meet future challenges such as those emerging from Big Data Analytics, Cyber-physical systems, Industry 4.0, Digital Twins and Smart environments.

Keywords: Simulation, Distributed Simulation, Operational Research, e-Science, Big Data, Industry 4.0

1. Introduction

Distributed Simulation (DS) is a field with roots in Computer Science, especially parallel and distributed computing, and has contributed to major successes in the simulation of large systems in defence, computer systems design and smart urban environments. As noted by Fujimoto (1990; 2016) the field emerged from two communities. In the 1970s, the *Parallel Discrete Event Simulation* (PDES) community investigated how to speed up simulations using multiple processors in high performance computing systems. Later, in the 1980s, principally led by efforts in the defence sector to enable simulation reuse, the *DS* community used PDES techniques to interconnect simulations together running on distributed computers connected by a network. Today, researchers active in these areas informally call the field *Parallel and Distributed Simulation (PADS)*. However, as the wider simulation community often refers to this as just *DS*, we shall therefore use this term in this review.

The main goals of DS are to use parallel and distributed computing techniques and multiple computers to speed up the execution of a simulation program and/or to link together simulations to support reusability (Fujimoto, 2000). Some authors have also used DS to refer to approaches that run simulation experiments and/or replications on distributed computers in parallel with the goal of reducing the time taken to analyse a system (Heidelberger, 1986). Following the "modes" of simulation introduced by Robinson (2002), Figure 1 shows these three main modes of DS: **Mode A** to speed up a single simulation, **Mode B** to link together and reuse several simulations, and **Mode C** to speed up simulation experimentation. In Mode A, a simulation that might be simulated on a single

computer is subdivided on some basis into separate simulations that are run on different computers interacting via a communications network – the possibility of speed up arises from the parallel execution of the separate simulations. In Mode B, several simulations running on different computers are linked together to form a single simulation again with interactions between models carried out via a communications network – larger models beyond the capability of a single computer can be created and simulations can be reused by connecting them to other simulations (so potentially reducing the cost of developing new simulations). In Mode C, experiments carried out sequentially, one-at-a-time, on a single computer are instead run in parallel using multiple computers coordinated by some experimentation manager via a communication network – the parallel execution of simulation runs can therefore significantly reduce experimentation time or significantly increase the number of simulation experiments possible in the same timeframe.

From an Operational Research (OR) perspective, simulation is a key technique in predictive and prescriptive analytics (Lustig, et al. 2010) as it allows different possible scenarios to be analysed by creating a model of a system and experimenting with it under different conditions (Law, 2015; Robinson, 2014). Conventional simulation practice in OR typically involves a single analyst building a single model and performing simulation experiments in sequence on a single computer (e.g. Proudlove, et al. 2017; Gogi, Tako, & Robinson, 2016; Tako & Kotiadis, 2015; Robinson, et al. 2012). Even though a single computer can be quite powerful, especially if it has multiple cores or CPUs, it places practical limitations in terms of processing and time. Arguably, this also means that "conventional" simulation practitioners are not taking advantage of exciting and significant innovations in parallel and distributed computing that are being exploited elsewhere.

For example, many scientists use grid computing or e-Infrastructures, integrated collections of computers, data, applications and sensors across different organizations to support international scientific projects (Foster et al., 2001; Bird, Jones, & Kee, 2009). Cloud computing offers relatively low cost on-demand scalable computing power (Mell & Grance, 2011). Cloud is being used to create affordable high performance Big Data and Analytics applications in a similar way to grid computing, especially when appropriate IT infrastructure does not exist locally (Yang, et al., 2017). Novel techniques to explore the "sensed environment" have given rise to the Internet of Things (IoT) and Cyber-physical systems (Atzori, Iera, & Morabito, 2010). The combination of these with cloud computing and Big Data Analytics are fuelling new Cyber-physical systems such as "Smart" applications in society and industry (Cocchia, 2014), Digital Twins (Tao et al., 2018) and new initiatives such as Industry 4.0 (Trappey, et al., 2017). Some of these applications need rapid responses from a computing infrastructure for monitoring and control purposes. Edge and Fog distributed computing have recently emerged to complement cloud computing to support these applications (Chiang & Zhang, 2016). In turn, Jungle computing has emerged as a collective term for applications that use a combination of grid, cloud and associated technologies (Tychalas & Karatza, 2017).

As outlined above, DS has the potential to allow OR practitioners and researchers to build and simulate large models, to run experiments faster or to do more experimentation within a project. As some DS research investigates how advances in distributed computing can be used in these contexts, DS also has the potential of acting as a "gateway" to the exploitation of these technologies by OR researchers and practitioners. However, DS is largely a stranger to mainstream OR research and practice. To argue the potential benefits of DS, this article presents an overview of the approaches and technologies of DS. The state-of-the-art of Distributed Simulation applications is then presented with common themes, areas of success and barriers to wider use. Based on this, the article discusses the potential of DS for OR and, drawing from experiences in e-Science, suggests a possible sustainable future of DS for OR that will help OR to meet future challenges such as those

emerging from innovations in Big Data Analytics, Cyber-physical systems, Industry 4.0, Digital Twins and Smart environments.

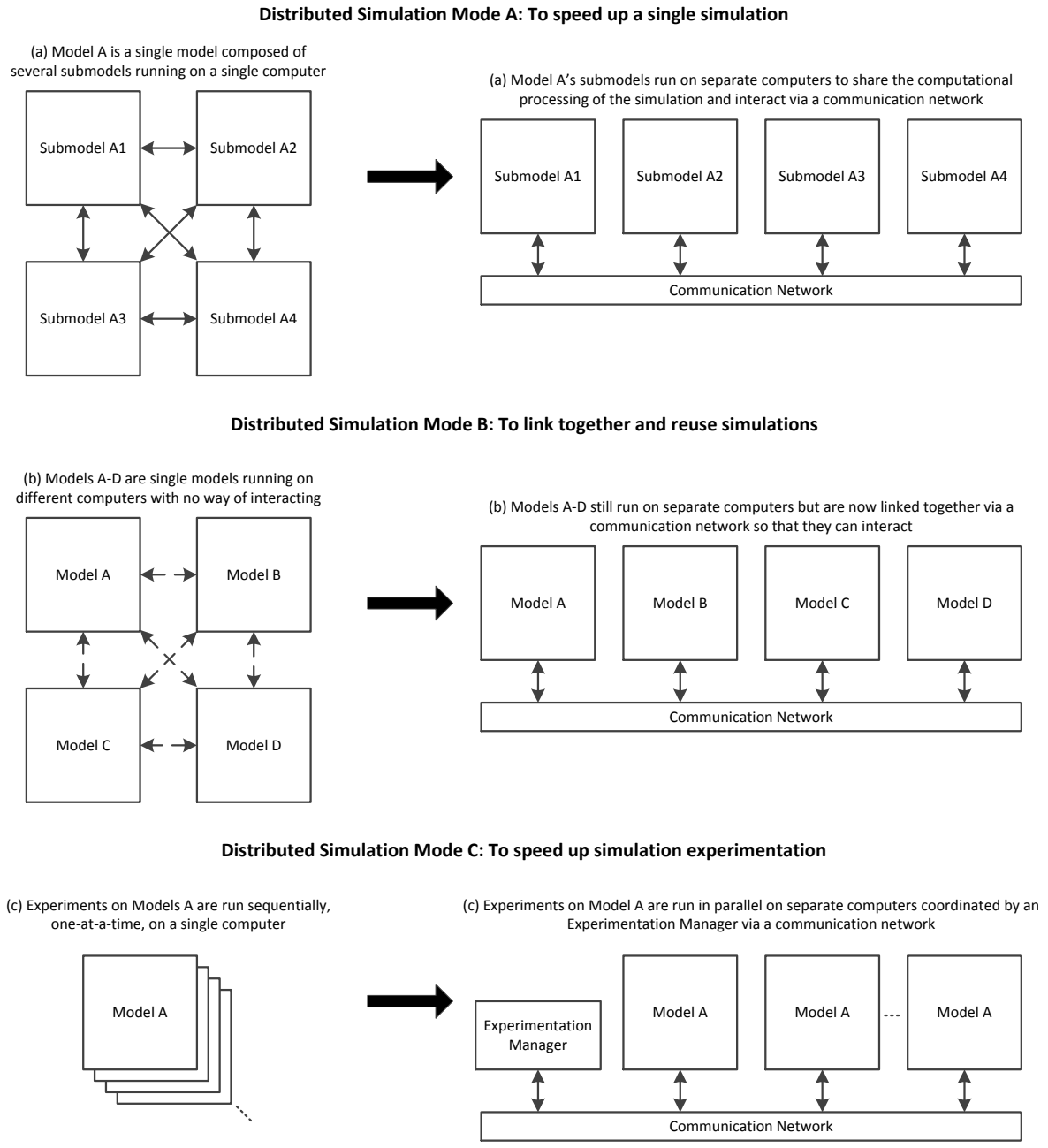

Figure 1: Modes of Distributed Simulation

This article is structured as follows. Section 2 gives an overview of DS is given in the next section with arguments for DS and an example of how DS works. Sections 3 and 4 present the main approaches to DS and the parallel and distributed computing technologies that have been used for DS. Section 5 reviews the current state-of-the-art of DS applications and shows the wide range of domains in which DS research is being applied. Based on this, and experiences of successful distributed computing initiatives in other areas such as e-Science, Section 6 discusses the potential

major impact of DS on OR and a possible sustainable future of DS in OR. Finally, Section 7 concludes the article with some key recommendations on how DS can grow to play a major role in OR.

2. An Overview of Distributed Simulation

The following section presents an overview of DS by first considering the arguments for DS and then an overview of how DS works.

2.1 Arguments for Distributed Simulation

What are the arguments for DS? These can be illustrated with the following supply chain example. Company A and B manufacture a range of widgets to sell to various consumers. Company A produces the widget bodies and Company B finishes these. The problem that both companies face is how to balance production so that Company A does not produce too much stock and Company B never starves of stock. It is assumed that both companies already use simulation to manage their own production processes. Both companies decide to create a single joint discrete-event simulation to manage balanced production. It is normal practice for simulations such as these to be developed in a commercial simulation package. For example, Company A's simulation might be developed in WITNESS (www.lanner.com) and Company B's simulation in Flexsim (www.flexsim.com). A Flexsim simulation cannot be "cut and pasted" into WITNESS or vice versa. If the scenario is changed slightly and both companies have simulations coded in a simulation language such as REPAST using JAVA (repast.github.io), then a single simulation could be created. However, integrating the two simulations would take time and most likely require some re-coding effort (e.g. clashing variable/method names, different data structures and sources, common simulation executive, etc.) When the new simulation has been created, both companies can now see the inner workings of each other, the detail at which each has simulated its processes and any assumptions that have been made. The data each company uses in its simulation will also be revealed. Assuming two companies are happy to share detail and data at this level, another issue arises. The new supply chain simulation and its supporting data is now distinct to the two separate company simulations and data. Any changes to a local simulation or data will need to be coordinated with the new simulation. Figure 1 (b) shows the far simpler alternative presented by DS where simulations are kept separate but run together. On this basis several authors have identified arguments for DS from an OR perspective (Mustafee, et al. 2012; Taylor, et al. 2012; Boer, de Bruin, & Verbraeck, 2009; Lendermann et al., 2007). These include:

Execution time. A large simulation can be slow to run. DS can be used to split the simulation across multiple computers to exploit parallel processing to speed up execution. DS may also allow simulation experimentation to be processed faster by using multiple computers.

Model composability and reuse. The development of a simulation can represent a significant investment in time and money. When building a new simulation it may be attractive to reuse a simulation as a sub-component. However, practical issues such as variable name clashes, variable type incompatibility, global variables and different verification/validation assumptions might mean extensive recoding and testing. Further, if the simulations have been developed in different simulation packages or languages then it might not be possible to combine them at all without starting from scratch. It may be more convenient to just link the simulations together as a DS.

Ownership and Management. Following the above, if a simulation has been composed from reused simulations then it may be difficult for a simulation owner or developer to update their simulation without having to update the entire simulation. DS allows simulations to be independently managed as they are still separate.

Privacy. Creating a single simulation from other simulations could also mean that the entire details of a simulation would be revealed to the developer of the single simulation. If a simulation contains secrets (e.g. the confidential inner workings of a factory, hospital or military system) then these would be visible to anyone running the newly composed simulation. DS preserves this separation and allows simulations to be composed from "black boxed" simulations.

Data integrity and privacy. Similar to the above problems is the issue of data integrity and privacy. If a simulation requires access to a specific database then when a new simulation is created that data may have to be copied to allow the new simulation to access it. This data may be confidential. Another issue is how can the integrity of the data be preserved (how can the copy be kept up-to-date)? DS allows data to remain with the owning simulation and therefore avoids this issue.

Hybrid simulation. There are very few commercial simulation packages that support hybrid simulations consisting of discrete-event, agent-based and/or system dynamics elements. DS allows simulations of these different types to be linked together.

To balance the above it must be noted that simply dividing a simulation across different processors does not automatically guarantee speed up. The overhead of coordination between the simulations could result in a slower execution than a single simulation. Similarly, reusing a simulation as part of a DS is also not automatic, especially if that simulation was not designed for that purpose (Robinson, et al. 2004). Currently, no commercial simulation package possesses "off-the-shelf" DS functionality. These need to be modified to enable models to interact over a communication network (which can be complex). Also, to realise a successful DS for OR project, knowledge is needed that combines DS, OR, computer networking, advanced computing technologies and significant programming experience. However, as will be seen in Section 5, there are many examples of where DS has yielded success and, in Section 6, investment in DS research and development may have significant impact on OR.

2.2 An Overview of How Distributed Simulation Works

How does DS actually work? In DS Mode A and B, speeding up a simulation or linking simulations together, the main issue in DS is the coordination of simulation time across distributed computers. Approaches to this have been derived from the coordination and synchronization of software processes running on different distributed computers (Tanenbaum & van Steen, 2007). One class of algorithms deal with how computers can synchronise their physical clocks across a communications network to coordinate the real-time execution of software processes. Building on this approach to coordination, rather than synchronising actual physical clocks, another class of algorithms focus on the logical ordering of actions across the computers in a distributed system. If there is no interaction between two computers in a distributed system there is no point synchronising clocks. However, the actions of the two computers might indirectly effect each other via interactions with other computers. Therefore, actions need to be ordered over the entire distributed system. Lamport (1978) described an algorithm that used *logical clocks*, action *timestamps* and the *happens-before* relation to impose a partial action ordering over a distributed system.

In a DS, several simulations interact with each other via event messages (i.e. if one simulation affects another then a timestamped event message is sent to represent this effect). Lamport's algorithm formed the basis for event message ordering, or *time management*, in DS. As not all simulations interact with each other all the time, a major goal in DS is to take advantage of this to ensure that all events are processed in the correct order (at least by the end of the simulation run) while maximising the amount of parallel execution of the simulations (to take advantage of the processing provided by multiple computers by reducing the amount of synchronization needed). To illustrate

the time management problem, consider the simple DS of Figure 2. This simulation consists of three single server queues (SSQ). SSQ1 and SSQ2 both have a full FIFO queue of entities. An entity is processed according to a server's processing time and then sent to SSQ3. Each SSQ is "mapped", or executed, on a different computer (i.e. there are three computers). Each SSQ simulation has all it needs to run a discrete-event simulation (state, event routines, event list, clock, simulation executive, etc.) In a DS, to represent the transfer of an entity from one SSQ to another SSQ, the originating simulation will send a timestamped event message to the destination simulation. For example, SSQ1 and SSQ2 are processing entities every 10 and 12 minutes respectively. When each server finishes processing it sends its entity to SSQ3 (no travel time).

Starting the simulation at zero and assuming that SSQs 1 and 2 never run out of entities to process, entities will arrive in the following order at SSQ3: e1 at 10 (time t1), e2 at 12 (time t2), e3 at 20 (time t3), e4 at 24 (time t4), e5 at 30 (time t5), etc. (i.e. t1<t2<t3…). These interactions are represented by timestamped event messages (the diagram shows two messages TEM1 and TEM2 representing the transfer of e1 and e2). Given that there is no single simulation clock coordinating time across the DS, we cannot make any assumptions of what time the SSQ3 simulation has reached when an event message is sent. There therefore needs to be some coordination algorithm that will ensure, at least by the end of the simulation, that all events are processed in the correct order. In this case, TEM1 represents the arrival of e1 at SSQ3 at t1 and TEM2 represents the arrival of e2 at SSQ3 at t2. We must ensure that the SSQ3 processes TEM1 and TEM2 in that order otherwise the simulation will have processed the events out of order.

Typically in DS we assume that a simulation is composed of a set of Logical Processes (LPs) that represent different parts of the simulation (e.g. the SSQs of the above example)[1]. LPs communicate via timestamped messages. The goal of a time management algorithm is to ensure that, by the end of the simulation, all LPs have processed their events in timestamp order (both "internal" events produced by the LP and "external" events represented by timestamped messages arriving at the LP). This is termed the *local causality constraint* and it can be shown that if all LPs maintain this constraint, the DS will be causally correct. This approach is essentially the basis for *conservative* approaches to time management. An alternative to this is to let LPs process events as soon as they are available and to recover dynamically from any causality errors. This is the basis for *optimistic* approaches. Note that the decomposition (division) of a simulation into LPs can be critical to the overall execution performance of a DS in that a balance has to be made between parallel execution and communication (Righter & Walrand, 1989). There is a third type, *real-time* approaches, that has emerged from efforts to standardise reuse and interoperability in defence applications that focus less on the management of time and more on simulating a real world training experience. The next section presents work on these approaches and underlying technologies.

[1] In this discussion it is assumed that the LPs are running on separate computers. These techniques were originated created on high performance computer systems in the 1970s and 1980s where LPs ran on multiple processors in shared memory high performance computers. This message passing approach meant that it was straightforward to move to distributed computers. In today's powerful computers that have multiple processors (i.e. multiple cores), it is again relatively straightforward to run LPs on each processor. A DS can therefore run on distributed computers, the multiple cores of a single computer, and on combinations of the two.

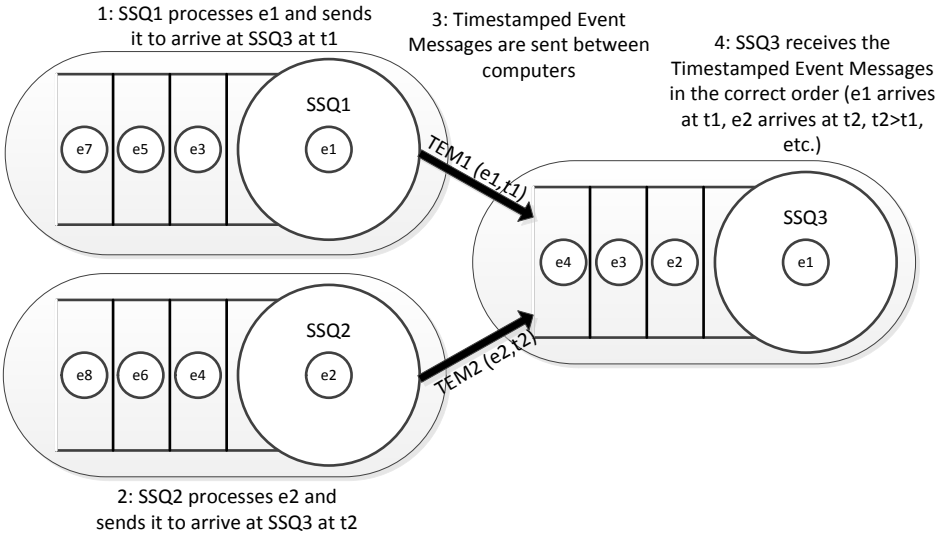

Figure 2: Simple Distributed Simulation

In DS Mode C, where DS can be used to speed up experimentation, the main issue is how to distribute and coordinate the execution of simulation experiments on different computers. Arguably, the need to speed up simulation experimentation is far more common than the need to speed up the simulation of a single model or to link several models together. Heidelberger (1986) was one of the first researchers to propose a theoretical foundation for the parallel execution of simulations. While the use of many computers to process experiments quickly in parallel is attractive, it does not give infinite processing and care must still be taken in identifying the right experiments to process. Further, time is taken to set up a distributed computer with the software needed to process an experiment, to send the simulation and associated parameters and data for processing, to receive the results and to manage the overall process. These are common problems in many scientific disciplines where some form of computing task needs speeding up by running multiple instances of the same program in parallel. As will be explored later, this was one of the concepts behind the emergence of "The Grid" (Foster et al., 2001) and "e-Science" (Hey & Trefethen, 2005), both of which have had widespread impact on scientific endeavour. Approaches and technologies from these have inspired emerging simulation technologies.

3. Distributed Simulation Approaches

The approaches to DS Modes A and B, where techniques are used to either speed up a simulation or to link simulations together, can be split into three types: conservative, optimistic and real-time. Approaches to Mode C will be discussed in the Distributed Simulation Technologies section.

3.1 Conservative Approaches

The conservative class of algorithms were the first to be developed in the late 1970s (Bryant, 1977; Chandy & Misra, 1979; Misra, 1986). These algorithms are conservative in the sense that they avoid time management errors. In the above example, if SSQ3 is at time 10 and has just processed TEM1 representing the arrival of e1 at T1 and it has received TEM2 representing the arrival of e2 at T2 (time 12) then under a conservative algorithm it will not process TEM2 until it knows it is safe to do so (i.e. no event will be scheduled before time 12). The approach taken by this early work assumed that the LPs and their interconnection (topology) was fixed and that each LP only sent messages in increasing timestamp order (and it was assumed that the communication network delivered messages in the order sent). An LP would therefore have a set of input queues for messages from

each LP. Each input queue would have a set of messages in increasing timestamp order. The process to pick the next safe message to process would therefore be a cycle of repeatedly checking the time of the next message in each queue and then processing the earliest one (advancing the simulation clock and executing this next event as one would do in a "normal" sequential simulation).

As it is possible for deadlock to occur in this scheme (caused by a network of empty input queues at LPs waiting in a chain for each other to act), timestamped *null* messages are sent as guarantees of behaviour. When an LP attempts to process a message, if the earliest timestamped message is a null message, the LP merely advances its simulation clock to that time. This might have the effect of enabling that LP to act (e.g. by sending a timestamped event message representing an entity leaving the LP as a result of an end of service event). If that LP cannot act then it will send a timestamped null message to all its connected LPs that will allow them to advance time. Key to this (and to performance) is *lookahead* which is the smallest time advance guarantee that an LP can make (e.g. a minimum service time). Relatively poor lookahead can lead to an excessive exchange of messages (and associated processing of large queues of null messages) and poor speedup as much of the processing time is consumed with the processing of null messages. There have been many variants proposed to improve the effectiveness of conservative algorithms, especially with respect to alternative approaches to null message generation and extracting better lookahead from application specific details. For example, broadcasting (Peacock, Manning, & Wong, 1980), shared resources (Reynolds, 1972), appointments (Nicol & Reynolds, 1984), LP interconnection topology (Kumar, 1986), bounded lag (Lubachevsky, 1989) and conditional lookahead (Fu, Becker, & Szczerbicka, 2014). Work continues in this area as researchers seek to produce more effective time management approaches for specific applications or new mapping/partitioning algorithms to load balance the processing of a DS.

3.2 Optimistic Approaches

In contrast to conservative algorithms, optimistic algorithms allow causality errors to occur and then recover from them so that all events are correctly processed in order by the end of the simulation. The aim is to generate more parallelism to process the simulation faster. Jefferson (1985) introduced *virtual time* for distributed systems that took inspiration from Lamport's clocks and approaches for virtual memory handling. Applying this to DS resulted in the *Time Warp* mechanism, the most well-known optimistic algorithm. In this LPs have three queues: input message queue, output message queue and a state queue. Event messages are sent between LPs and are stored in the input queue. Event messages are processed as they arrive in the hope that they will arrive in the correct order. Each time an event message is processed a new state is made (reflecting the current state of the LP at time t) and stored in the state queue. If new event messages are generated then these are stored in the output queue and sent to their destination LPs. If an out of order event message is processed then the LP returns to the last safe state prior to the timestamp of the event message being processed (this is termed a "rollback"). Copies of output messages stored in the output queue with timestamps greater than the time of the current new state are then sent with negative signs (termed "antimessages"). If an antimessage arrives in an input queue it causes the receiving LP to rollback to the last current safe time and in turn to send its antimessages. Once antimessages are sent the LP processes the now correctly ordered event message in its input queue and continues as before.

Major issues with optimistic approaches include the storage and access of large numbers of simulation states and rollback cascades. A protocol to establish *Global Virtual Time* (GVT) can be used to establish a lower bound on global time – this means that LPs will not rollback to an earlier time and therefore simulation states with an earlier timestamp can be eliminated. As with conservative algorithms many variants have been proposed. Many of these focus on reducing the number of rollbacks and cascades, the need to reprocess event messages (for example when an

arriving event message does not change the simulation state) and more efficient methods of calculating GVT. Examples include lazy cancellation (Gafni, 1988), lazy re-evaluation (Fujimoto, 1990), restricted rollback (Damani, Wang, & Garg, 1997), adaptive state saving (Rönngren & Ayani, 1994; Lin, et al. 1993), message aggregation (Chetlur, et al. 1998), global checkpointing (Moreira, Santana, & Santana, 2005) and speculative computing (Venu & Joe, 2014). Some attempts have also been made to control unconstrained rollback by combining optimistic and conservative approaches (e.g. Breathing Time Buckets (Steinman, 1992)).

3.3 Real-Time Approaches

Both conservative and optimistic approaches support the analysis of systems using some form of discrete-event simulation. An alternative to this is real-time simulation where a simulation attempts to respond as if it is the real world (or at least give a real-time response to a user's actions). In a DS multiple simulations need to interact with multiple "players" and their own responses in as close to real-time as possible. This goal requires different approaches to balancing processing and communication. These are classed as "Real-Time Approaches" and do not use time management as described above.

Real-time simulations, or distributed virtual environments, were motivated by the need to connect, and reuse, expensive military simulations and simulators for training. The SIMulator NETworking (SIMNET) Project was funded by DARPA in 1983 (and subsequently by the US Army) and represents a major milestone in the evolution of real-time DS. The project created a network of around 250 simulators, installed at nine operational training sites (five in the US and four in Europe). This started the first major standardization effort in DS. The way many distributed systems technologies (such as the Internet, the World Wide Web and many applications) operate and communicate is defined by various standards. These standards tend to define the communications protocol ("how we speak to each other"), the format of the data transmitted ("what we say to each other") and any supporting software needed by the technology. The communications protocols developed in SIMNET became the Distributed Interactive Simulation (DIS) Standard Protocol IEEE standard 1278-1993 (IEEE, 1993). This was approved on March 17, 1993 and was later superseded by newer versions and extensions. Related work on the Aggregate Level Simulation Protocol (ALSP) (Wilson & Weatherly, 1994) and general experiences from research on conservative and optimistic mechanisms, formed the basis for the development of a more ambitious standard for DS.

Based on these efforts, in 1996 the Defense Modeling and Simulation Office (DMSO) produced an initial proposal for the High Level Architecture (HLA), a standard to support the reuse and interoperation of distributed simulations (Dahmann, Fujimoto, & Weatherly, 1997). This was formally ratified as the IEEE 1516-2000 Standard for Modeling and Simulation (M&S) High Level Architecture in 2000 (updated in 2010). The HLA is actually a suite of standards. These include the definition of a framework and rules (IEEE, 2010a), the definition of supporting software (the Runtime Infrastructure (RTI)) (IEEE, 2010b), data definitions (the Object Model Template (OMT) (IEEE, 2010c)) and a development process (IEEE, 2010d). The suite of standards are curated and evolved by an IEEE working group and the Simulation Interoperability Standards Organization Standards Activity Committee (SISO SAC)[2].

It is normal for a set of standards to have a supporting community. SISO has also produced several standards in support of the HLA. The majority are defence-specific (e.g. data formats for military vehicles, weapons, etc.) Two have been developed with OR applications in mind. The SISO-STD-006-

[2] Note that standards play a critical role in distributed computing as they define how software communicates over a network (i.e. data and message format) and how software interfaces will supporting software (i.e. middleware).

2010 Standard for COTS Simulation Package Interoperability Reference Models (IRMs) was developed to standardise requirements of OR/MS interoperability problems (SISO, 2010a; Taylor et al., 2012). The SISO-STD-008-2010 Standard for Core Manufacturing Simulation Data (CMSD) was developed to provide a standardized data interface for manufacturing systems federations (SISO, 2010; Strassburger & Taylor, 2012). The HLA also has facilities for discrete time-based simulations with support for conservative and optimistic time management. As shown in figure 3, HLA-based DS is organized slightly differently to one using LPs. Instead of LPs, each simulation is a *federate* and the collection of interacting (or interoperating) simulations is a federation. The data that each simulation can send and expects to receive is defined in the federate's Federate Object Model (FOM) based on the general OMT. In Figure 3, Federate F1 has integer state variables s1, s2 and s3 that represent, for example, the position of three tanks. F1 performs the simulation of the tank positioned at s1. F2 and F3 are responsible for the simulation of their tanks at s2 and s3, respectively. The FOM for each federate with define the variables and their types. Federates interact through the RTI and the RTI is responsible for the overall synchronization and control of the federation. In distributed computing terms, all the common computing and communication services are placed in the RTI middleware. A limited number of commercially available RTI middleware and open source RTI middleware are available.

Simplistically, a real-time DS works in the following way. In the above real-time wargame distributed simulation each federate simulates its own tank and reacts to the positions of the others. Before performing its simulation for the next time period a federate will "ask" the RTI if there are any updates from the other federates (as defined by the FOM). In the case of Federate F1 it will therefore ask if there are updates on the position of the other two tanks (s2 and s3). If there are updates, the RTI passes these to the federate and then that federate updates its simulation (e.g. the new positions of s2 and s3). The federate will then perform its simulation and then informs the RTI of the new position of its tank (the new value of s1). The other federates will do this in parallel at the same time. In the HLA, depending on the needs of the DS, this interchange of information can be coordinated in several ways with respect to real-time progress (Fujimoto, 1998).

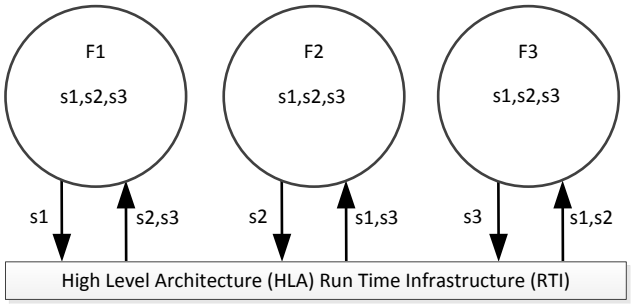

Figure 3: A Distributed Simulation using the High Level Architecture

4. Distributed Simulation Technologies

4.1 Modes A and B: Speeding Up and/or Linking Simulations

There have been attempts to create software libraries or Application Programming Interfaces (APIs) to simplify the development of DS. Early examples include the Maisie discrete-event simulation language that supported sequential or DS (conservative or optimistic) (Bagrodia & Liao, 1994) and the APOSTLE language that enabled the development of optimistic DS (Wonnacott & Bruce, 1996). More recently, Falcone, et al. (2016) introduced the HLA development kit framework that uses Java annotations to reduce the development time of HLA-based DS. Alternative approaches combine

specialist programming language extensions with operating system-level support for advanced computer architectures (e.g. supercomputers). These include the Georgia Tech Time Warp (GTW) system (Das, et al., 1994), WarpIV (based on the Synchronous Parallel Environment for Emulation and Discrete Event Simulation (SPEEDS) framework) (Steinman, 2005) and Rensselaer's Optimistic Simulation System (ROSS) (Carothers, Bauer, & Pearce, 2002). This complements research into producing DS algorithms designed to execute on specific high performance computing architectures (Gan, et al., 2001; Liu, 2013).

Research in this area continues on many different themes including balancing the processing load (De Grande & Boukerche, 2011; Alghamdi, De Grande, & Boukerche, 2016; Alkharboush, De Grande, & Boukerche, 2014), process mapping (Peschlow, Honecker, & Martini, 2007), and managing the amount of data transferred between LPs/federates (Morse, Bic, & Dillencourt, 2000; Raczy, Tan, & Yu, 2005). There has also been a similar theme that has focussed on the development of DS technologies for distributed agent-based simulation that combines both distributed computing and high performance computing (REPAST HPC (Collier, Ozik, & Macal, 2015), D-MASON (Cordasco et al., 2013), JADE (jade.tilab.com), PDES-MAS (Suryanarayanan, Theodoropoulos, & Lees, 2013), etc.) Researchers have studied DS load balancing issues (Lee, Park, Song, & Youn, 2012; Suryanarayanan & Theodoropoulos, 2013). General approaches to hybrid DS consisting of agent-based and discrete-event simulations have also been proposed (Nouman, Anagnostou, & Taylor, 2013; Anagnostou & Taylor, 2017). More examples of hybrid simulation will be given in the Applications section.

Some researchers have attempted to exploit advances in software engineering for DS. Bocciarelli, D'Ambrogio & Fabiani (2012) investigated how SysML (Systems Modelling Language), the UML-based general purpose modelling language for systems engineering, could be used to assist in the creation of DS. Similar investigations have been carried out into Functional Mock-up Interface (FMI) (Awais, Cvetkovic, & Palensky, 2017), Layered Architectures (Topçu & Oğuztüzün, 2013) and Software Patterns (Möller, Antelius, & Karlsson, 2013). Several authors have attempted to use programming language-dependent features to reduce the complexity of DS implementation, particularly with respect to the HLA (Santos, Leal, & Chiroque, 2013; Falcone et al., 2016; Van Tendeloo & Vangheluwe, 2015). Anagnostou & Taylor (2017) have developed a combined OR/MS and DS methodology that extends contemporary OR simulation approaches with DS features.

Advances in DS often reflect contemporary computing technologies of the time with research appearing two to three years after the appearance of the technology. The object-like structures with defined message communication interfaces of DS LPs and federates were reconceptualized in object broker architectures such as the Common Object Request Broker Architecture (CORBA) (D'Ambrogio & Gianni, 2004) and server-based architectures such as web services, grid services and service-oriented architectures (Pullen et al., 2004; Lendermann et al., 2005; Theodoropoulos et al., 2006; Chen, Cai, Turner, & Wang, 2006; Vanmechelen, De Munck, & Broeckhove, 2012; Xie, Teo, Cai, & Turner, 2005; Al-Zoubi & Wainer, 2013). Aspects of this work formed the basis for investigations into cloud computing for DS (Vanmechelen, De Munck, & Broeckhove, 2012; Fujimoto, Malik, & Park, 2010; Guan, De Grande, & Boukerche, 2016; D'Angelo & Marzolla, 2014; Chaudhry, et al. 2016). Initial work has studied the impact of Big Data Analytics technologies such as MapReduce and Hadoop on DS (Kim, et al. 2014).

Some of these efforts are leading to the development of Modelling & Simulation as a Service (MSaaS) that aims to create simulation services in the same standardized context as other developments in cloud-based services and systems (Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS)) (e.g. the CloudSME Simulation Platform (Taylor, et al., 2014) and cloud standards for MSaaS (Siegfried, et al. 2014). Bocciarelli, et al. (2013) continued previous work to investigate how SysML could be used to deploy a DS on cloud.

Researchers have also sought to maximise performance from multi-core CPUs (Liu & Wainer, 2012; De Munck, Vanmechelen, & Broeckhove, 2014) and GPU systems (Tang & Yao, 2013; Li, Cai, & Turner, 2016). Some have proposed dedicated computer hardware for DS (Lynch & Riley, 2009). D'Angelo & Marzolla (2014) discuss issues involved in scaling from multiple CPUs to cloud. Mobile devices have also been investigated as platforms for DS, especially in terms of energy consumption (Biswas & Fujimoto, 2016; Fujimoto & Biswas, 2016).

4.2 Mode C: Speeding Up Simulation Experimentation

In the past decade this work has focussed on approaches to using fixed computing resources (a *computing grid*) such as networks of PCs in an organization (a *desktop grid*) or using cloud-based (virtual) computers on demand. For example, (Mustafee & Taylor, 2009) developed the WINGRID desktop grid system that was used to speed up credit risk simulations in a well-known European bank. Experiences from this formed the basis for SAKERGRID (Kite, et al., 2011), a desktop grid and computing cluster system in use at Saker Solutions and Sellafield PLC, a cluster-based high performance simulation system in use in the Ford Motor Company and a desktop grid that was used for simulations of biochemical pathways in cancer (Liu et al., 2014). Choi, Seo, and Kim (2014) also developed a similar system for use with dedicated computing clusters. In terms of cloud-based systems, the JADES platform was adapted to run agent-based simulations in parallel on cloud resources (Rak, Cuomo, & Villano, 2012) and the CloudSME Simulation Platform has been used to run simulation experiments over multiple clouds (Taylor, et al., 2015). GridSpice (Anderson, Du, Narayan, & Gamal, 2014) also uses a cloud to speed up experimentation of distributed smart grid models. Yao, et al. (2017) reports on the parallel execution of multiple DS experiments. Their approach has been developed for specialized high performance computing systems. The Cloud Orchestration at the Level of Application (COLA) project[3] is developing an auto-scaling approach for simulation experimentation on cloud.

5. Distributed Simulation Applications

What is the current state-of-the-art of applications of DS? To capture this, a review was performed of publications appearing between 2010 and 2016 in DS (using SCOPUS and the search terms "Distributed Simulation" or "Parallel Simulation"). These were read for evidence of applications and then classified into areas (with some papers being disregarded (e.g. where "distributed" was used in the sense of distribution network analysis in supply chains or distributed systems in computer architectures and networks). Overall 246 papers were collated and of these 113 papers had some form of DS application. To get an overview of the state-of-the-art, the broad goals of applications were captured and classified by model type (agent-based simulation (ABS), discrete-event simulation, (DES), timestepped, continuous, real-time and hybrid), goal (Mode A (speeding up a single model), Mode B (linking models and reusability), Mode C (speeding up experimentation by running multiple simulations in parallel)), technique (time managed (conservative, optimistic, real-time) and technology focus (e.g. cloud, grid, CPUs, etc.)

[3] project-cola.eu

Table 1: Distributed Simulation Application by Area

Area	Model Type	Mode A Model Speedup	Mode B Model Linking/Reuse	Mode C Experimentation Speedup	Time Managed	Real-Time	Technology Focus	Publications
Computer Architecture and Control Systems	DES, Timestepped, Hybrid (Discrete/ Continuous)	X	X		X		General, Supercomputer	8 (7.1%)
Defence	Real-Time, some hybrid (Real-Time/DES)		X			X	HLA, Cloud, SOA	18 (16.0%)
Energy	ABS, Hybrid (Continuous, ABS)	X	X	X	X		General, Cloud	12 (10.6%)
Environment	Hybrid (DES, ABS, Continuous)		X		X		General	12 (10.6%)
Evacuation	ABS	X			X		General	9 (8.0%)
Healthcare	DES, Hybrid (DES/ABS)	X	X	X	X		General, Cloud	7 (6.1%)
Humanities	ABS	X			X		General	2 (1.8%)
Manufacturing and Supply Chain	DES		X	X	X		General, HLA, Grid, Cloud	28 (24.8%)
Maritime	Real-Time		X			X	General	5 (4.4%)
Networks	DES/ Timestepped	X			X		General	11 (9.7%)
Space	Real-Time		X			X	General	10 (8.8%)
Transportation/Traffic	DES, ABS, Hybrid (DES/ABS)		X		X		General, Cloud	11 (9.7%)

Table 1 shows DS by application area along with the split by proportion of publications. The following gives an overview of work performed in each application area (with illustrative references). Research into military or *Defence* applications of DS tends to follow two trends, either research into the development of real-time distributed training applications (e.g. McIntyre, Smith, & Goode, 2013; Tozzi & Zini, 2011; Zhang & Zhang, 2013) and/or research into improvements in the HLA and associated standards to support these applications (Möller et al., 2012; Prasad, Singh, Gangadhar, & Kumar, 2014; Wang, Gao, Wei, & Yin, 2012)). There is some hybrid work where real-time simulations are being linked to discrete-event simulations (Ha, Cha, Roh, & Lee, 2012). SISO plays a major role in standards development (via its Standards Activity Committee (SAC)) as does NATO (with its international working groups for M&S (the Modelling Simulation Groups (MSGs)). NATO is leading standardization work on Cloud and Service Oriented Architectures for DS in the area. There is also evidence of research being carried out in these technologies in this area (Li, et al., 2014).

There are several examples of where DS is being applied to link multiple hybrid models to investigate various problems in *Environment* (Hennicker & Ludwig, 2012; Yahiaoui & Sahraoui, 2012). *Energy* work mainly consists of approaches to link hybrid models (typically discrete/continuous) for different applications in power system technology design (Bottura et al., 2013) or agent-based simulations for power grid analysis (producer/consumer) (Pipattanasomporn, Feroze, & Rahman, 2009; Perkonigg, Brujic, & Ristic, 2013). GridSpice (Anderson et al., 2014) as noted earlier is being used to support distributed experimentation of distributed smart grid models using a cloud. A separate application of DS in this area is used to link models to analyse the maintenance requirements of offshore wind farms (Mustafee, Sahnoun, Smart, & Godsiff, 2015). Work in *Evacuation* focusses on approaches to speeding up the execution of large distributed agent-based simulations (or simulations with similar constructs to agents) of crowd behaviour or the evacuation of facilities (Dimakis, Filippoupolitis, & Gelenbe, 2010; Zia, Farrahi, Riener, & Ferscha, 2013).

Computer architectures and control systems investigate how to speed up the simulation of large scale designs (e.g. VLSI (Very Large Scale Integration) design (Gonsiorowski, Carothers, & Tropper, 2012; Tsirogiannis & Theodoropoulos, 2013) and nanomachines (Akkaya, Genc, & Tugcu, 2014)), or the hybrid simulation of Cyber-physical systems or embedded system design with discrete and continuous components (Brito et al., 2016; Garraghan, McKee, Ouyang, Webster, & Xu, 2016; Pfeifer, Gerstlauer, & Valvano, 2013). Some authors address extremely large-scale supercomputer design that in turn uses DS on supercomputers (Liu, 2013; Mubarak, Carothers, Ross, & Carns, 2012).

Healthcare applications have investigated the feasibility of large-scale healthcare simulation where multiple simulations (DES, ABS) of facilities are linked together or the speeding up of large models (Anagnostou, Nouman, & Taylor, 2013; Katsaliaki, Mustafee, Taylor, & Brailsford, 2009; Lee, Kang, & Prabhu, 2013). There are several examples of using DS in the sense of distributed access to a model used for some form of medical or surgical training (Khan, Khan, Dasgupta, & Ahmed, 2015; Kneebone et al., 2010). One paper used a multi-objective simulation optimization algorithm implemented on a cloud to speed up the experimentation of medical resource allocation in emergency departments (Chen, 2014). Work in *Humanities* used distributed agent-based simulation to study logistics issues in historical military scenarios (Craenen, Murgatroyd, Theodoropoulos, Gaffney, & Suryanarayanan, 2012).

The main theme of *Manufacturing and supply chains* is the linking together of models (Fujii et al., 2012; Hibino, Fukuda, & Yura, 2015; Medina, et al., 2013). Some work has focussed on the standard representation of data across a supply chain (Lin, et al., 2012; Long, 2014). Several authors have proposed ontologies to support model reuse in DS (Bell et al., 2008; Dragoicea, Bucur, Tsai, & Sarjoughian, 2012; Sarli, Leone, & Gutiérrez, 2016). Long (2016) presents a comprehensive (non-HLA)

DS for supply chain simulation. As noted earlier in this paper, some work has produced standards that have augmented the HLA for use in this area (Strassburger & Taylor, 2012; Taylor et al., 2012). Associated work has studied the transformation of Business Process Models into DS (P. Bocciarelli, Pieroni, Gianni, & D'Ambrogio, 2012). There are also some examples of distributed training where the access to a model is granted over a network (Bruzzone, Massei, & Bocca, 2012; Silvente et al., 2012). There are several examples where systems have been developed to speed up simulation, some of which use some form of parallel optimization (Frank, Laroque, & Uhlig, 2013; Zhang & Anosike, 2012). There is also some evidence of the use of e-Science / e-Infrastructure architectures to implement this (Kiss et al., 2014; Rossetti & Chen, 2012).

Maritime applications are similar to defence ones in that they focus on real-time access and reusability in the context of ports, ships and associated facilities (Dibbern, Hahn, & Schweigert, 2014; Henry et al., 2015; Massei, Tremori, Poggi, & Nicoletti, 2013). *Network* applications use DS to speed up large models of future networking systems such as wireless sensor networks, Mobile Ad-Hoc Networks (MANETs) and Wireless Vehicular Ad-Hoc Networks (Bononi, Di Felice, D'Angelo, Bracuto, & Donatiello, 2008; Huang, Alexopoulos, Hunter, & Fujimoto, 2012; Krzyszton & Niewiadomska-Szynkiewicz, 2016; Niewiadomska-Szynkiewicz & Sikora, 2012). DS to speed up large-scale models of the Internet have also been proposed (Coudert, et al., 2012).

Space applications also focus on real-time and reusability (Rabelo et al., 2013). As with defence there are standardization activities taking place through SISO on areas such as data specification. Uniquely, there is also an international education initiative that aims to train graduates in the development of DS in this area (the Simulation Exploration Experience (SEE)) with published work describing how international teams of students annually build a DS of a moonbase[4] (Elfrey & Severinghaus, 2015; Falcone, Garro, Longo, & Spadafora, 2014; Taylor, et al., 2014). *Transportation/Traffic* applications range from the design of Intelligent Transportation Systems (ITS) in smart cities (Ventresque et al., 2012; Xu, Aydt, & Lees, 2012) to traffic prediction (De Grande, Boukerche, Guan, & Aljeri, 2016; Suh, Hunter, & Fujimoto, 2014). Some work has also investigated the use of cloud to support large-scale DS of transportation networks (Hanai, Suzumura, Ventresque, & Shudo, 2015; Zehe, Knoll, Cai, & Aydt, 2015).

Overall, there is a broad range of DS applications. The main areas are manufacturing and supply chains (24.8%) and defence (16.0%). Most manufacturing and supply chain application papers attempt to implement some idealized supply chain or manufacturing system rather than study a specific real-world problem. Most defence papers reflect some kind of real or proposed training-related defence application. Scientific application papers (e.g. energy, environment, computer architecture, humanities and networks) tend to study some specified scientific problem. Transportation papers are a mix of abstract studies and those contributing to some identified real world problem. Overall there is a range of work in all DS modes that are attempting to speed up large models, to link and reuse existing models and to speed up simulation experimentation. DS application areas tend to be predominantly time managed when simulations are used for investigation and real-time when simulations are used for training. Most application areas have some research addressing the impact of contemporary technologies such as grid or cloud. The HLA

[4] Since 2011, led by NASA and SISO, up to 15 teams annually work together from countries across the world on different federates of a DS of a moonbase. These have included astronauts, rovers, spaceships, satellites, asteroids and asteroid interceptors. The majority were real-time simulations. However, agent-based and discrete-event simulations have also been used. The teams jointly develop a narrative to coordinate the actions of their simulations in the moonbase DS. In the "graduation" event each year the teams work together to run the DS. The moonbase visualization is done by another federate and uses the Unity engine. See www.exploresim.com for further details.

is only used in defence, space and some manufacturing and logistics work (usually associated with standards development). The following can also be observed:

- simulation software that appears in these papers tends to be "home grown" or open source applications or libraries. There are very few examples of DS applications with commercial simulation packages, possibly due to the difficulty in interfacing to these (e.g. accessing the event list/clock);
- work tends to either address the speed up of large models, reusability of large real-time simulations or the speed up of experimentation with single non-DS models;
- some papers have a multi-disciplinary set of authors and there is evidence of the "right" problem being addressed; however there are also examples of "a solution looking for a problem" where there is no evidence of involvement of domain stakeholders;
- there are DS research groups across the world who have advanced the state-of-the-art; however in general research tends to be isolated with many DS applications being "one-offs" that do not really build on the work of these groups;
- there are examples of large-scale DS frameworks being developed (e.g. where DS is used as part of a wider enterprise system) as a result of funded research but there is no real evidence of these systems being used after the end of the funding period;
- a very specific skillset is required and researcher/developers tend to have some kind of computing, computer science or software engineering background; in almost all cases the application was highly complex to develop and implement;
- many applications reported good performance; however, increased performance was usually achieved by extra research into load balancing, improvement of time management, etc.;
- some applications were implemented on general purpose computing systems and some were implemented on highly specific ones (e.g. supercomputers); the latter achieved better performance;
- there is evidence of generic "off-the-shelf" solutions for real-time DS but not for other forms possibly due to a lack of widespread acceptance/knowledge of the HLA outside of defence and space applications;
- there is very little evidence of DS being used in non-defence/space industries (with the exception of speeding up experimentation by distributing individual experiments over distributed computers); there is much evidence of DS being used to support research and some examples of use in transportation/traffic.

These observations follow those made in surveys by Boer et al. (2009) and Strassburger, Schulze, & Fujimoto (2008) and reflect the general complexity of DS development and a lack of general purpose systems, despite good reports on performance. Indeed this has recently been commented on by Fujimoto (2016). An encouraging trend, however, is that there is evidence of multi-disciplinary research teams. There was also evidence of at least one DS international training program. When the above cited reviews were conducted most DS research did not involve stakeholders from the application domain. This has changed a little and there are some good examples of the "right" problem being addressed from the application area. These tend to be led by established DS research groups across the world and have advanced the state-of-the-art usually by attracting substantially funded projects to create large-scale DS application frameworks or dedicated DS systems (e.g. projects which include many industrial partners such as CloudSME, CloudFLOW and Fortissimo[5] as well as the NATO Modelling & Simulation Groups which are composed of industrial and defence stakeholders). These have worked particularly well in defence and on-going research projects in computer systems where there is some sustainable skill base.

[5] www.cloudsme.eu, eu-cloudflow.eu and www.fortissimo-project.eu

However, in other areas such as manufacturing, there is little evidence of these frameworks being used beyond the lifetime of the project, again perhaps due to the issues of implementation complexity and a lack of sustainable supporting industrial skill base. However, there might be a simpler reason. Many simulation projects develop a model to solve a specific problem. When the project is finished, the model is finished. An underlying assumption in DS is that it enables model reuse. In defence many models are created and then used repeatedly to study, for example, different combat scenarios. Models of different military elements (tanks, aircraft, warships, terrain, red/blue/white forces, etc.) might be reused together in these different scenarios. DS is therefore practiced in defence. In computer architecture, control systems and networks large-scale models require large-scale computing power and, arguably, without DS researchers would not be able to study the future architecture of the Internet, advanced computer architectures or emerging cyber-physical systems and the Internet of Things. However, in other areas such as manufacturing systems and healthcare the lack of DS might just be due to a misunderstanding of the volume of available, reusable models. It might also be that commercial software tools do not easily support DS and therefore make the creation of large-scale simulations composed of reusable models extremely difficult. In the author's experience unpublished work does exist. For example, Sellafield PLC (supported by Saker Solutions) and the Ford Motor Company are examples of where DS of large-scale simulations of industrial processes are being built. The DS at Sellafield is also capable of running experiments in parallel using the SAKERGRID system described earlier (Kite, et al., 2011; Kite, 2017). Ford has a high performance simulation system that runs many simulation experiments in parallel and there are plans to run the Ford DS on their system as well. Both examples are supported by a software development team in partnership with the simulation software vendor. Both have models that are reused to address larger-scale problems.

The next section will consider the consequences of the above, the potential for OR and a possible way forward.

6. The Potential Impact of Distributed Simulation on Operational Research

What is the potential impact of DS on OR? The introduction identified several potential benefits of DS. The previous section identified the main themes of DS that have appeared in recent years. These are speeding up large models, linking and reusing existing models and speeding up simulation experimentation. In the short term, the last of these could have the greatest impact on OR. In the longer term, as the demand for decision making based on Big Data Analytics grows, DS may be a necessity to enable simulation techniques to cope with the needs of large-scale data processing and large-scale systems. Each of these points will be discussed in turn. The possibility of a sustainable approach to DS is then discussed.

6.1 High Speed Experimentation

Although not "mainstream" DS, the use of distributed computing to speed up simulation experimentation is possibly the easiest application of DS to conceptualise and implement. The notion is very attractive if it is simple to use. Simulation experimentation is typically time limited; one may only have a certain amount of time to experiment. For example, a model has two input parameters and one output KPI. The input parameters can take 10 values each. This leads to 100 possible experiments. Replications need to be included. Assuming 10 replications per experiment leads to 1000 simulation runs for the overall experiment. If the model takes an hour to simulate then the experimentation will take 1000 hours (just under 42 days). Many models run faster than this; however, even if a model took one minute to run the experimentation would take just under 17 hours. In theory at least, if 10 computers were used then the experimentation time of our two

models would be reduced to 4.2 days and 1.7 hours respectively (in reality it would be more than this as distributed computing introduces a delay where computers are initialized, data/models are sent to the computers and results are returned – however there would still be a substantial time saving). Cloud computing in this context is very attractive in that many more computers could be leased just for the time needed for experimentation. In the above, hiring a computer on the Amazon EC2 cloud costs at the time of writing around $0.023/hour. In the above hiring 10 computers for 100 hours would cost around $20 (this would vary depending on the profile of the computing instance needed for the simulation). Overall, the proposition to do more experimentation at low cost is a very attractive proposition for OR.

In addition to the research described earlier several OR researchers have been investigating this form of DS. Fu (1994) was one of the earliest to note the potential of massively parallel simulation in stochastic optimization. More recently, Nelson (2016) notes that "parallel simulation is becoming easy to do, and any simulation experiment that requires multiple replications or multiple scenarios can benefit dramatically from parallel simulation." He also goes on to comment that both computational efficiency and statistical validity have roles to play in novel parallel optimization strategies. Several recent articles have explored a combination of these strategies and different technological implementations (Luo, Hong, Nelson, & Wu, 2015; Ni, Ciocan, Henderson, & Hunter, 2017; Ni, Hunter, & Henderson, 2013). Some commercial simulation packages are offering limited parallel processing either by using machines attached to a local area network or multiple CPUs of a computer, if available. Overall, the work described in these papers show the feasibility of optimization-based techniques in addition to "simple" experimentation. However, particularly in commercial simulation software, access to large amounts of computing resources is far from simple due to the need of additional management software and the use of different grid/cloud computing approaches. The question remains as to how these can be made commonplace and easily accessible by OR researchers and practitioners.

6.2 Big Data and Simulation

There is a growing trend in business systems to produce big data and the need for associated analytical techniques, or *Big Data Analytics* (Chen, Chiang, & Storey, 2012). McAfee & Brynjolfsson (2012) characterise big data in terms of volume (amount of data), velocity (speed of creation) and variety (multiple sources). Lustig, et al. (2010) propose the view that analytics in OR comprises three distinct aspects: descriptive (what happened), predictive (what will happen next?) and prescriptive (what should the business do next?). Mortenson, Doherty, & Robinson (2015) argue that Big Data and real-time analytics (amongst others) are critical research areas in OR. It is reasonable to assume that the same expectations of Big Data Analytics could be made of simulation. One might expect that if simulation is to be used in the context of emerging large-scale systems such as mass customization, complex supply chains and the Cyber-physical systems of Industry 4.0, Digital Twins and Smart Cities, simulation approaches need to be able to deal with larger amounts of data, larger models and richer analysis through more experimentation.

Khan (in Taylor et al., 2015, p. 649) proposed "Big Simulation" as sets of coupled simulations that take big data input and produce big data output in near to real time. Symbiotic simulation has emerged as a possible approach to integrating simulation with Digital Twins, IoT and Big Data (Aydt, et al., 2009; Yang, Shen, & Wang, 2018). Nelson (2016, p.4) suggested that the success of data analytics in business and industry will "lead simulation users to expect the same sort of fine-grained, conditional analysis from their simulations". DS could be the way forward to enable simulation to deal with linking and reusing models to create simulations of big systems, the processing of large-scale big simulations and the processing of the associated outputs. In the same sense of Big Data Analytics, one might call this *Big Simulation Analytics* where predictive and prescriptive analytical

techniques are applied to “big” problems. As noted in the previous section there are examples of technological infrastructure that have been created for large-scale simulation. However, as also noted, these tend to exist for as long as the associated simulation project exists and there is little evidence (apart from the exceptions noted) that there are sustainable infrastructures being created. Again, how could these techniques be made commonplace?

6.3 Experiences with e-Infrastructures and e-Science

Many large-scale international scientific projects such as the Large Hadron Collider (LHC) and the forthcoming Square Kilometre Array, as well as many “smaller” scale projects in biology and medicine, produce large amounts of data and involve international collaborations of hundreds of institutes. In many of these simulation plays a key role. In the LHC project, simulations enable theories to be tested against observed data and require large computing resources to perform experiments in a reasonable amount of time. It is a major challenge to provide effective access for scientists across the world to this enormous amount of data, simulations and supporting computing facilities. The concept of a worldwide distributed computing infrastructure known as “The Grid” was first coined in the mid-1990s in the same sense as plugging into an electricity power grid; scientists would be able to easily plug into a scientific grid of computers, data, applications and sensors across organizations to support international scientific projects (Foster et al., 2001, p. 200).

Through dedicated funding programs The Grid has evolved into an international system of high performance networks and computers termed *e-Infrastructures or cyberinfrastructures* (the former is a European term and the latter an American term for the same concept) (Bird, Jones, & Kee, 2009). E-Science denotes the pursuit of science enhanced with these advanced distributed infrastructures (Hey & Trefethen, 2005). Large-scale infrastructure providers have established a sustainable funding base over the long term and are supporting a range of scientific and, to some extent, industrial projects (e.g. the European Grid Initiative (EGI) (European distributed computing infrastructure), GEANT (European high performance networking infrastructure) and supporting National Research and Education Networks (NRENs) (e.g. JANET in the UK)) (Barjak, et al. 2013). The infrastructures continue to evolve; the EGI is developing the EGI Federated Cloud (Fernández-del-Castillo, Scardaci, & García, 2015) and the European Commission is leading the European Open Science Cloud[6] both of which aim to give a scalable and flexible e-Infrastructure to the European research community. Programs such as the EC’s i4MS (ICT Innovation for Manufacturing SMEs) are supporting projects that build on e-Infrastructure developments for industry to create cloud-based advanced modelling, simulation and data analytics services for European engineering and manufacturing SMEs (e.g. see the success stories from projects including CloudSME[7], CloudFLOW[8] and Fortissimo[9]).

There are various sophisticated software systems that exist to use e-Infrastructure facilities, typically by giving “single sign-on” secure access to multiple computers across multiple administrative domains and the ability to manage the execution of jobs on those computers (e.g. WS-PGRADE/gUSE (Peter Kacsuk et al., 2012; Kiss et al., 2014) and the FutureGateway that has evolved from the DECIDE framework (Ardizzone et al., 2012)). E-Infrastructure applications can be created from these by first deploying the application service on the e-Infrastructure and registering it in some form for service catalogue (see below) and then accessing the service via a science gateway (a web-based system that allow scientists to use e-Infrastructures with a simple front end that has been developed for their needs) or some kind of programming interface (usually some kind of RESTful interface)

[6] https://ec.europa.eu/research/openscience/index.cfm?pg=open-science-cloud
[7] http://www.cloudsme-apps.com/simulation-applications/
[8] http://eu-cloudflow.eu/project/impact.html
[9] https://www.fortissimo-project.eu/success-stories

integrated into software that is familiar to the user (for a wide range of examples of these see www.sci-gaia.eu/community, www.cloudsme-apps.com and catalog.sciencegateways.org/#/home for examples of science gateways). Software applications or services are being increasingly developed in a standard way so that they can be stored, browsed and reused from a standardized service catalogue (e.g. the EGI service catalogue (https://www.egi.eu/services/) and the INDIGO service catalogue (www.indigo-datacloud.eu)). Applications can be linked together by workflows, sequences of tasks that are translated into jobs executed on specific computing systems supported by the above software infrastructures (Deelman, et al., 2009; Liew et al., 2016). Examples of workflow systems include Pegasus (Deelman et al., 2016), Kepler (Ludäscher et al., 2006), Taverna (Wolstencroft et al., 2013), Swift (Zhao et al., 2007) and WS-PGRADE/gUSE (Peter Kacsuk et al., 2012).

Science gateways, workflows and e-Infrastructures therefore represent sustainable high performance systems that are used for science and, to an emerging extent, industry. What might such a system look like for DS and OR? The next section briefly outlines a possible future.

6.4 A Future of DS in OR?

In a possible future where DS is commonly used in OR, a user might access an e-Infrastructure via a web-based science gateway, configure a workflow to execute a series of tasks and instruct those tasks to be run. As shown in Figure 4, such a workflow might have five steps: Management, Acquisition, Composition, Experimentation and Analysis.

In the *Management* task a user first selects an experimentation service. This could be for direct experimentation (just run experiments based on KPIs and parameters) or some form of ranking & selection algorithm, some form of parallel optimization, etc. The user then identifies what experiments he or she would like to run and specifies KPIs, parameters, confidence intervals, etc. S/he might choose what *infrastructure* to run on. The choice might be an internal computing resource (e.g. a cluster), different external clouds, a dedicated high performance computing facility, etc. Cost/time information might be given for each infrastructure to help the user to decide which to select. For example, the user might enter the budget they have for the experiments or the time that they would like the experimentation to take. The user could enter the number of replications they would like to run (or use an estimation technique based on the confidence internals (Hoad, Robinson, & Davies, 2011)). The number of runs could then be calculated by combining the number of replications with the KPIs and parameters. Multiplying this with an approximation of the runtime of the simulation would give the estimated overall runtime. The management service could then give cost based on the number of computers (actual CPUs of a cluster, virtual CPUs of a cloud) that would be needed to complete the experimentation in the specified time (or alternatively how long the experimentation would take if there was a specific budget). Once the infrastructure and CPU number have been selected, the user then pays if necessary, and then instructs the management task to run the experiments. The system would then manage the runs over the selected infrastructure (looping as necessary if some optimization service has been selected) and report to the user the progress of the experimentation and when it is complete.

Following the Management task is the *Acquisition* task. This is arguably implicit in any simulation process as experimentation cannot start until the data sources required by the model(s) (databases, spreadsheets, etc.) have been updated. Model(s) might also use a range of statistical distributions that need updating from these sources. In the case of Symbiotic Simulation, Cyber-physical systems or a Digital Twin, this might involve direct data collection from the sensors in a physical system. We may assume that the selection of services in this task has been predefined and the task runs these to perform the updates. At the end of this task the model and its associated data have been

successfully acquired and updated and are ready for the *Composition* task. With a single simulation this task would just ready the model and its supporting components for uploading to the infrastructure. A DS would require several models to be composed (i.e. a set of federates being composed into a federation) and a supporting workflow service could be selected to automate this.

The *Experimentation* task would then create "jobs" based on each experiment, submit these to a queue for the infrastructure to process, manage the execution of the jobs (e.g. relaunching any failed jobs) and then collate the results from each job as their results are returned from the infrastructure. A user could instead select an optimization service that would drive experimentation (e.g. hill climbing, a genetic algorithm, etc.) The final step is the *Analysis* task. Users could select from a set of services that analyse the output from experimentation. This could include, for example a service that produces summary statistics or some deeper time series-based analysis. The Analysis service could itself be workflow based and run over distributed computing resources to reduce the time taken to analyse the output. Indeed, it is possible that a user could request several analyses to be performed at the same time and the results from this be brought together in some kind of hierarchical workflow. In these cases the Management task could be extended to give further cost estimates for analysis. Similar extensions could be made to reflect the on-going cost of optimization.

Based on this workflow, Figure 5 shows a possible conceptualization of an e-Science approach for DS in OR that shows the workflow realized on an e-Infrastructure using a science gateway. This is influenced by the workflow system WS-PGRADE/gUSE and is based on recent experiences with the CloudSME project where several commercial cloud-based simulation systems using e-Infrastructure approaches were created.

Consider the following example. An enterprise is capable of manufacturing a range of widgets for a number of consumers. The manager of the enterprise in this supply chain wants to understand how the behaviour of her factory responds to changes in demand and supply over time. She has a discrete-event model of her factory and agent-based models of her suppliers and consumers (perhaps a more reasonable large supply chain model as this does not assume that other discrete-event simulations in the supply chain exist but does assume that the enterprise has detailed information about supplier/consumer behaviour over time). We assume that a management interface similar to a science gateway has been set up and a workflow has been defined in WS-PGRADE/gUSE. The manager might want to (for example) investigate the most reliable set of suppliers based on a 20% increase in consumption across her product range and to identify the most critical areas in her factory in terms of machine utilization and operator utilization (we assume that a mix of machines and operators are used in her factory to produce the widgets).

In the Management task, she sets up the experiments on her management interface (the equivalent of a science gateway) and chooses an analytics service that can correlate and cluster the simulation results. She then investigates the best available infrastructure to run the experiments within a reasonable amount of time (e.g. compares the cost of Amazon Cloud, Microsoft Azure and a High Performance Computing centre available in her region against running over a local desktop grid), makes her selection and begins the experimentation. The workflow then begins automatic execution by executing the Acquisition task. This executes in parallel to load the most recent data and model into the infrastructure. The *Composition* task then composes the DS by bringing together the three models with HLA standard software for time management. The *Experimentation* task would then create "jobs" based on each experiment and dispatch these through the infrastructure to (say) virtual computers running on the Amazon cloud. As results begin to come in, the infrastructure passes these onto the *Analysis* task. This task takes each set of results and, in turn, sends these jobs out for processing on the infrastructure using a clustering and classification service that runs for each job and then collates these together for display on the management interface. The manager

then makes her decisions within hours rather than months. In the case of Symbiotic Simulation or Digital Twins, once set up, this process might run constantly as the system monitors and attempts to improve the performance of the system via simulation.

7. Conclusions.

This paper has presented the state-of-the-art of DS approaches, technologies and applications from the perspective of three Modes of DS. It has discussed the significant potential of DS for OR and has suggested an approach inspired by successful experiences in e-Science. It is hoped that this article will bring the DS and OR communities closer together by presenting the opportunities and benefits of collaboration. Reflecting on observations made of DS applications, future research in this area should consider:

- **Interdisciplinary Research**: Future research collaborations should consist of teams of OR and DS researchers to give a balance of expertise. Further, to overcome the issue of "a solution looking for a problem", collaborations should also involve domain stakeholders (e.g. potential end users) to, at the very least, validate the overall direction of the work or science gateways developed for end users;
- **Integrating commercial simulation software**: Many OR simulation users tend to use commercial simulation software. If this is the case then software vendors should be involved in collaboration as the effective integration of their software into any solution is vital;
- **Simple access for users**: DS has been criticized as being hard to develop, implement and use. Science gateways have been shown to facilitate the use of complex software systems and may well be a way of making DS easy to use by non-computing experts;
- **Align work with the international e-Science and e-Infrastructure communities**: Developments in this area should also involve the wider e-Infrastructure communities (e.g. those represented by the European Grid Initiative) and efforts should be made to create standardized services that can form part of wider service catalogues (essentially directories or repositories of software that users can access and add to their computations). This would facilitate service reuse so making it significantly easier to build on the work of others;
- **Seek framework funding with the right interdisciplinary team**: Despite criticisms made earlier of large-scale frameworks and related funding initiatives, there are many examples in e-Science where communities have used funding programmes to create successful and sustainable e-Infrastructures. Such a funded initiative may well be required to create an e-Science infrastructure for OR;
- **Develop combined OR/DS methodologies and techniques**: Both OR and DS have simulation methodologies and techniques that use their own "language"; work needs to be done to bring the two areas closer together by developing common approaches that "speak" to the widest possible OR/DS communities;
- **Use and develop OR/DS standards**: Standards are vital to ensure the widest possible compatibility between technologies developed by international groups (e.g. the HLA, standards developed by the Simulation Interoperability Standards Organization and those associated with e-Science and e-Infrastructures); and
- **Develop education and innovation through Hackfests**: The establishment of an international Hackfest, or Hackathon, that involves student and professional teams drawn from OR, DS, e-Science, vendors and stakeholder communities might be an exciting way of jumpstarting innovations in this area.

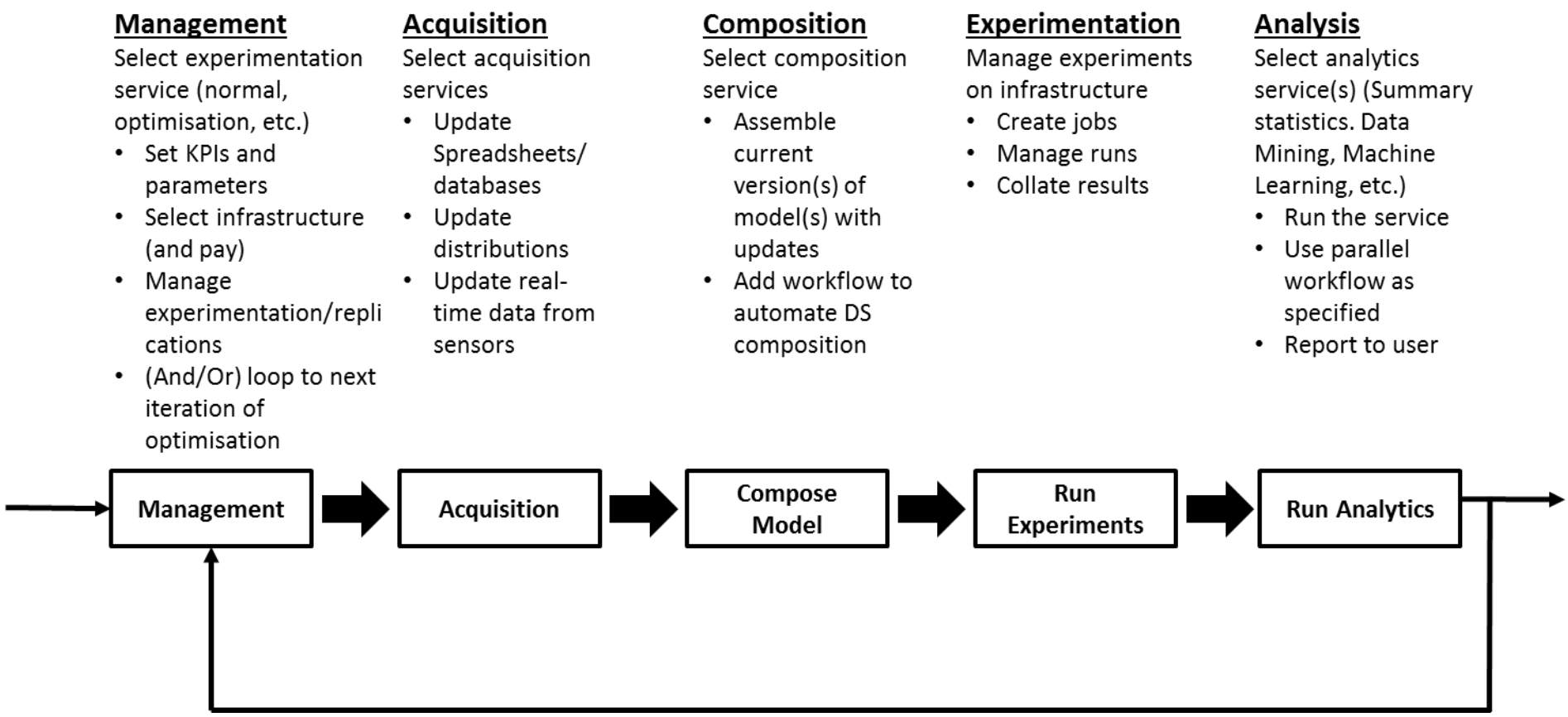

Figure 4: A Workflow for Distributed Simulation in Operational Research

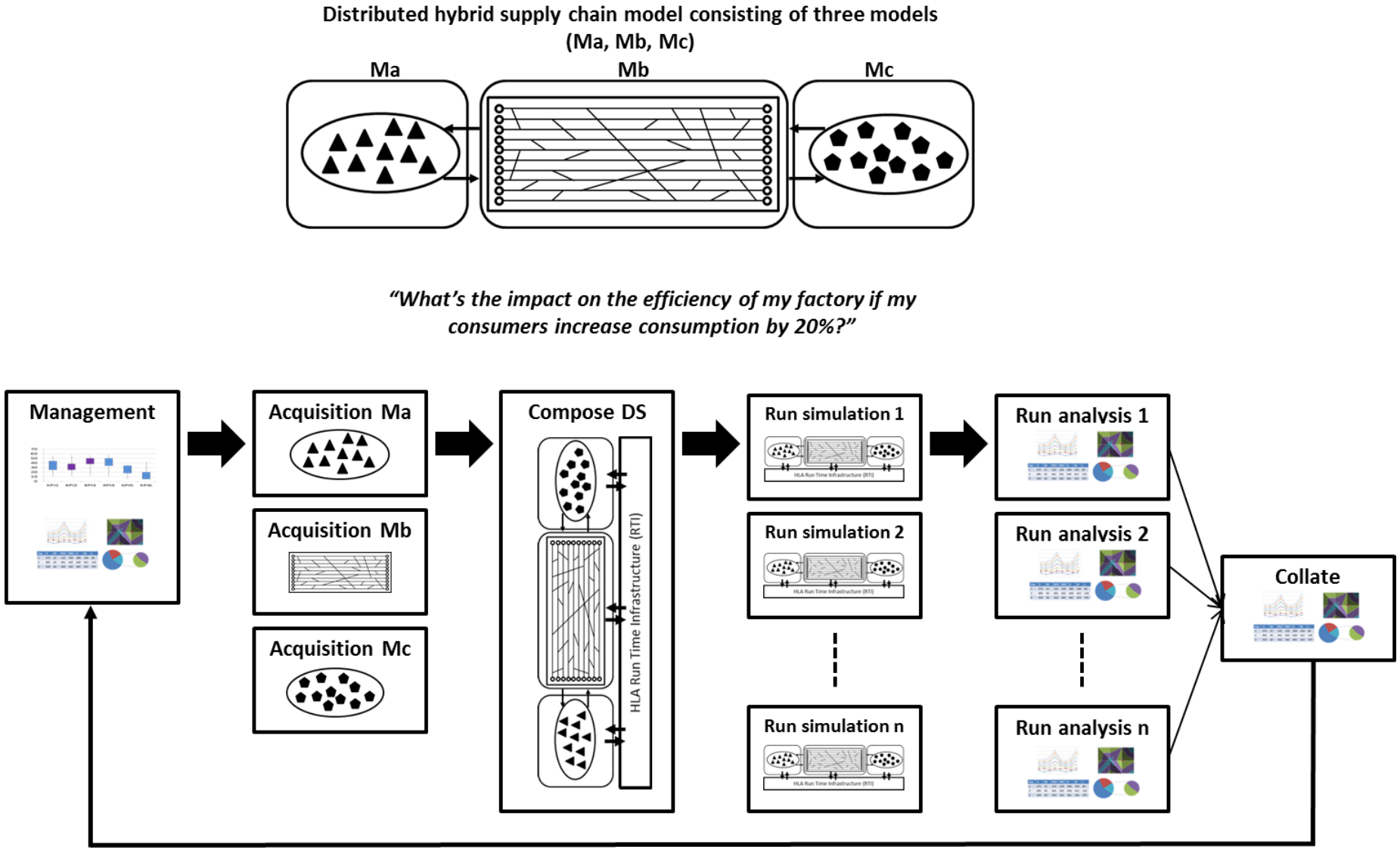

Figure 5: An e-Science Vision for Distributed Simulation in Operational Research

Acknowledgements

The author would like to thank Professor Stewart Robinson for his helpful comments and Dr Anastasia Anagnostou, Dr Tamas Kiss and Professor Roberto Barbera for their discussions on e-Science and OR.

References

Akkaya, A., Genc, G., & Tugcu, T. (2014). HLA based architecture for molecular communication simulation. *Simulation Modelling Practice and Theory*, *42*, 163–177. https://doi.org/10.1016/j.simpat.2013.12.012

Al-Zoubi, K., & Wainer, G. (2013). RISE: A general simulation interoperability middleware container. *Journal of Parallel and Distributed Computing*, *73*(5), 580–594. https://doi.org/10.1016/j.jpdc.2013.01.014

Alghamdi, T. G., De Grande, R. E., & Boukerche, A. (2015). Enhancing load balancing efficiency based on migration delay for large-scale distributed simulations. In *Proceedings of the IEEE/ACM 19th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)* (pp. 33–40). https://doi.org/10.1109/DS-RT.2015.33

Alkharboush, R., De Grande, R. E., & Boukerche, A. (2014). Federate migration decision-making methods for HLA-based distributed simulations. In *Proceedings of the IEEE/ACM 18th International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 190–197). https://doi.org/10.1109/DS-RT.2014.31

Anagnostou, A., Nouman, A., & Taylor, S. J. E. (2013). Distributed hybrid agent-based discrete event emergency medical services simulation. In *Proceedings of the 2013 Winter Simulation Conference (WSC)* (pp. 1625–1636). https://doi.org/10.1109/WSC.2013.6721545

Anagnostou, A., & Taylor, S. J. E. (2017). A distributed simulation methodological framework for OR/MS applications. *Simulation Modelling Practice and Theory*, *70*, 101–119. https://doi.org/10.1016/j.simpat.2016.10.007

Anderson, K., Du, J., Narayan, A., & Gamal, A. E. (2014). GridSpice: A distributed simulation platform for the smart grid. *IEEE Transactions on Industrial Informatics*, *10*(4), 2354–2363. https://doi.org/10.1109/TII.2014.2332115

Ardizzone, V., Barbera, R., Calanducci, A., Fargetta, M., Ingrà, E., Porro, I., … Schenone, A. (2012). The DECIDE Science Gateway. *Journal of Grid Computing*, *10*(4), 689–707. https://doi.org/10.1007/s10723-012-9242-3

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, *54*(15), 2787–2805. https://doi.org/https://doi.org/10.1016/j.comnet.2010.05.010

Awais, M. U., Cvetkovic, M., & Palensky, P. (2017). Hybrid simulation using implicit solver coupling with HLA and FMI. *International Journal of Modeling, Simulation, and Scientific Computing*, *8*(4), December 2017. https://doi.org/10.1142/S1793962317500556

Aydt, H., Turner, S. J., Cai, W., & Low, M. Y. H. (2009). Research issues in symbiotic simulation. In *Proceedings of the 2009 Winter Simulation Conference (WSC)* (pp. 1213–1222). https://doi.org/10.1109/WSC.2009.5429419

Bagrodia, R. L., & Wen-Toh Liao. (1994). Maisie: a language for the design of efficient discrete-event simulations. *IEEE Transactions on Software Engineering*, *20*(4), 225–238. https://doi.org/10.1109/32.277572

Barjak, F., Eccles, K., Meyer, E. T., Robinson, S., & Schroeder, R. (2013). The Emerging Governance of E-Infrastructure. *Journal of Computer-Mediated Communication*, *18*(2), 1–24. https://doi.org/10.1111/jcc4.12000

Bell, D., Mustafee, N., de Cesare, S., Taylor, S. J. E., Lycett, M., & Fishwick, P. A. (2008). Ontology

engineering for simulation component reuse. *International Journal of Enterprise Information Systems*, *4*(4), 1–15.

Bird, I., Jones, B., & Kee, K. F. (2009). The organization and management of grid infrastructures. *Computer*, *42*(1), 36–46. https://doi.org/10.1109/MC.2009.28

Biswas, A., & Fujimoto, R. (2016). Profiling energy consumption in distributed simulations. In *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (PADS)* (pp. 201–209). https://doi.org/10.1145/2901378.2901395

Bocciarelli, P., D'Ambrogio, A., & Fabiani, G. (2012). A model-driven approach to build HLA-based distributed simulations from SysML models. In *Proceedings of the 2012 International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH)* (pp. 49–60).

Bocciarelli, P., D'Ambrogio, A., Giglio, A., & Gianni, D. (2013). A SaaS-based automated framework to build and execute distributed simulations from SysML models. In *Proceedings of the 2013 Winter Simulation Conference (WSC)* (pp. 1371–1382). https://doi.org/10.1109/WSC.2013.6721523

Bocciarelli, P., Pieroni, A., Gianni, D., & D'Ambrogio, A. (2012). A model-driven method for building distributed simulation systems from business process models. In *Proceedings of the 2012 Winter Simulation Conference (WSC)* (pp. 1–12). https://doi.org/10.1109/WSC.2012.6465106

Boer, C. A., de Bruin, A., & Verbraeck, A. (2009). A survey on distributed simulation in industry. *Journal of Simulation*, *3*(1), 3–16. https://doi.org/10.1057/jos.2008.9

Bononi, L., Di Felice, M., D'Angelo, G., Bracuto, M., & Donatiello, L. (2008). MoVES: A framework for parallel and distributed simulation of wireless vehicular ad hoc networks. *Computer Networks*, *52*(1). https://doi.org/10.1016/j.comnet.2007.09.015

Bottura, R., Babazadeh, D., Zhu, K., Borghetti, A., Nordström, L., & Nucci, C. A. (2013). SITL and HLA Co-simulation platforms: Tools for analysis of the integrated ICT and electric power system. In *Proceedings of the 2013 EuroCon Conference* (pp. 918–925). https://doi.org/10.1109/EUROCON.2013.6625092

Brito, A. V., Costa, L. F. S., Bucher, H., Sander, O., Becker, J., Oliveira, H., & Melcher, E. U. K. (2016). A distributed simulation platform using HLA for complex embedded systems design. In *Proceedings of the IEEE/ACM 19th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)* (pp. 195–202). https://doi.org/10.1109/DS-RT.2015.16

Bruzzone, A. G., Massei, M., & Bocca, E. (2012). Simulation models and serious games for project team training in engineering, procurement, construction & commissioning. In *Proceedings of the IASTED African Conference on Modelling and Simulation, AfricaMS 2012* (pp. 761–16). https://doi.org/10.2316/P.2012.761-016

Bryant, R. E. (1977). *Simulation of Packet Communication Architecture Computer Systems*. Massachusetts Institute of Technology. Retrieved from http://www.dtic.mil/dtic/tr/fulltext/u2/a048290.pdf

Carothers, C. D., Bauer, D., & Pearce, S. (2002). ROSS: A high-performance, low-memory, modular time warp system. *Journal of Parallel and Distributed Computing*, *62*(11), 53–60. https://doi.org/10.1016/S0743-7315(02)00004-7

Chandy, K. M., & Misra, J. (1979). Distributed Simulation: A Case Study in Design and Verification of Distributed Programs. *IEEE Transactions on Software Engineering*, *SE-5*(5), 440–452. https://doi.org/10.1109/TSE.1979.230182

Chaudhry, N. R., Nouman, A., Anagnostou, A., & Taylor, S. J. E. (2016). WS-PGRADE workflows for cloud-based distributed simulation. In *Proceedings of the Operational Research Society Simulation Workshop 2016* (pp. 192–201).

Chen, H., Chiang, R. H. L., & Storey, V. C. (2012). Business intelligence and analytics: From big data to big impact. *MIS Quarterly: Management Information Systems*, *36*(4).

Chen, T.-L. (2014). Decision support system based on distributed simulation optimization for medical resource allocation in emergency department. *Lecture Notes in Computer Science*, *LNCS 8527*,

15–24. https://doi.org/10.1007/978-3-319-07293-7_2

Chen, X., Cai, W., Turner, S. J., & Wang, Y. (2006). SOAr-DSGrid: Service-Oriented Architecture for Distributed Simulation on the Grid. In *Proceedings of the 2006 IEEE Workshop on Principles of Advanced and Distributed Simulation (PADS)* (pp. 65–73). https://doi.org/10.1109/PADS.2006.33

Chetlur, M., Abu-Ghazaleh, N., Radhakrishnan, R., & Wilsey, P. A. (1998). Optimizing communication in Time-Warp simulators. In *Proceedings of the 12th IEEE Workshop on Parallel and Distributed Simulation (PADS)* (pp. 64–71).

Chiang, M., & Zhang, T. (2016). Fog and IoT: An Overview of Research Opportunities. *IEEE Internet of Things Journal*, *3*(6), 854–864. https://doi.org/10.1109/JIOT.2016.2584538

Choi, C., Seo, K.-M., & Kim, T. G. (2014). DEXSim: an experimental environment for distributed execution of replicated simulators using a concept of single simulation multiple scenarios. *SIMULATION*, *90*(4), 355–376. https://doi.org/10.1177/0037549713520251

Cocchia, A. (2014). Smart and Digital City: A Systematic Literature Review. In R. P. Dameri & C. Rosenthal-Sabroux (Eds.), *Smart City* (pp. 13–43). Springer International Publishing. https://doi.org/10.1007/978-3-319-06160-3_2

Collier, N., Ozik, J., & Macal, C. M. (2015). Large-scale agent-based modeling with repast HPC: A case study in parallelizing an agent-based model. *Lecture Notes in Computer Science*, *9523*, 454–465. https://doi.org/10.1007/978-3-319-27308-2_37

Cordasco, G., de Chiara, R., Mancuso, A., Mazzeo, D., Scarano, V., & Spagnuolo, C. (2013). Bringing together efficiency and effectiveness in distributed simulations: The experience with D-Mason. *SIMULATION*, *89*(10), 1236–1253. https://doi.org/10.1177/0037549713489594

Coudert, D., Hogie, L., Lancin, A., Papadimitriou, D., Perennes, S., & Tahiri, I. (2012). Feasibility study on distributed simulations of BGP. In *Proceedings of the 2012 ACM SIGSIM Conference on Principles of Advanced and Distributed Simulation (PADS)* (pp. 96–98). https://doi.org/10.1109/PADS.2012.19

Craenen, B., Murgatroyd, P., Theodoropoulos, G., Gaffney, V., & Suryanarayanan, V. (2012). MWGrid: A system for distributed agent-based simulation in the digital humanities. In *Proceedings of the 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 124–131). https://doi.org/10.1109/DS-RT.2012.24

D'Ambrogio, A., & Gianni, D. (2004). Using CORBA to Enhance HLA Interoperability in Distributed and Web-Based Simulation. *Lecture Notes in Computer Science*, *LNCSl 3280*, 696–705. https://doi.org/10.1007/978-3-540-30182-0_70

D'Angelo, G., & Marzolla, M. (2014). New trends in parallel and distributed simulation: From many-cores to Cloud Computing. *Simulation Modelling Practice and Theory*, *49*, 320–335. https://doi.org/10.1016/j.simpat.2014.06.007

Dahmann, J. S., Fujimoto, R. M., & Weatherly, R. M. (1997). The Department of Defense High Level Architecture. In *Proceedings of the 1997 Winter Simulation Conference (WSC)* (pp. 142–149). https://doi.org/10.1145/268437.268465

Damani, O. P., Wang, Y. M., & Garg, V. K. (1997). Optimistic distributed simulation based on transitive dependency tracking. In *Proceedings of the 1997 IEEE Workshop on Parallel and Distributed Simulation (PADS)* (pp. 90–97).

Das, S., Fujimoto, R., Panesar, K., Allison, D., & Hybinette, M. (1994). GTW: a time warp system for shared memory multiprocessors. In *Proceedings of the 1994 Winter Simulation Conference (WSC)* (pp. 1332–1339).

De Grande, R. E., & Boukerche, A. (2011). Dynamic balancing of communication and computation load for HLA-based simulations on large-scale distributed systems. *Journal of Parallel and Distributed Computing*, *71*(1), 40–52. https://doi.org/10.1016/j.jpdc.2010.04.001

De Grande, R. E., Boukerche, A., Guan, S., & Aljeri, N. (2016). A modular distributed simulation-based architecture for intelligent transportation systems. *Concurrency and Computation: Practice and Experience*, *28*(12), 3409–3426. https://doi.org/10.1002/cpe.3801

De Munck, S., Vanmechelen, K., & Broeckhove, J. (2014). Revisiting conservative time synchronization protocols in parallel and distributed simulation. *Concurrency and Computation: Practice and Experience*, *26*(2), 468–490. https://doi.org/10.1002/cpe.3007

Deelman, E., Gannon, D., Shields, M., & Taylor, I. (2009). Workflows and e-Science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, *25*(5), 528–540. https://doi.org/10.1016/j.future.2008.06.012

Deelman, E., Vahi, K., Rynge, M., Juve, G., Mayani, R., & Da Silva, R. F. (2016). Pegasus in the cloud: Science automation through workflow technologies. *IEEE Internet Computing*, *20*(1), 70–76. https://doi.org/10.1109/MIC.2016.15

Dibbern, C., Hahn, A., & Schweigert, S. (2014). Interoperability in co-simulatons of maritime systems. In *Proceedings of the 28th European Conference on Modelling and Simulation, (ECMS)* (p. 96).

Dimakis, N., Filippoupolitis, A., & Gelenbe, E. (2010). Distributed building evacuation simulator for smart emergency management. *Computer Journal*, *53*(9), 1384–1400. https://doi.org/10.1093/comjnl/bxq012

Dragoicea, M., Bucur, L., Tsai, W.-T., & Sarjoughian, H. (2012). Integrating HLA and service-oriented architecture in a simulation framework. In *Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2012)* (pp. 861–866). https://doi.org/10.1109/CCGrid.2012.76

Elfrey, P., & Severinghaus, R. (2015). The Simulation Exploration Experience education opportunity in higher education: Preparing college students to thrive in chaos. In *Proceedings of the 14th International Conference on Modeling and Applied Simulation (MAS)* (pp. 162–171).

Falcone, A., Garro, A., Longo, F., & Spadafora, F. (2014). Simulation exploration experience: A communication system and a 3D real time visualization for a moon base simulated scenario. In *Proceedings of the 18th IEEE International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 113–120). https://doi.org/10.1109/DS-RT.2014.22

Falcone, A., Garro, A., Taylor, S. J. E., Anagnostou, A., Chaudhry, N. R., & Salah, O. (2016). Experiences in simplifying distributed simulation: The HLA development kit framework. *Journal of Simulation*, *10*(37), 1–20. https://doi.org/10.1057/s41273-016-0039-4

Fernández-del-Castillo, E., Scardaci, D., & García, Á. L. (2015). The EGI Federated Cloud e-Infrastructure. *Procedia Computer Science*, *68*, 196–205. https://doi.org/10.1016/j.procs.2015.09.235

Foster, I., Kesselman, C., & Tuecke, S. (2001). The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *The International Journal of High Performance Computing Applications*, *15*(3), 200–222. https://doi.org/10.1177/109434200101500302

Frank, M., Laroque, C., & Uhlig, T. (2013). Reducing computation time in simulation-based optimization of manufacturing systems. In *Proceedings of the 2013 Winter Simulation Conference (WSC)* (pp. 2710–2721). https://doi.org/10.1109/WSC.2013.6721642

Fu, D., Becker, M., & Szczerbicka, H. (2014). Accelerating distributed discrete event simulation through exchange of conditional look-ahead. In *Proceedings of the 17th IEEE International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 183–189). https://doi.org/10.1109/DS-RT.2014.30

Fu, M. C. (1994). Optimization via simulation: A review. *Annals of Operations Research*, *53*(1), 199–247. https://doi.org/10.1007/BF02136830

Fujii, S., Fujii, N., Tsumaya, A., Iwamura, K., Morinaga, E., Inoue, T., & Mariyama, T. (2012). A basic study on a highly distributed simulation of manufacturing systems under the ubiquitous environment. In *Proceedings of the ASME/ISCIE 2012 International Symposium on Flexible Automation (ISFA)* (pp. 321–324). https://doi.org/10.1115/ISFA2012-7208

Fujimoto, R., & Biswas, A. (2016). An empirical study of energy consumption in distributed simulations. In *Proceedings of the IEEE/ACM 19th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)* (pp. 163–170). https://doi.org/10.1109/DS-RT.2015.32

Fujimoto, R. M. (1990). Parallel Discrete Event Simulation. *Commun. ACM*, *33*(10), 30–53. https://doi.org/10.1145/84537.84545

Fujimoto, R. M. (1998). Time Management in The High Level Architecture. *SIMULATION*, *71*(6), 388–400. https://doi.org/10.1177/003754979807100604

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

Fujimoto, R. M. (2016). Research Challenges in Parallel and Distributed Simulation. *ACM Transactions on Modeling and Computer Simulation*, *26*(4), 1–29. https://doi.org/10.1145/2866577

Fujimoto, R. M., Malik, A. W., & Park, A. (2010). Parallel and distributed simulation in the cloud. *SCS M&S Magazine*, *3*, 1–10.

Gafni, A. (1988). Rollback Mechanisms for Optimistic Distributed Simulation Systems. In *Proceedings of the 1988 SCS Multiconference on Distributed Simulation* (pp. 61–67).

Gan, B.-P., Low, Y.-H., Cai, W., Turner, S. J., Jain, S., Hsu, W. J., & Huang, S. Y. (2001). The Development of Conservative Superstep Protocols for Shared Memory Multiprocessor Systems. *Scalable Computing: Practice and Experience*, *4*(1), 46. https://doi.org/10.12694/SCPE.V4I1.222

Garraghan, P., McKee, D., Ouyang, X., Webster, D., & Xu, J. (2016). SEED: A Scalable Approach for Cyber-Physical System Simulation. *IEEE Transactions on Services Computing*, *9*(2), 199–212. https://doi.org/10.1109/TSC.2015.2491287

Gogi, A., Tako, A. A., & Robinson, S. (2016). An experimental investigation into the role of simulation models in generating insights. *European Journal of Operational Research*, *249*(3), 931–944. https://doi.org/10.1016/j.ejor.2015.09.042

Gonsiorowski, E., Carothers, C., & Tropper, C. (2012). Modeling large scale circuits using massively parallel discrete-event simulation. In *Proceedings of the IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)* (pp. 127–133). https://doi.org/10.1109/MASCOTS.2012.24

Guan, S., De Grande, R. E., & Boukerche, A. (2016). Enabling HLA-based simulations on the cloud. In *Proceedings of the 19th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DS-RT)* (pp. 112–119). https://doi.org/10.1109/DS-RT.2015.36

Ha, S., Cha, J.-H., Roh, M.-I., & Lee, K.-Y. (2012). Implementation of the submarine diving simulation in a distributed environment. *International Journal of Naval Architecture and Ocean Engineering*, *4*(3), 211–227. https://doi.org/10.3744/JNAOE.2012.4.3.211

Hanai, M., Suzumura, T., Ventresque, A., & Shudo, K. (2015). An adaptive VM provisioning method for large-scale agent-based traffic simulations on the cloud. In *Proceedings of the International Conference on Cloud Computing Technology and Science (CloudCom)* (Vol. 2015–Febru, pp. 130–137). https://doi.org/10.1109/CloudCom.2014.164

Heidelberger, P. (1986). Statistical Analysis of Parallel Simulation. In *Proceedings of the 1986 Winter Simulation Conference (WSC)* (pp. 2278–2288).

Hennicker, R., & Ludwig, M. (2012). View-based development of a simulation framework for multi-disciplinary environmental modelling. *Lecture Notes in Computer Science*, *LNCS 7539*, 224–250. https://doi.org/10.1007/978-3-642-34059-8_12

Henry, G. K., Fiddes, S. P., Burkett, C. W., Duncan, J., McTaggart, K. A., Stuntz, N., & Toni, D. (2015). International simulation of replenishment at sea using the virtual ship standard. In *Proceedings of the 2015 International Conference on Computer Applications in Shipbuilding (ICCAS)* (Vol. 1, p. 30).

Hey, T., & Trefethen, A. E. (2005). Cyberinfrastructure for e-Science. *Science*, *308*(5723), 817–821. https://doi.org/10.1126/science.1110410

Hibino, H., Fukuda, Y., & Yura, Y. (2015). A synchronization mechanism with shared storage model for distributed manufacturing simulation systems. *International Journal of Automation Technology*, *9*(3), 248–260.

Hoad, K., Robinson, S., & Davies, R. (2011). AutoSimOA: A framework for automated analysis of simulation output. *Journal of Simulation*, *5*(1), 9–24. https://doi.org/10.1057/jos.2010.22

Huang, Y.-L., Alexopoulos, C., Hunter, M., & Fujimoto, R. M. (2012). Ad hoc distributed simulation methodology for open queueing networks. *SIMULATION*, *88*(7), 784–800. https://doi.org/10.1177/0037549711408486

IEEE. (1993). *1278-1993 - IEEE Standard for Information Technology - Protocols for Distributed Interactive Simulations Applications. Entity Information and Interaction*. IEEE Computer Society Press. https://doi.org/10.1109/IEEESTD.1993.115125

IEEE. (2010a). *IEEE 1516-2010 IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules*. IEEE Computer Society Press. https://doi.org/10.1109/IEEESTD.2010.5953411

IEEE. (2010b). *IEEE 1516.1-2010 IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification*. New York, N.Y.: IEEE Computer Society Press. https://doi.org/10.1109/IEEESTD.2010.5954120

IEEE. (2010c). *IEEE 1516.2-2010 IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Object Model Template (OMT) Specification*. New York, N.Y.: IEEE Computer Society Press. https://doi.org/10.1109/IEEESTD.2010.5953408

IEEE. (2010d). *IEEE 1730-2010 - IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP)*. New York, N.Y.: IEEE Computer Society Press. https://doi.org/10.1109/IEEESTD.2011.5706287

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

Kacsuk, P., Farkas, Z., Kozlovszky, M., Hermann, G., Balasko, A., Karoczkai, K., & Marton, I. (2012). WS-PGRADE/gUSE Generic DCI Gateway Framework for a Large Variety of User Communities. *Journal of Grid Computing*, *10*(4), 601–630. https://doi.org/10.1007/s10723-012-9240-5

Katsaliaki, K., Mustafee, N., Taylor, S. J. E., & Brailsford, S. (2009). Comparing conventional and distributed approaches to simulation in a complex supply-chain health system. *Journal of the Operational Research Society*, *60*(1), 43–51. https://doi.org/10.1057/palgrave.jors.2602531

Khan, R., Aydin, A., Khan, M. S., Dasgupta, P., & Ahmed, K. (2015). Simulation-based training for prostate surgery. *BJU International*, *116*(4), 665–74. https://doi.org/10.1111/bju.12721

Kim, B. S., Lee, S. J., Kim, T. G., & Song, H. S. (2014). MapReduce based experimental frame for parallel and distributed simulation using hadoop platform. In *Proceedings of the 28th European Conference on Modelling and Simulation* (pp. 664–669). Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84905718753&partnerID=40&md5=0fbf0544536f18091bf9562780917224

Kiss, T., Kacsuk, P., Takacs, E., Szabo, A., Tihanyi, P., & Taylor, S. J. E. (2014). Commercial use of WS-PGRADE/gUSE. In P. Kacsuk (Ed.), *Science Gateways for Distributed Computing Infrastructures: Development Framework and Exploitation by Scientific User Communities* (pp. 271–286). Spinger Cham. https://doi.org/10.1007/978-3-319-11268-8-19

Kite, S. (2017). *SAKERGRID Technical Paper*. Retrieved from www.sakersolutions.com/sakergrid

Kite, S., Wood, C., Taylor, S. J. E., & Mustafee, N. (2011). SAKERGRID: Simulation experimentation using grid enabled simulation software. In *Proceedings of the 2011 Winter Simulation Conference (WSC)* (pp. 2278–2288). https://doi.org/10.1109/WSC.2011.6147939

Kneebone, R., Arora, S., King, D., Bello, F., Sevdalis, N., Kassab, E., … Nestel, D. (2010). Distributed simulation Accessible immersive training. *Medical Teacher*, *32*(1), 65–70. https://doi.org/10.3109/01421590903419749

Krzyszton, M., & Niewiadomska-Szynkiewicz, E. (2016). Mobile ad hoc network for a heavy gas cloud boundary estimation and tracking. In *Proceedings of the 21st International Conference on Methods and Models in Automation and Robotics (MMAR)* (pp. 1004–1009). https://doi.org/10.1109/MMAR.2016.7575275

Kumar, D. (1986). Simulating Feedforward Systems Using a Network of Processors. In *Proceedings of the 1986 Annual Simulation Symposium* (pp. 127–144).

Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system. *Communications*

of the ACM, *21*(7), 558–565. https://doi.org/10.1145/359545.359563
Law, A. M. (2015). *Simulation modeling and analysis* (5th ed.). McGraw-Hill.
Lee, S., Kang, Y., & Prabhu, V. V. (2013). Continuous variable control approach for home care crew scheduling. In *Proceedings of the 2013 Winter Simulation Conference (WSC)* (pp. 2262–2273). https://doi.org/10.1109/WSC.2013.6721602
Lee, Y. J., Park, G. Y., Song, H. K., & Youn, H. Y. (2012). A load balancing scheme for distributed simulation based on multi-agent system. In *Proceedings of the 2012 International Computer Software and Applications Conference* (pp. 613–618). https://doi.org/10.1109/COMPSACW.2012.111
Lendermann, P., Heinicke, M. U., McGinnis, L. F., McLean, C., Strassburger, S., & Taylor, S. J. E. (2007). Panel: distributed simulation in industry - A real-world necessity or ivory tower fancy? In *Proceedings of the 2007 Winter Simulation Conference (WSC)* (pp. 1053–1062). https://doi.org/10.1109/WSC.2007.4419704
Lendermann, P., Low, M. Y. H., Gan, B. P., Julka, N., Chan, L. P., Lee, L. H., … Buckley, S. (2005). An integrated and adaptive decision-support framework for high-tech manufacturing and service networks. In *Proceedings of the 2005 Winter Simulation Conference (WSC)* (pp. 2052–2062). https://doi.org/10.1109/WSC.2005.1574487
Li, D. C., Li, Q., Cheng, N., & Song, J. Y. (2014). SOA-cloud computing based fast and scalable simulation architecture for advanced flight management system. *Advanced Materials Research*, *1016*, 471–477. https://doi.org/10.4028/www.scientific.net/AMR.1016.471
Li, X., Cai, W., & Turner, S. J. (2016). Supporting efficient execution of continuous space agent-based simulation on GPU. *Concurrency and Computation: Practice and Experience*, *28*(12), 3313–3332. https://doi.org/10.1002/cpe.3808
Liew, C. S., Atkinson, M. P., Galea, M., Ang, T. F., Martin, P., & Hemert, J. I. Van. (2016). Scientific Workflows: Moving Across Paradigms. *ACM Computing Surveys*, *49*(4), 1–39. https://doi.org/10.1145/3012429
Lin, J., Wang, G., Hu, Z., & Long, Q. (2012). A supply chain architecture based on ontology for distributed simulation and modeling. *International Journal of Digital Content Technology and Its Applications*, *6*(5), 225–234. https://doi.org/10.4156/jdcta.vol6.issue5.27
Lin, Y. B., Preiss, B. R., Loucks, W. M., & Lazowska, E. D. (1993). Selecting the checkpoint interval in time warp simulation. *SIGSIM Simulation Digest*, *23*, 3–10.
Liu, E. S. (2013). A parallel approach of interest management in exascale simulation systems. In *Proceedings of the12th International Conference on Modeling and Applied Simulation* (pp. 57–60). Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84898820024&partnerID=40&md5=09bbe67d4d1d282baa56ad58c1ed2718
Liu, Q., & Wainer, G. (2012). Multicore acceleration of Discrete Event System Specification systems. *SIMULATION*, *88*(7), 801–831. https://doi.org/10.1177/0037549711412237
Liu, X., Taylor, S. J. E., Mustafee, N., Wang, J., Gao, Q., & Gilbert, D. (2014). Speeding up systems biology simulations of biochemical pathways using Condor. *Concurrency Computation Practice and Experience*, *26*(17), 2727–2742. https://doi.org/10.1002/cpe.3161
Long, Q. (2014). Distributed supply chain network modelling and simulation: Integration of agent-based distributed simulation and improved SCOR model. *International Journal of Production Research*, *52*(23), 6899–6917. https://doi.org/10.1080/00207543.2014.910623
Long, Q. (2016). A novel research methodology for supply network collaboration management. *Information Sciences*, *331*, 67–85. https://doi.org/10.1016/j.ins.2015.10.035
Lubachevsky, B. D. (1989). Scalability of the bounded lag distributed discrete event simulation. In *Proceedings of the 1989 SCS Multiconference on Distributed Simulation* (Vol. 21, pp. 100–107). SCS.
Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., … Zhao, Y. (2006). Scientific workflow management and the Kepler system. *Concurrency Computation Practice and Experience*, *18*(10), 1039–1065. https://doi.org/10.1002/cpe.994

Luo, J., Hong, L. J., Nelson, B. L., & Wu, Y. (2015). Fully sequential procedures for large-scale ranking-and-selection problems in parallel computing environments. *Operations Research*, *63*(5), 1177–1194. https://doi.org/10.1287/opre.2015.1413

Lustig, I., Dietrich, B., Johnson, C., & Dziekan, C. (2010). The Analytics Journey. *Analytics Magazine*, (November/December), 11–13. Retrieved from http://analytics-magazine.org/the-analytics-journey/

Lynch, E. W., & Riley, G. F. (2009). Hardware Supported Time Synchronization in Multi-core Architectures. In *Proceedings of the 2009 ACM/IEEE Workshop on Principles of Advanced and Distributed Simulation (PADS)* (pp. 88–94). https://doi.org/10.1109/PADS.2009.19

Massei, M., Tremori, A., Poggi, S., & Nicoletti, L. (2013). HLA-based real time distributed simulation of a marine port for training purposes. *International Journal of Simulation and Process Modelling*, *8*(1), 42–51. https://doi.org/10.1504/IJSPM.2013.055206

McAfee, A., & Brynjolfsson, E. (2012). Big data: the management revolution. *Harvard Business Review*, *90*(10), 60–66.

McIntyre, H. M., Smith, E., & Goode, M. (2013). United Kingdom mission training through distributed simulation. *Military Psychology*, *25*(3), 280–293. https://doi.org/10.1037/h0094969

Medina, A. C., Nardin, L. G., Pereira, N. N., Botter, R. C., & Sichman, J. S. (2013). A distributed simulation model of the maritime logistics in an iron ore supply chain management. In *Proceedings of the 2013 International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMUTECH)* (pp. 453–460).

Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing. National Institute of Standards and Technology*. Gaithersburg, MD. Retrieved from http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

Misra, J. (1986). Distributed discrete-event simulation. *ACM Computing Surveys*, *18*(1), 39–65. https://doi.org/10.1145/6462.6485

Möller, B., Antelius, F., & Karlsson, M. (2013). Developing the HLA tutorial part two: Towards federation design patterns. In *Proceedings of the 2013 Fall Simulation Interoperability Workshop* (pp. 202–205). Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84884750549&partnerID=40&md5=b2a84cf6eb4c91a88981dd53e89e24b1

Möller, B., Croom-Johnson, S., Hartog, T., Huiskamp, W., Verkoelen, C., Jones, G., & Bennett, M. (2012). Security in NATO collective mission training - Problem analysis and solutions. In *Proceedings of the Spring Simulation Interoperability Workshop 2012* (p. 12S-SIW–032).

Moreira, E. M., Santana, R. H. C., & Santana, M. J. (2005). Using consistent global checkpoints to synchronize processes in distributed simulation. In *Proceedings of the 2005 IEEE International Symposium on Distributed Simulation and Real-Time Applications* (pp. 43–50).

Morse, K. L., Bic, L., & Dillencourt, M. (2000). Interest management in large-scale virtual environments. *Presence: Teleoperators and Virtual Environments*, *9*(1), 52–68. Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-0039623198&partnerID=40&md5=e58e3324c444619fd20c1928101ecab9

Mortenson, M. J., Doherty, N. F., & Robinson, S. (2015). Operational research from Taylorism to Terabytes: A research agenda for the analytics age. *European Journal of Operational Research*, *241*(3), 583–595. https://doi.org/10.1016/j.ejor.2014.08.029

Mubarak, M., Carothers, C. D., Ross, R., & Carns, P. (2012). Modeling a million-node dragonfly network using massively parallel discrete-event simulation. In *Proceedings of the 2012 SC Companion: High Performance Computing, Networking Storage and Analysis* (pp. 366–376). https://doi.org/10.1109/SC.Companion.2012.56

Mustafee, N., Sahnoun, M., Smart, A., & Godsiff, P. (2015). An application of distributed simulation for hybrid modeling of offshore wind farms. In *Proceedings of the 2015 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (SIGSIM-PADS)* (pp. 171–172). https://doi.org/10.1145/2769458.2769492

Mustafee, N., & Taylor, S. J. E. (2009). Speeding up simulation applications using WinGrid.

Concurrency Computation Practice and Experience, *21*(11), 1504–1523. https://doi.org/10.1002/cpe.1401

Mustafee, N., Taylor, S., Katsaliaki, K., Dwivedi, Y., & Williams, M. (2012). Motivations and barriers in using distributed supply chain simulation. *International Transactions in Operational Research*, *19*(5), 733–751. https://doi.org/10.1111/j.1475-3995.2011.00838.x

Nelson, B. L. (2016). "Some tactical problems in digital simulation" for the next 10 years. *Journal of Simulation*, *10*(1), 2–11. https://doi.org/10.1057/jos.2015.22

Ni, E. C., Ciocan, D. F., Henderson, S. G., & Hunter, S. R. (2017). Efficient ranking and selection in parallel computing environments. *Operations Research*, *65*(3), 821–836. https://doi.org/10.1287/opre.2016.1577

Ni, E. C., Hunter, S. R., & Henderson, S. G. (2013). Ranking and selection in a high performance computing environment. In *Proceedings of the 2013 Winter Simulation Conference (WSC)* (pp. 833–845). https://doi.org/10.1109/WSC.2013.6721475

Nicol, D. M., & Reynolds, P. F. (1984). Problem Oriented Protocol Design. In *Proceedings of the 1984 Winter Simulation Conference (WSC)* (pp. 471–474).

Niewiadomska-Szynkiewicz, E., & Sikora, A. (2012). A software tool for federated simulation of wireless sensor networks and mobile ad hoc networks. *Lecture Notes in Computer Science*, *LNCS 7133*, 303–313. https://doi.org/10.1007/978-3-642-28151-8_30

Nouman, A., Anagnostou, A., & Taylor, S. J. E. (2013). Developing a distributed agent-based and DES simulation using poRTIco and repast. In *Proceedings of the 17th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 97–104). https://doi.org/10.1109/DS-RT.2013.18

Peacock, J. . K., Manning, E., & Wong, J. W. (1980). Synchronization of distributed simulation using broadcast algorithms, *4*(1), 3–10. https://doi.org/10.1016/0376-5075(80)90024-0

Perkonigg, F., Brujic, D., & Ristic, M. (2013). MAC-Sim: A multi-agent and communication network simulation platform for smart grid applications based on established technologies. In *Proceedings of the 2013 IEEE International Conference on Smart Grid Communications (SmartGridComm)* (pp. 570–575). https://doi.org/10.1109/SmartGridComm.2013.6688019

Peschlow, P., Honecker, T., & Martini, P. (2007). A flexible dynamic partitioning algorithm for optimistic distributed simulation. In *Proceedings of the 2007 IEEE Workshop on Principles of Advanced and Distributed Simulation* (pp. 219–228). https://doi.org/10.1109/PADS.2007.6

Pfeifer, D., Gerstlauer, A., & Valvano, J. (2013). Dynamic resolution in distributed cyber-physical system simulation. In *Proceedings of the 2013 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (PADS)* (pp. 277–284). https://doi.org/10.1145/2486092.2486127

Pipattanasomporn, M., Feroze, H., & Rahman, S. (2009). Multi-agent systems in a distributed smart grid: Design and implementation. In *Proceedings of the 2009 IEEE/PES Power Systems Conference and Exposition (PSCE)* (pp. 1–8). https://doi.org/10.1109/PSCE.2009.4840087

Prasad, M. V. K. S., Singh, A., Gangadhar, M., & Sobhan Kumar, L. (2014). Real Time Simulation system for aerospace interceptors. In *Proceedings of the International Conference on Computing and Communication Technologies (ICCCT)* (pp. 1–5). https://doi.org/10.1109/ICCCT2.2014.7066714

Proudlove, N. C., Bisogno, S., Onggo, B. S. S., Calabrese, A., & Levialdi Ghiron, N. (2017). Towards fully-facilitated discrete event simulation modelling: Addressing the model coding stage. *European Journal of Operational Research*, *263*(2), 583–595. https://doi.org/10.1016/j.ejor.2017.06.002

Pullen, J. M., Brunton, R., Brutzman, D., Drake, D., Hieb, M., Morse, K. L., & Tolk, A. (2004). Using Web Services to Integrate Heterogeneous Simulations in a Grid Environment. *Lecture Notes in Computer Science*, *LNCS 3038*, 835–847. https://doi.org/10.1007/978-3-540-24688-6_108

Rabelo, L., Sala-Diakanda, S., Pastrana, J., Marin, M., Bhide, S., Joledo, O., & Bardina, J. (2013). Simulation modeling of space missions using the high level architecture. *Modelling and*

Simulation in Engineering, *2013*, Article No. 11. https://doi.org/10.1155/2013/967483
Raczy, C., Tan, G., & Yu, J. (2005). A sort-based DDM matching algorithm for HLA. *ACM Transactions on Modeling and Computer Simulation*, *15*(1), 14–38. https://doi.org/10.1145/1044322.1044324
Rak, M., Cuomo, A., & Villano, U. (2012). mJADES: Concurrent simulation in the cloud. In *Proceedings of the 2012 International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)* (pp. 853–860). https://doi.org/10.1109/CISIS.2012.134
Reynolds, P. F. (1972). A Shared Resource Algorithm for Distributed Simulation. *ACM SIGARCH Computer Architecture News*, *10*(3), 259–266. Retrieved from http://dl.acm.org/citation.cfm?id=801734
Righter, R., & Walrand, J. C. (1989). Distributed simulation of discrete event systems. *Proceedings of the IEEE*, *77*(1), 99–113. https://doi.org/10.1109/5.21073
Robinson, S. (2002). Modes of simulation practice: approaches to business and military simulation. *Simulation Modelling Practice and Theory*, *10*(8), 513–523. https://doi.org/https://doi.org/10.1016/S1569-190X(02)00117-X
Robinson, S. (2014). *Simulation: The Practice of Model Development and Use* (2nd ed.). Palgrave Macmillan UK. Retrieved from https://he.palgrave.com/page/detail/Simulation/?K=9781137328021
Robinson, S., Nance, R. E., Paul, R. J., Pidd, M., & Taylor, S. J. E. (2004). Simulation model reuse: Definitions, benefits and obstacles. *Simulation Modelling Practice and Theory*, *12*(7–8), 479–494. https://doi.org/10.1016/j.simpat.2003.11.006
Robinson, S., Radnor, Z. J., Burgess, N., & Worthington, C. (2012). SimLean: Utilising simulation in the implementation of lean in healthcare. *European Journal of Operational Research*, *219*(1), 188–197. https://doi.org/10.1016/j.ejor.2011.12.029
Rönngren, R., & Ayani, R. (1994). Adaptive Checkpointing in Time Warp. In *Proceedings of the 1994 IEEE Workshop on Parallel and Distributed Simulation (PADS)* (pp. 110–117).
Rossetti, M. D., & Chen, Y. (2012). A cloud computing architecture for supply chain network simulation. In *Proceedings of the 2012 Winter Simulation Conference (WSC)* (pp. 1–12). https://doi.org/10.1109/WSC.2012.6465196
Santos, A., Leal, K., & Chiroque, L. F. (2013). Building an HLA-based distributed simulation: A metadata approach. In *Proceedings of the 2013 IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 153–160). https://doi.org/10.1109/DS-RT.2013.24
Sarli, J. L., Leone, H. P., & Gutiérrez, M. D. L. M. (2016). Ontology-based semantic model of supply chains for modeling and simulation in distributed environment. In *Proceedings of the 2016 Winter Simulation Conference (WSC)* (pp. 1182–1193). https://doi.org/10.1109/WSC.2016.7822175
Siegfried, R., Van Den Berg, T., Cramp, A., & Huiskamp, W. (2014). M&S as a service: Expectations and challenges. In *Proceedings of the 2014 Fall Simulation Interoperability Workshop* (pp. 248–257). Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84910128682&partnerID=40&md5=473cd1b1c9e5bf36a18da1ce3e2815ce
Silvente, J., Zamarripa, M., & Espuña, A. (2012). Use of a distributed simulation environment for training in Supply Chain decision making. *Computer Aided Chemical Engineering*, *30*, 1402–1406. https://doi.org/10.1016/B978-0-444-59520-1.50139-1
SISO. (2010a). *SISO-STD-006-2010 Standard for COTS Simulation Package Interoperability Reference Models*. (S. J. E. Taylor, Ed.). Orlando, FL.: Simulation Interoperability Standards Organization.
SISO. (2010b). *SISO-STD-008-2010 Standard for Core Manufacturing Simulation Data*. Orlando, FL.: Simulation Interoperability Standards Organization.
Steinman, J. S. (1992). SPEEDES - A multiple-synchronization environment for parallel discrete-event simulation. *International Journal in Computer Simulation*, 251–286. Retrieved from https://ntrs.nasa.gov/search.jsp?R=19930070828

Steinman, J. S. (2005). The WarpIV Simulation Kernel. In *Proceedings of the 2005 IEEE Workshop on Principles of Advanced and Distributed Simulation* (pp. 161–170). https://doi.org/10.1109/PADS.2005.32

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 (WSC)* (pp. 777–785). https://doi.org/10.1109/WSC.2008.4736140

Strassburger, S., & Taylor, S. J. E. (2012). A comparison of the CSPI and CMSD standards. In *Proceedings of the 2012 Spring Simulation Interoperability Workshop* (pp. 82–89).

Suh, W., Hunter, M. P., & Fujimoto, R. (2014). Ad hoc distributed simulation for transportation system monitoring and near-term prediction. *Simulation Modelling Practice and Theory*, *41*, 1–14. https://doi.org/10.1016/j.simpat.2013.11.002

Suryanarayanan, V., & Theodoropoulos, G. (2013). Synchronised range queries in distributed simulations of multiagent systems. *ACM Transactions on Modeling and Computer Simulation*, *23*(4), 1–25. https://doi.org/10.1145/2517449

Suryanarayanan, V., Theodoropoulos, G., & Lees, M. (2013). PDES-MAS: Distributed simulation of multi-agent systems. In *Procedia Computer Science* (Vol. 18, pp. 671–681). https://doi.org/10.1016/j.procs.2013.05.231

Tako, A. A., & Kotiadis, K. (2015). PartiSim: A multi-methodology framework to support facilitated simulation modelling in healthcare. *European Journal of Operational Research*, *244*(2), 555–564. https://doi.org/10.1016/j.ejor.2015.01.046

Tanenbaum, A. S., & van Steen, M. (2007). *Distributed Systems: Principles and Paradigms* (2nd ed.). Upper Saddle River, N.J.: Prentice Hall. Retrieved from http://dl.acm.org/citation.cfm?id=1202502

Tang, W., & Yao, Y. (2013). A GPU-based discrete event simulation kernel. *SIMULATION*, *89*(11), 1335–1354. https://doi.org/10.1177/0037549713508839

Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. (2018). Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology*, *94*(9), 3563–3576. https://doi.org/10.1007/s00170-017-0233-1

Taylor, S. J. E., Anagnostou, A., Kiss, T., Terstyanszky, G., Kacsuk, P., & Fantini, N. (2015). A tutorial on Cloud computing for Agent-based Modeling & Simulation with Repast. In *Proceedings of the 2015 Winter Simulation Conference* (pp. 192–206). https://doi.org/10.1109/WSC.2014.7019888

Taylor, S. J. E., Khan, A., Morse, K. L., Tolk, A., Yilmaz, L., Zander, J., & Mosterman, P. J. (2015). Grand challenges for modeling and simulation: Simulation everywhere - From cyberinfrastructure to clouds to citizens. *SIMULATION*, *91*(7), 648–665. https://doi.org/10.1177/0037549715590594

Taylor, S. J. E., Kiss, T., Terstyanszky, G., Kacsuk, P., & Fantini, N. (2014). Cloud computing for simulation in manufacturing and engineering: Introducing the CloudSME simulation platform. *Simulation Series*, *46*(2), 89–96.

Taylor, S. J. E., Revagar, N., Chambers, J., Yero, M., Anagnostou, A., Nouman, A., … Elfrey, P. R. (2014). Simulation exploration experience: A distributed hybrid simulation of a lunar mining operation. In *Proceedings of the 18th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 107–122). https://doi.org/10.1109/DS-RT.2014.21

Taylor, S. J. E., Turner, S. J., Strassburger, S., & Mustafee, N. (2012). Bridging the gap: A standards-based approach to OR/MS distributed simulation. *ACM Transactions on Modeling and Computer Simulation*, *22*(4), 1–23. https://doi.org/10.1145/2379810.2379811

Theodoropoulos, G., Yi Zhang, Chen, D., Minson, R., Turner, S. J., Wentong Cai, … Logan. (2006). Large Scale Distributed Simulation on the Grid. In *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid* (pp. 63–63). https://doi.org/10.1109/CCGRID.2006.1630953

Topçu, O., & Oğuztüzün, H. (2013). Layered simulation architecture: A practical approach. *Simulation Modelling Practice and Theory*, *32*, 1–14. https://doi.org/10.1016/j.simpat.2012.11.001

Tozzi, D., & Zini, A. (2011). Naval operations' assessment through HLA based simulations. In *Proceedings of the 2011 International Conference on Computer Applications in Shipbuilding (ICCAS)* (Vol. 3). Retrieved from https://www.rina.org.uk/ICCAS2011_Papers.html

Trappey, A. J. C., Trappey, C. V, Hareesh Govindarajan, U., Chuang, A. C., & Sun, J. J. (2017). A review of essential standards and patent landscapes for the Internet of Things: A key enabler for Industry 4.0. *Advanced Engineering Informatics*, *33*, 208–229. https://doi.org/https://doi.org/10.1016/j.aei.2016.11.007

Tsirogiannis, E., & Theodoropoulos, G. (2013). Profiling multilevel partitioning for asynchronous VLSI distributed simulation. *Communications in Computer and Information Science*, *402*, 310–324. https://doi.org/10.1007/978-3-642-45037-2_29

Tychalas, D., & Karatza, H. (2017). High performance system based on Cloud and beyond: Jungle Computing. *Journal of Computational Science*, *22*, 131–147. https://doi.org/https://doi.org/10.1016/j.jocs.2017.03.027

Van Tendeloo, Y., & Vangheluwe, H. (2015). PythonPDEVS: A distributed parallel DEVS simulator. In *Simulation Series* (Vol. 47, pp. 91–98). Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-84928138677&partnerID=40&md5=16a7b18ef8b1286ebffff12f47e8978c

Vanmechelen, K., De Munck, S., & Broeckhove, J. (2012). Conservative Distributed Discrete Event Simulation on Amazon EC2. In *Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)* (pp. 853–860). https://doi.org/10.1109/CCGrid.2012.73

Ventresque, A., Bragard, Q., Liu, E. S., Nowak, D., Murphy, L., Theodoropoulos, G., & Liu, Q. (2012). SParTSim: A space partitioning guided by road network for distributed traffic simulations. In *Proceedings of the 16th IEEE International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)* (pp. 202–209). https://doi.org/10.1109/DS-RT.2012.37

Venu, M., & Joe, I. (2014). Improving performance of optimistic simulation for distributed simulation system using speculative computation. In *Proceedings of the 2014 International Conference on ICT Convergence* (pp. 428–432). https://doi.org/10.1109/ICTC.2014.6983173

Wang, Z.-N., Gao, Q., Wei, X., & Yin, S.-H. (2012). *Design and implementation of a simulation system for armored communication countermeasure training*. *Applied Mechanics and Materials* (Vol. 128–129). https://doi.org/10.4028/www.scientific.net/AMM.128-129.1363

Wilson, A. L., & Weatherly, R. M. (1994). The Aggregate Level Simulation Protocol: An Evolving System. In *Proceedings of the 1994 Winter Simulation Conference (WSC)* (pp. 781–787). Retrieved from http://dl.acm.org/citation.cfm?id=194367

Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owen, S., … Goble, C. (2013). The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. *Nucleic Acids Research*, *41*, W557-61. https://doi.org/10.1093/nar/gkt328

Wonnacott, P., & Bruce, D. (1996). The APOSTLE simulation language: granularity control and performance data. In *Proceedings of the 1996 IEEE Workshop on Parallel and Distributed Simulation* (pp. 114–123). IEEE. https://doi.org/10.1109/PADS.1996.761569

Xie, Y., Teo, Y. M., Cai, W., & Turner, S. J. (2005). Service provisioning for HLA-based distributed simulation on the Grid. In *Proceedings of the 2005 IEEE Workshop on Principles of Advanced and Distributed Simulation* (pp. 282–291).

Xu, Y., Aydt, H., & Lees, M. (2012). SEMSim: A distributed architecture for multi-scale traffic simulation. In *Proceedings of the 2012 ACM SIGSIM Workshop on Principles of Advanced and Distributed Simulation (PADS)* (pp. 178–180). https://doi.org/10.1109/PADS.2012.40

Yahiaoui, A., & Sahraoui, A.-E.-K. (2012). A framework for distributed control and buildings performance simulation. In *Proceedings of the 2012 Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE* (pp. 232–237). https://doi.org/10.1109/WETICE.2012.44

Yang, C., Huang, Q., Li, Z., Liu, K., & Hu, F. (2017). Big Data and cloud computing: innovation

opportunities and challenges. *International Journal of Digital Earth*, *10*(1), 13–53. https://doi.org/10.1080/17538947.2016.1239771

Yang, C., Shen, W., & Wang, X. (2018). The Internet of Things in Manufacturing: Key Issues and Potential Applications. *IEEE Systems, Man, and Cybernetics Magazine*, *4*(1), 6–15. https://doi.org/10.1109/MSMC.2017.2702391

Yao, Y., Meng, D., Zhu, F., Yan, L., Qu, Q., Lin, Z., & Ma, H. (2017). Three-level-parallelization support framework for large-scale analytic simulation. *Journal of Simulation*, *11*(3), 194–207. https://doi.org/10.1057/s41273-017-0057-x

Zehe, D., Knoll, A., Cai, W., & Aydt, H. (2015). SEMSim Cloud Service: Large-scale urban systems simulation in the cloud. *Simulation Modelling Practice and Theory*, *58*, 157–171. https://doi.org/10.1016/j.simpat.2015.05.005

Zhang, D. Z., & Anosike, A. I. (2012). Modelling and simulation of dynamically integrated manufacturing systems. *Journal of Intelligent Manufacturing*, *23*(6), 2367–2382. https://doi.org/10.1007/s10845-010-0494-0

Zhang, Y., & Zhang, J. (2013). The study of design for torpedo homing-performance simulation system based on high level architecture. In *Proceedings of the 2013 IEEE International Conference on Software Engineering and Service Sciences, ICSESS* (pp. 1045–1048). https://doi.org/10.1109/ICSESS.2013.6615486

Zhao, Y., Hategan, M., Clifford, B., Foster, I., Von Laszewski, G., Nefedova, V., … Wilde, M. (2007). Swift: Fast, reliable, loosely coupled parallel computation. In *Proceedings of the 2007 IEEE Congress on Services* (pp. 199–206). https://doi.org/10.1109/SERVICES.2007.63

Zia, K., Farrahi, K., Riener, A., & Ferscha, A. (2013). An agent-based parallel geo-simulation of urban mobility during city-scale evacuation. *SIMULATION*, *89*(10), 1184–1214. https://doi.org/10.1177/0037549713485468

Glossary

Agent-based Simulation	ABS
Aggregate Level Simulation Protocol	ALSP
Application Programming Interfaces	APIs
Central Processing Unit	CPU
Common Object Request Broker Architecture	CORBA
Core Manufacturing Simulation Data	CMSD
Defense Advanced Research Projects Agency	DARPA
Defense Modeling and Simulation Office	DMSO
Discrete-event Simulation	DES
Distributed Interactive Simulation	DIS
Distributed Simulation	DS
Federate Object Model	
First In First Out	FIFO
Functional Mock-up Interface	FMI
Georgia Tech Time Warp	GTW
Global Virtual Time	GVT
Graphical Processing Unit	GPU
High Level Architecture	HLA
High Performance Computing	HPC
Information and Communication Technologies	ICT
Infrastructure as a Service	IaaS
Innovation for Manufacturing SMEs	i4MS
Institute of Electrical and Electronics Engineers	IEEE
Internet of Things	IoT

Interoperability Reference Models	IRMs
Logical Processes	LPs
Modelling & Simulation	M&S
Modelling & Simulation as a Service	MSaaS
National Research and Education Network	NREN
Object Model Template	OMT
Operational Research	OR
OR/MS	Operational Research/Management Science
Parallel and Distributed Simulation	PADS
Parallel Discrete Event Simulation	PDES
Platform as a Service	PaaS
Rensselaer's Optimistic Simulation System	ROSS
Run Time Infrastructure	RTI
Service Oriented Architecture	SOA
Simulation Interoperability Standards Organization	SISO
SIMulator NETworking	(SIMNET
Single Server Queue	SSQ
Small-to-Medium Enterprise	SME
Software as a Service	SaaS
Standard for COTS Simulation Package Interoperability Reference Models	SISO-STD-006-2010
Standards Activity Committee	SAC
Synchronous Parallel Environment for Emulation and Discrete Event Simulation	SPEEDS
Systems Modelling Language	SysML
The Standard for Core Manufacturing Simulation Data	SISO-STD-008-2010
Timestamped Event Message	TEM
Unified Modelling Language	UML
Very Large Scale Integration	VLSI
World Wide Web	WWW