SPEEDING-UP THE EXECUTION OF CREDIT RISK SIMULATIONS USING DESKTOP GRID COMPUTING: A CASE STUDY

Dr. Navonil Mustafee
School of Business and Economics
Swansea University, Singleton Park
Swansea, Wales, UK
n.mustafee@swansea.ac.uk

Dr. Simon J E Taylor
School of Information Systems and Computing
Brunel University
Uxbridge, UK
simon.taylor@brunel.ac.uk

ABSTRACT:
This paper describes a case study that was undertaken at a leading European Investment bank in which desktop grid computing was used to speed-up the execution of Monte Carlo credit risk simulations. The credit risk simulations were modelled using commercial-off-the-shelf simulation packages (CSPs). The CSPs did not incorporate built-in support for desktop grids, and therefore the authors implemented a middleware for desktop grid computing, called WinGrid, and interfaced it with the CSP. The performance results show that WinGrid can speed-up the execution of CSP-based Monte Carlo simulations. However, since WinGrid was installed on non-dedicated PCs, the speed-up achieved varied according to users' PC usage. Finally, the paper presents some lessons learnt from this case study. It is expected that this paper will encourage simulation practitioners and CSP vendors to experiment with desktop grid computing technologies with the objective of speeding-up simulation experimentation.

Keywords: Grid Computing, Desktop Grids, Monte Carlo Simulation, COTS Simulation Package, WinGrid

1. INTRODUCTION

The grid computing (or grids) vision of providing users continuous access to computing resources, similar to public utility services such as electricity and telephone, can be traced back to the Multiplexed Information and Computing Service (Multics) system that arguably discussed this in the context of time-sharing of a CPU among jobs of several users (Corbato and Vyssotsky, 1965). Grid computing has the potential to provide users on-demand access to large amounts of computing power, just as power grids provide users with consistent, pervasive, dependable and transparent access to electricity, irrespective of its source (Baker et al, 2002). It has been identified that simulation modelling can potentially benefit from this since computing power can be an issue in the time taken to get results from a simulation (Robinson, 2005; Taylor and Robinson, 2006).

In the industry, simulations are often modeled and executed using *Commercial-Off-The-Shelf (COTS) Simulation Packages (CSPs)*. For simulation practitioners to benefit from grid computing, it is important that the CSP vendors incorporate grid support into their products. However, it is generally the case that the vendors incorporate additional functionality into their software on an incremental basis, and only if they are assured of a guaranteed *Return On Investment* since the cost of development can be prohibitively expensive. Another alternative for simulation practitioners to realise the power of the grids is to implement "distributed computing code" that makes it possible to execute CSP-based simulation over multiple PCs. However, the users of these packages tend to be skilled in simulation and not computer science and therefore it is not practical to expect such programming expertise from the vast majority of simulation users. Thus, in order to increase the adoption of grid technologies in the field of CSP-based simulation, it is important to develop grid computing software (subsequently referred to as either *grid computing middleware* or just *middleware*) that grid-enables existing CSPs using a solution that requires little or no change to them. Our system *WinGrid* (Mustafee et al, 2006; Mustafee and Taylor, 2009) aims to deliver such a low intervention technological solution to grid-enable existing Windows-based CSP applications.

By means of a case study conducted in conjunction with a leading European investment bank, we investigate how a grid computing middleware implemented via our system (*WinGrid*) can increase the performance of Monte

Carlo simulation experimentation. Our approach differs from previous attempts to use distributed computing to speed up simulation experimentation (Anagnostopoulos and Nikolaidou, 2003; Biles and Kleijnen, 2003; Paris and Pierreval, 2001; Yücesan et al, 2001) by using a grid composed of commodity PCs at workplace specifically aimed at Windows applications and by transparently, in as much as possible, grid-enabling simulation within an enterprise context (i.e. by changing the existing simulation application as little as possible to encourage adoption of this technology).

The paper is structured as follows. Section 2 presents an overview of Monte Carlo Simulation and the CSPs. This is followed by a discussion on grid computing in Section 3. In Section 4 we describe the architecture of *WinGrid*. Section 5 outlines the case study that was conducted in a leading investment bank. The experiments, corresponding results and related discussions are presented in Section 6. Section 7 outlines some of the lessons learnt and draws the paper to a close.

2. MONTE CARLO SIMULATION AND CSPs

Monte Carlo simulation is a statistical technique which uses a sequence of random numbers to generate values from a known probability distribution associated with a source of uncertainty (Rubinstein, 1981). It is formed by a class of computational algorithms that rely on repeated random sampling to compute a result. This method is usually employed when it is impossible or infeasible to compute an exact result using fixed values or deterministic algorithms.

Monte Carlo simulation is extensively used in application areas such as finance and insurance (Herzog and Lord, 2002). Commercially available spreadsheet applications (Lotus 1-2-3™, etc.), spreadsheet add-ins (Crystal Ball™, @Risk™, etc.) and Monte Carlo simulation packages (Analytica™, etc.) are often used for modelling Monte Carlo simulations in industry (Swain, 2007). In this paper we collectively refer to these different groups of software, namely, spreadsheet applications, spreadsheet add-ins and Monte Carlo simulation packages, as *Commercial-Off-The-Shelf (COTS) Simulation Packages (CSPs)*.

Swain (2005) has made a comprehensive survey of commercially available simulation tools based on the information provided by vendors in response to a questionnaire requesting product information. This list consists of 56 CSPs, of which 12 are CSPs for Monte Carlo simulation. Spreadsheet applications such as Microsoft Excel™ and IBM Lotus1-2-3™ have not been included in Swain's survey, but will nonetheless be considered as CSPs since they can be used to create Monte Carlo simulations that are deployed in organizations (rather than developed entirely as programmed code). SunGard Analytics™ software is used in banking and finance for Monte Carlo-based credit risk simulations, and this too will be considered as a CSP for Monte Carlo simulation. These products have been specifically mentioned because the investment bank case study, discussed later in this paper, has used Monte Carlo applications built using Microsoft Excel™ and SunGard Analytics™. The next paragraph gives an overview of CSP Analytics™.

Analytics™ is the calculation engine for the Credient credit risk system that provides algorithms to calculate time-dependent profiles of credit exposure using MCSs (Credient Analytics, 2007). Analytics™ consists of three separate applications, namely, Analytics™ Desktop, Analytics™ Market Data Manager (MDM) and Analytics™ Server COM Object. The Analytics™ Desktop application is a standalone application that uses a calculation engine to construct and analyse financial portfolios. It links to the Market Data Manager to derive both current and historical market data which serve as inputs to these calculations. Analytics™ Server COM Object is essentially a COM interface to the Analytics™ Desktop and can be invoked by external systems.

Of the total 12 CSPs for Monte Carlo simulation that have been identified from Swain's survey, all the 12 are supported by the Windows Operating System platform. Furthermore, Microsoft Excel™ and SunGard Analytics™ are supported *only* on the Windows platform. The platform support for CSPs is important when considering different grid technologies that can be potentially used with existing CSPs. As has been mentioned earlier, *WinGrid* is a grid computing middleware that is targeted at Windows-based applications, and it aims to deliver low intervention technological solution to grid-enable existing Windows-based CSP applications.

3. GRID COMPUTING

Grid computing focuses on large-scale resource sharing, innovative applications and high-performance orientation, with the objective of coordinated resource sharing and problem solving in dynamic multi-institutional virtual organizations (Foster et al, 2001). The development of applications that can benefit from grid computing (faster execution speed, linking of geographically separated computational and data resources, workflows, interoperation of software, etc.) typically requires the installation of complex supporting software and an in-depth knowledge of how this complex supporting software works (Jaesun and Daeyeon, 2003). This software is commonly referred to as grid middleware. A grid middleware is a distributed computing software that integrates network-connected computing resources (computer clusters, data servers, standalone PCs, sensor networks, etc.), that may span multiple administrative domains, with the objective of making the combined resource pool available to user applications for number crunching, remote data access, remote application access, among others (Mustafee and Taylor, 2008). Examples of grid middleware include Globus (Foster et al, 2002), gLite (Berlich et al, 2005), and VDT (VDT, 2009).

The majority of grid computing middleware used for e-Science projects, such as those mentioned earlier, are targeted at dedicated and high-performance cluster computers running various UNIX/LINUX distributions - we refer to them as cluster-based grid computing middleware. *Cluster-based grid computing* can be contrasted with *desktop-based grid computing,* which refers to the aggregation of non-dedicated, de-centralized, commodity PCs connected through a network and running (mostly) the Microsoft Windows operating system (Mustafee and Taylor, 2009).

Desktop grid computing or desktop grids addresses the potential of harvesting the idle computing resources of desktop PCs for processing of parallel, multi-parameter applications which consist of a lot of instances of the same computation with its own input parameters (Choi et al, 2004). Middleware for cluster-based grid computing severely limits the ability to effectively utilize the vast majority of Windows-based resources that are common place in both enterprise and home environments, and therefore development of middleware for desktop-based grid computing is important with the growing industry interest in grids (Luther et al, 2005). The focus of this paper is on utilising the vast pool of underutilised desktop resources in an enterprise – we refer to this as *Enterprise-wide Desktop Grid Computing (EDGC)*.

EDGC is a grid infrastructure that is confined to an institutional boundary, where the spare processing capacities of an enterprise's desktop PCs are used to support the execution of the enterprise's applications (Chien et al, 2003). User participation in such a grid is not usually voluntary and is governed by enterprise policy. Applications such as Condor (Litzkow et al, 1988), Platform LSF (Zhou, 1992), Entropia DCGrid (Kondo et al, 2004), and Digipede Network (Digipede Technologies, 2009) are all examples of EDGC middleware.

In order to increase the enterprise-wide adoption of Windows-based grid technologies, it is also imperative to develop new grid software to specifically deal with Windows issues and grid-enable existing Windows applications. With regard to the former, for example, a .NET-based grid computing framework called Alchemi has been developed that provides the runtime machinery and programming environment required to construct Windows-based desktop grids and develop grid applications (Luther et al, 2005). As for the latter, it requires development of a grid-enabling solution that requires little or no change to existing Windows applications. As mentioned earlier in this paper, our system *WinGrid* aims to deliver such a low intervention technological solution to grid-enable existing Windows applications. The architecture of WinGrid is described next.

4. WINGRID: A MIDDLEWARE FOR DESKTOP GRID COMPUTING

WinGrid is an EDGC middleware that is targeted at the Windows operating system. WinGrid is based on the master-worker distributed computing architecture. This architecture consists of one master entity and multiple workers entities, where the master entity decomposes the problem into small tasks, distributes these tasks among a farm of worker processes and gathers the partial results to produce the final result of the computation; and the worker entities receive message from the master with the next task, process the task and send back the result to the master (Heymann et al, 2000).

WinGrid consists of four different components: the *Manager Application* (MA), the *WinGrid Job Dispatcher* (WJD), the *Worker Application* (WA) and the *WinGrid Thin Client* (WTC). The interactions between the WinGrid components are illustrated in Figure 1. A user submits a job through the MA (1), which in turn interacts with the WJD process (2) in the manager computer to send work (3) to the WinGrid workers and their WTCs (4). The WTC pass this work to their WA for processing (5) and returns the result to the WJD (6). The results of all the sub-jobs are communicated back to the MA which then collates the results and presents it to the user.

The reader is referred to Mustafee (2007a) and Mustafee and Taylor (2009) for more information on WinGrid.

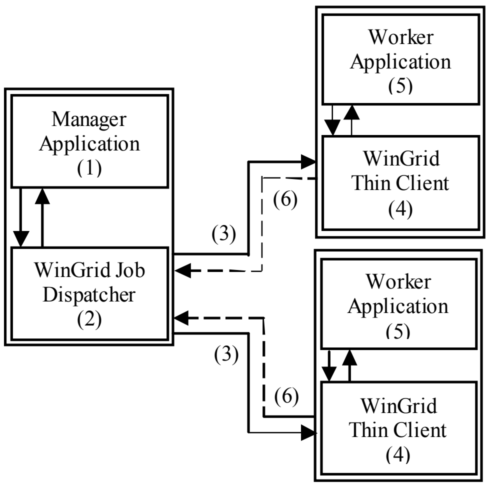

Figure 1 *WinGrid architecture*

5. CASE STUDY: IRS-RBF SIMULATION APPLICATION

The investment bank uses CSP SunGard Analytics™ for Monte Carlo-based credit risk simulations of counterparty transactions. Credit risk is the potential that the counterparty will fail to meet its obligations in accordance with the agreed terms (Basel Committee on Banking Supervision, 1999). Credit risk simulations are usually used to calculate the credit exposure over a period of time, i.e., in the event of a default, how large will the outstanding obligation be when the default occurs? The *IRS-RBF application* takes its name from two different products, namely, Interest Rate Swaps (IRS) and Risky Bond Forwards (RBF), which it simulates. This application comprises of different Excel™ spreadsheets, VBA modules and Analytics™.

The IRS-RBF application requires two stages of processing, we refer to it as *Stage one* and *Stage two* respectively, and involves three distinct operations that have to be "manually-executed". These operations are (1) *generate profiles*, (2) *create EPE tables*, and (3) *create PFE tables*. The *EPE/PFE create table* operations can only start after successful execution of the *generate profile* operation. The time taken to execute both these stages for the IRS-RBF application is shown in Table 1. The numbers of currencies that are simulated by these products are 23 and 13 respectively. Ideally, the bank would expect to run the IRS and RBF simulations with 37 currencies. This implies that the execution time would increase further. Please note that the data for Table 1 has been provided by the credit risk analysts who have developed the IRS-RBF application.

Table 1 *Execution time for different products using the original (non-grid version) IRS-RBF application*

Products	*Generate Profiles (Stage one)*	*Create EPE and PFE Tables (Stage two)*	Currencies
Interest Rate Swaps (IRS)	1 hour 15 minutes	12 hours	23
Risky Bond Forwards (RBF)	4 hours 30 minutes	1 hour 20 minutes	13

This case study was undertaken as it became essential to reduce the execution time of the simulation. With the aim of creating an enterprise desktop grid middleware capable of distributed parallel execution of the IRS-RBF application, WinGrid middleware was installed on computers belonging to the credit risk division of the investment bank. These computers already had Excel™ and Analytics™ installed on them. For the IRS-RBF application to utilise the resources made available through WinGrid, it had to be integrated to the WTC and the WJD (please refer to Section 4 and Figure 1).

WinGrid Worker Application (WA) and WTC: The WA is the *IRS-RBF application*. Integration of the Excel-based IRS-RBF application with WTC was achieved using Excel's COM interface. A custom built IRS-RBF adapter was developed which encapsulated the COM function calls required by WTC to interact with the IRS-RBF application.

WinGrid Manager Application (MA) and WJD: The MA that controls the IRS and RBF simulation execution is called the *WJD Application Specific Parameter (ASP) Tool for IRS-RBF application*. It is an Excel-based tool that consists of specific parameters that are required for processing the IRS-RBF application; for example, the name of the product to simulate (IRS or RBF), the operation to perform (*create table*, *create profiles* or both), the filename to simulate, etc. The WJD APS tool also consists of two other worksheets, namely "RBF" and "IRS", which contains data specific to the RBF and the IRS simulations respectively. Each worksheet has a list of currencies. Each currency is a separate unit of computation (job). The interaction between the MA and WJD is by means of an Excel Adapter. This adapter contains specific COM calls required by WJD to access the MA.

6. EXPERIMENTS, RESULTS AND DISCUSSION

Identical IRS-RBF experiments for this case study were conducted on (1) one dedicated WinGrid node (running both WJD and WTC), (2) 4 non-dedicated WinGrid nodes connected through the investment bank's corporate LAN, and (3) 8 non-dedicated WinGrid nodes connected with the corporate LAN. The grid-enabled IRS-RBF application was used for running experiments over the different test beds. The reason for not using the original IRS-RBF application for execution over one dedicated, standalone PC (i.e., test bed 1) was, (a) the original IRS-RBF application was modified to a large extent by the authors, and (b) to execute the IRS and RBF simulations using the original application meant that three different operations (create profiles, create EPE tables and create PFE tables) had to be manually invoked by the user. The execution of the grid-version of this application, on the other hand, was fully automated.

The experiments were conducted over a period of two days during normal working hours of the investment bank. The 4-node and the 8-node WinGrid experiments were run using production machines that were also being used by the analysts to do their jobs. The one node experiments were conducted using a PC that was not being used. There were three computers with 2.99GHz Intel P-IV processor and 512MB RAM. Five other computers had 2.13GHz Intel P-II processor with 2GB RAM. All the computers were installed with Microsoft XP operating system.

The dedicated WinGrid node used for performing the standalone experiments had a 2.99GHz Intel Pentium IV processor with 512MB RAM. The 4 non-dedicated WinGrid nodes used for the experiments comprised of different subsets of the machines at different times. The results of the IRS and RBF simulations are presented in Figure 2. These results are based on two separate runs for each workload (ideally we would have liked to conduct more runs; however, running experiments on a production network meant that we did not have the required flexibility. Furthermore, the time taken to execute the simulations using non-dedicated nodes would depend on the usage patterns of the underlying PCs). The execution of all the four workloads, pertaining to either IRS or RBF simulation, was fastest using the 8 non-dedicated WinGrid nodes. The slowest execution was recorded by the standalone, dedicated WinGrid node.

For workloads *[30 workunits (IRS)]*, *[69 workunits (IRS)]* and *[15 workunits (RBF)]* the time taken to execute the IRS-RBF simulations using the 4 node WinGrid test-bed was comparable to its 8 node counterpart. One reason for this may be that, with 8 nodes the number of Excel files created in Phase 2 (create EPE table) and Phase 3 (create PFE table) of the workflow are double the number of Excel files created when running the simulation using 4 nodes. Thus, the sequential MA operation that collated data from the EPE and PFE tables would generally take more time in the case of the former. An additional reason could be the specific usage pattern of the PCs during the experiments. It is therefore likely that the majority of the PCs in the 8 node set-up had their WTC clients manually or automatically shut down because the analysts were using the computers for their own work. The WTC program can be shut down manually through WinGrid's graphical user interface. This can also happen automatically as the WTC program is designed to continuously monitor CPU and the memory usage on a PC, and if the resource usage crosses the pre-determined CPU/RAM threshold levels then the user jobs are immediately stopped. Similarly, jobs are started automatically again when the CPU and memory usage decreases as a result of a resource not being used. Thus, *the time taken to execute the simulations on non-dedicated WinGrid nodes is very much related to the usage pattern of the underlying desktop PCs. Arguably, this is best shown by the results of workload [39 workunits (RBF)] in relation to its execution over 4 non-dedicated WinGrid nodes, where the time taken to complete the simulation is comparable to that of its standalone counterpart (approximately 1200 seconds for both).* In case the WinGrid nodes were dedicated resources, this result would have been surprising since it is generally expected that with more computers the execution time of the simulation would decrease (through adopting of "divide and conquer" strategy). However, in case of non-dedicated WinGrid nodes (as in our example), the availability of the PCs for executing the simulation actually depends upon its use by the user. Thus, if the users are working on their PCs then it is possible that WinGrid may not be able to execute the simulation until the users have finished their work. And this behavior of non-dedicated WinGrid resources is aptly reflected in Figure 2 below (especially, with regard to *workload [39 workunits (RBF)]*).

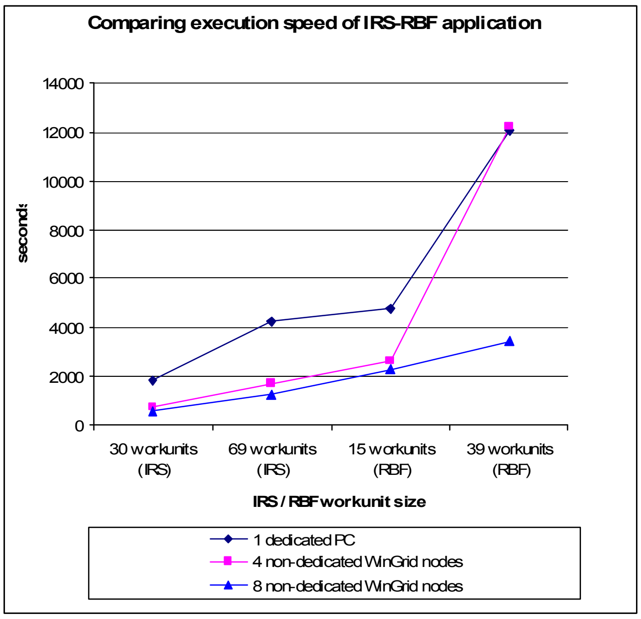

Figure 2 *Time taken to execute the IRS-RBF application using different workloads*

7. LESSONS LEARNT AND CONCLUSIONS

In the concluding section of this paper we describe some of the lessons that we learnt from this case study. The reader should note that the lessons learnt focus on the process of implementing a desktop grid solution and our engagement with the stakeholders (and the experiment results).

- Show a technology demonstrator

Our experience shows that developing a technology demonstrator (a demo!) and presenting it to the stakeholders is very important for gaining their confidence. This also provides an opportunity for the stakeholders to give their inputs and enables them to have a "picture in mind" of the intended end-product.

- Good relation with stakeholder is the key

It is imperative that the stakeholders consider themselves as part of the project. The onus is on the researcher/developer to build such a rapport with the stakeholders.

- Security of the network is paramount

Computer security is understandably very important for any organisation. Computer support is generally paranoid about installing software that has not been internally tested and certified. The cause for their concern increases manifolds in cases where the software being installed opens communication ports for computers to communicate with each other – as is the case with grid computing middleware like WinGrid. Thus, any solution should take into account the security concerns of the organisation and try to work around it. WinGrid therefore uses only one communication port. This is unlike several other grid computing middleware that use multiple ports for communication, for example, Condor uses multiple, bi-directional, static and dynamic ports (Beckles et al, 2005).

- Provide regular updates

From our experience we find that giving stakeholders regular updates is very important. Showing demos of ongoing work is very effective.

- Write a user guide for the application

The researcher/developer should spend time in writing a comprehensive user manual. This manual should include system requirements, installation instructions, information on software dependencies, instructions for working effectively with the software, troubleshooting, FAQ, etc. For example, WinGrid has a comprehensive user guide titled *"WinGrid 0.2 User Documentation: WinGrid-Excel Integration for Speeding up IRS and RBF Simulations"* (Mustafee, 2007b).

- Provide ongoing support

It is important to recognise that the organisation will generally use the software until ongoing support is provided. The ongoing support is required since bugs may be encountered in the program, the requirements may change, new functionality has to be introduced, etc. Being researchers, it is difficult to provide such support. It is therefore important that the computer support in the organisations where the software is deployed be trained to take-over the maintenance of the software. However, this is only possible if the support personnel have the necessary expertise to do so and there are necessary indemnity clauses in the original contract (between the University/researcher and the organisation).

In conclusion the deployment of WinGrid and the collaboration yielded successful results. It is hoped that our lessons learnt will be useful to others undertaking such industry-based ventures in the future.

ACKNOWLEDGEMENTS

The authors would like to thank Robert Watson, Rahul Talwalkar and Jonathan Berryman from the counterparty risk management group at the investment bank where the case study was conducted.

REFERENCES

D. Anagnostopoulos and M. Nikolaidou (2003) "Executing a Minimum Number of Replications to Support the Reliability of FRTS Predictions". In *Proceedings of the 7th IEEE International Symposium on Distributed Simulation and Real-Time Applications*, October 23 – 25 2003, Delft, The Netherlands, pp. 138-146.

M. Baker, R. Buyya, and D. Laforenza (2002). "Grids and grid technologies for wide-area distributed computing". *Software - Practice and Experience*, 32(15): 1437-1466.

Basel Committee on Banking Supervision (1999). "Principles for the management of credit risk". Available online *www.bis.org/publ/bcbs54.pdf*. Last accessed 17 November 2009.

B. Beckles, S. Se-Chang, and J. Kewley (2005). "Current methods for negotiating firewalls for the Condor system". In *Proceedings of the 4th UK e-Science All Hands Meeting*. Available online http://www.cs.wisc.edu/condor/doc/Condorand Firewalls.pdf. Last accessed 18 November 2009.

R. Berlich, M. Kunze, and K. Schwarz (2005). "Grid computing in Europe: from research to deployment". In *Proceedings of the 2005 Australasian Workshop on Grid Computing and e-Research*, pp. 21-27. Australian Computer Society, Darlinghurst, Australia.

W. E. Biles, and J. P.C. Kleijnen (2003). "Statistical Methodology for WEB-Based Simulation". In *Proceedings of the 7th IEEE International Symposium on Distributed Simulation and Real-Time Applications*, October 23 – 25 2003, Delft, The Netherlands, pp. 147-149.

A. Chien, B. Calder, S. Elbert, and K. Bhatia (2003). "Entropia: architecture and performance of an enterprise desktop grid system". *Journal of Parallel and Distributed Computing*, 63(5): 597-610.

S. Choi, M. Baik, C. Hwang, J. Gil, and H. Yu (2004). "Volunteer availability based fault tolerant scheduling mechanism in desktop grid computing environment". In *Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications*, pp. 366-371. IEEE Computer Society, Washington, DC, USA.

F. J. Corbato, and V. A. Vyssotsky (1965). "Introduction and overview of the Multics system". In *Proceedings of the AFIPS Fall Joint Computer Conference*, pp. 185–196. IEEE Educational Activities Department, Piscataway, NJ, USA. Available online *http://www.multicians.org/fjcc1.html*. Last accessed 17 November 2009.

Credient Analytics (2007). "Credit risk management system - Credient Analytics 2.3 user guide". SunGard Corporation *http://www3.sungard.com/financial/*. Last accessed 9 May 2008.

Digipede Technologies (2009). "The Digipede Network". Accessible online *http://www.digipede.net/products/digipede-network.html*. Last accessed 17 November 2009.

I. Foster, C. Kesselman, and S. Tuecke (2001). "The anatomy of the grid: enabling scalable virtual organizations". *International Journal of High Performance Computing Applications*, 15(3): 200-222.

I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke (2002). "Grid services for distributed system integration". *IEEE Computer*, 35(6): 37-46.

T. N. Herzog, and G. Lord (2002). "Applications of monte carlo methods to finance and insurance". Winstead, Conn: ACTEX Publications. *Available online http://books.google.com*. Last accessed 11 March 2007.

E. Heymann, M. A. Senar, E. Luque, and M. Livny (2000) "Adaptive scheduling for master-worker applications on the computational grid". *Grid Computing - GRID 2000, Lecture Notes in Computer Science*; R. Buyya and M. Baker, Eds.; Springer Berlin / Heidelberg, pp. 214–227.

H. Jaesun, and P. Daeyeon (2003). "A lightweight personal grid using a supernode network". In *Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, pp. 168-175.

D. Kondo, A. Chien, and H. Casanova (2004). "Resource management for rapid application turnaround on enterprise desktop grids". In *Proceedings of the 2004 Conference on Supercomputing (SC'04)*, paper 17. IEEE Computer Society, Washington, DC, USA.

M. Litzkow, M. Livny, and M. Mutka (1988). "Condor - a hunter of idle workstations". In *Proceedings of the 8th International Conference of Distributed Computing Systems*, pp.104-111. IEEE Computer Society, Washington, DC, USA.

A. Luther, R. Buyya, R. Ranjan, and S. Venugopal (2005). "Alchemi: a .NET-based enterprise grid computing system". In *Proceedings of the 6th International Conference on Internet Computing (ICOMP'05)*, pp. 269-278. CSREA Press, USA.

N. Mustafee (2007a). "A grid computing framework for commercial simulation packages". *PhD thesis*. School of Information Systems, Computing and Mathematics, Brunel University, UK.

N. Mustafee (2007b). "WinGrid 0.2 User Documentation: WinGrid-Excel Integration for Speeding up IRS and RBF Simulations".

Included as an appendix in Ph.D. thesis (Mustafee, 2007a).

N. Mustafee, and S. J. E. Taylor (2008). "Investigating Grid Computing Technologies for Use with Commercial Simulation Packages". In *Proceedings of the OR Society 4th Simulation Workshop (SW08)*, pp. 297-307.

N. Mustafee, and S. J. E. Taylor (2009). "Speeding Up Simulation Applications Using WinGrid". *Concurrency and Computation,* 21(11): 1504-1523.

N. Mustafee, A. Alstad, B. Larsen, S. J. E. Taylor, and J. Ladbrook (2006). "Grid-enabling FIRST: speeding up simulation applications using WinGrid". In *Proceedings of the 10th International Symposium on Distributed Simulation and Real-Time Applications (DSRT 2006)*, pp. 157-164. IEEE Computer Society, Washington, DC, USA.

J. L. Paris, and H. Pierreval (2001). "A Distributed Evolutionary Simulation Optimization Approach for Configuration of Multiproduct Kanban Systems". *International Journal of Computer Integrated Manufacturing*, 14 (5), 421-430.

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

R. Y. Rubinstein (1981). "Simulation and the monte carlo method". *John Wiley & Sons*, Inc: New York, USA.

J. J. Swain (2005). "Gaming reality: biennial survey of discrete-event simulation software tools". *OR/MS Today (December 2005).* Institute for Operations Research and the Management Sciences (INFORMS), USA. Available online *http://www.lionhrtpub.com/orms/orms-12-05/frsurvey.html*. Last accessed 17 November 2009.

J. J. Swain (2007). INFORMS simulation software survey. OR/MS Today. Institute for Operations Research and the Management Sciences (INFORMS), USA. *Available http://www.lionhrtpub.com/orms/surveys/Simulation/Simulation.html*. Last accessed 4th April 2007.

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

VDT (2009). "Virtual Data Toolkit (homepage)". Website *http://vdt.cs.wisc.edu*. Last accessed 17 November 2009.

E. Yücesan, Y. C. Luo, C. H. Chen, and I. Lee (2001) "Distributed Web-Based Simulation Experiments for Optimization". *Simulation Practice and Theory*, 9(1), 73-90.

S. Zhou (1992). "LSF: Load sharing in large-scale heterogeneous distributed systems". In *Proceedings of the 1992 Workshop on Cluster Computing*. Supercomputing Computations Research Institute, Florida State University, Florida, USA.

AUTHOR BIOGRAPHIES

NAVONIL MUSTAFEE is a lecturer in the School of Business and Economics at Swansea University. Prior to this, he was a Research Fellow in Brunel University and Warwick Business School. His research interests are in e-Infrastructures and Grid Computing, Information Systems, Operations Research and Healthcare Simulation. He completed his PhD in Information Systems and Computing from Brunel University in 2007. He is a member of the drafting group of the COTS Simulation Package Interoperability Product Development Group (CSPI-PDG) under the Simulation Interoperability Standards Organization. His email address is N.Mustafee@Swansea.ac.uk.

SIMON J E TAYLOR is the Founder and Chair of the COTS Simulation Package Interoperability Product Development Group (CSPI-PDG) under the Simulation Interoperability Standards Organization. He is the co-founding Editor-in-Chief of the UK Operational Research Society's (ORS) Journal of Simulation and the Simulation Workshop series. He was Chair of ACM's Special Interest Group on Simulation (SIGSIM) (2005-2008). He is a Reader in the School of Information Systems, Computing and Mathematics at Brunel and has published over 100 articles in modelling and simulation. His recent work has focused on the development of standards for distributed simulation in industry. http://people.brunel.ac.uk/~csstsjt/