

Grid-aware Large Scale Distributed Simulation of Agent-based Systems

Yi Zhang, Georgios Theodoropoulos, Rob Minson
University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK
{yxz, gkt, rzm}@cs.bham.ac.uk

Stephen Turner, Wentong Cai
School of Computer Engineering
Nanyang Technological University
Singapore 639798
{ASSJTurner,aswtcai}@ntu.edu.sg

Yong Xie
Computer Science Department
Singapore-MIT Alliance
Singapore 117576
smaxy@nus.edu.sg

Brian Logan
School of Computer Science & Information Technology
University of Nottingham
Nottingham NG8 1BB, UK
bsl@cs.nott.ac.uk

Abstract

The development of many complex simulation applications requires collaborative effort from researchers with different domain knowledge and expertise, possibly at different locations. These simulation systems often require huge computing resources and data sets, which may be geographically distributed. In order to support collaborative model development and to cater for the increasing complexity of such systems, it is necessary to harness distributed resources over the Internet. While the High Level Architecture (HLA) enables interoperability, it does not provide support for collaborative development of simulation applications, nor does it provide any mechanism for managing the resources where the simulation is being executed. The emergence of Grid technologies provide exciting new opportunities for large scale distributed simulation, enabling collaboration and the use of distributed computing resources, while also facilitating access to geographically distributed data sets. This paper presents HLA_GRID_REPAST, a system for executing large scale distributed simulations of agent based systems over the Grid. Performance results in LAN and WAN

environments are also presented.

1 Introduction

Modelling and simulation is an essential tool in many areas of science and engineering, for example, for analysing natural phenomena or predicting the behaviour of new systems being designed. The development of such complex simulation applications usually requires collaborative effort from researchers with different domain knowledge and expertise, possibly at different locations. Furthermore, these simulation systems often require huge computing resources and the data sets required by the simulation may also be geographically distributed (e.g. in a supply chain simulation involving different companies, the most up-to-date data will be in the individual companies). In order to support collaborative model development and to cater for the increasing complexity of such systems, it is necessary to harness distributed resources over the Internet.

While the High Level Architecture (HLA) enables interoperability and the construction of large-scale distributed simulations using existing and possibly dis-

tributed simulation components, it does not provide support for collaborative development or configuration of simulation applications, nor does it provide any mechanism for managing the resources where the simulation is being executed.

The emergence of Grid technologies provide exciting new opportunities for large scale distributed simulation, enabling collaboration and the use of distributed computing resources, while also facilitating access to geographically distributed data sets. In the last few years, there has been an increasing interest in taking advantage of Grid technologies to execute HLA simulations over the Internet.

A seminal work in this direction is the Extensible Modeling & Simulation Framework¹ (XMSF), a collaborative initiative to develop a web enabled RTI [9, 10]. Within the XMSF framework, multiple federates reside as web services on a WAN and the Federation's FOM is mapped to an XML tagset, allowing the inter operation with other distributed applications supported by web services. The federates communicate using the Simple Object Access Protocol (SOAP) and the Blocks Extensible Exchange Protocol (BEEP).

Another important initiative is HLA_GRID, originally developed at the Nanyang Technological University in Singapore [3, 12]. In HLA_GRID, federates can be instantiated as Grid services which are used to facilitate communication between Grid service federates and the RTI.

In [5] we presented HLA_REPAST, a system for executing agent-based simulations with HLA. HLA_REPAST permits integration, through an HLA federation, of multiple instances of the Java-based lightweight-agent simulation toolkit RePast. This paper discusses the integration of HLA_REPAST And HLA_GRID into a new system, HLA_GRID_REPAST to execute large scale distributed simulations of agent-based systems over the GRID. HLA_GRID_REPAST forms part of a larger collaborative project which aims to develop a "Grid plug-and-play distributed simulation system", a distributed collaborative simulation environment where researchers with different domain knowledge and expertise, at different locations, develop, modify, as-

¹<http://www.movesinstitute.org/xmsf/xmsf.html>

semble and execute distributed simulation components over the Grid².

The rest of the paper is organized as follows: Sections 2 and 3 provide some background information on the HLA_GRID and HLA_REPAST systems respectively. HLA_GRID_REPAST is described in section 4. Section 5 provides an evaluation of the performance of HLA_GRID_REPAST in LAN and WAN environments and discusses security issues. Finally, section 6 concludes the paper and provides some ideas for future work.

2 HLA_Grid

HLA_GRID is a framework designed to extend the HLA to the Grid. In particular, it focuses on improving the interoperability and composability of simulation components.

The framework, which is illustrated in figure 1, achieves interoperability between different simulators (federates) by using a Federate-Proxy-RTI architecture, in which different participants (clients) in the same simulation run their federate codes at their local sites, and the RTIEXEC and FEDEXEC are executed on the remote resource. A new entity, *proxy*, is introduced to act on behalf of the clients' federate code to communicate with proxies of other clients through the RTI. Proxies are executed at remote grid resources. Federate codes and their respective proxies communicate with each other through Grid services, and a Grid-enabled HLA library, which provides the standard HLA API to the federate codes, is implemented to translate the communications into Grid services invocations.

HLA_GRID includes additional Grid services to support the creation of the RTI, discovery of federations, etc. The framework hides the heterogeneity of the simulators, execution platforms, and how the simulators communicate with the RTI. All interfaces used

²"DS-GRID: Large Scale Distributed Simulation on the Grid". This is one of only four such projects funded by the UK e-Science Programme in 2003. It is a collaboration between MeSC (the Midlands e-Science Center of Excellence) at the University of Birmingham, UK, the Intelligent Agents Group at the University of Nottingham and the Parallel and Distributed Computing Centre (PDCC), at the Nanyang Technological University, Singapore. <http://www.cs.bham.ac.uk/research/projects/dsgrid>.

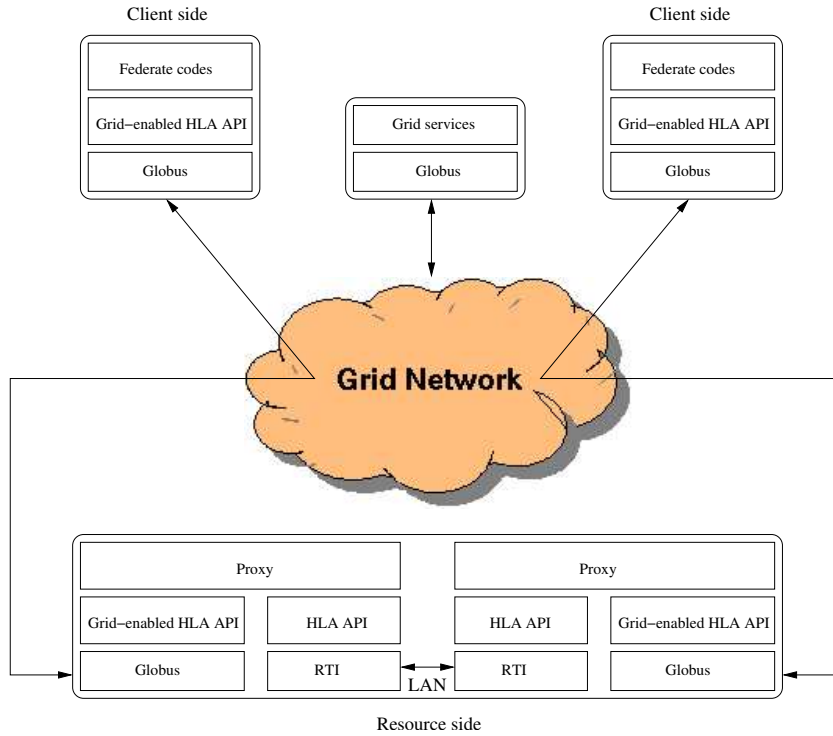


Figure 1. Architecture of Proxy-based HLA simulation on the Grid.

in the HLA_GRID comply with the standard HLA interface specification.

A prototype of HLA_GRID has been implemented in Java using the Globus Toolkit version 3. The prototype includes standard HLA/RTI APIs to support Federation, Time, Object, Declaration and Ownership Management. The prototype has been tested using the benchmark programs in the DMSO RTI package. More details of HLA_GRID's design, implementation and experiments can be found in [3, 12].

3 HLA_RePast

The REPAST system [2] is a Java-based toolkit for the development of lightweight agents and agent models. It was developed at the University of Chicago's Social Science Research Computing division and is derived from the Swarm simulation toolkit. It has become a popular and influential toolkit, assessed by [11] as the most effective development platform currently available for large-scale simulations of social phenomena.

The system provides an inter-dependent collection of tools and structures, which are generally useful for

the simulation of agents and a sequential discrete event simulation kernel for the execution of the model.

HLA_REPAST is a middleware layer which enables the execution of a federation of multiple interacting instances of REPAST models within HLA as depicted in figure 2.

The main task of the HLA_REPAST middleware is to detect the occurrence of events in the REPAST model and consistently, reliably and transparently communicate them to the RTI. To achieve that, HLA_REPAST provides mechanisms for mapping REPAST state-transitions to events in RTI (via the `PublicObject` scheme), for conflict resolution and for integrating the REPAST scheduling system with RTI (as depicted in figure 3). More information about the HLA_REPAST system can be found in [5].

4 Design of HLA_Grid_RePast

HLA_GRID_REPAST is designed for executing distributed, large scale simulations of agent-based systems over Grids. It integrates HLA_REPAST and HLA_GRID as depicted in Figure 4. It acts as a middleware to glue simulation code written in REPAST

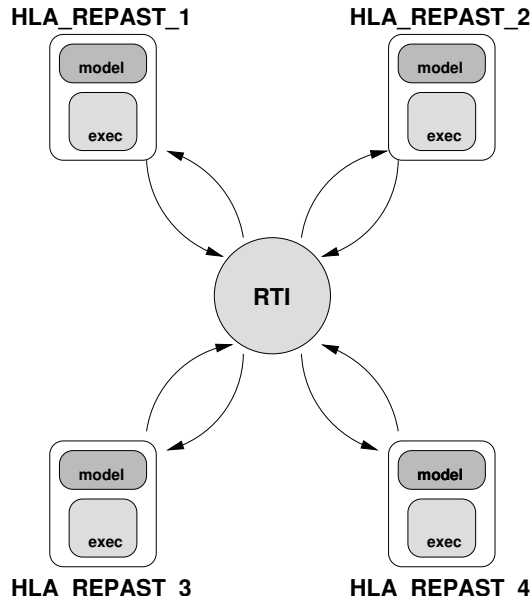


Figure 2. An HLA_REPASt Federation

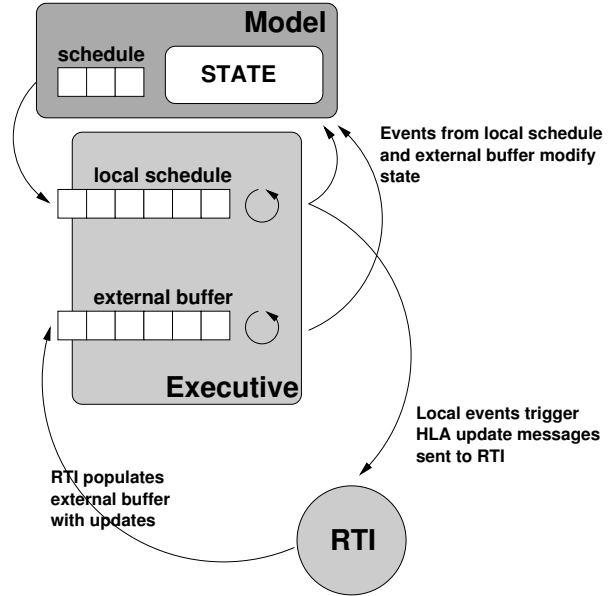


Figure 3. The HLA_REPASt Model-Executive Interface (A Single HLA_REPASt Federate)

with HLA.

Each federate in HLA_GRID_REPASt is divided into two parts: the client side and the proxy side, which usually run on different machines. The client side contains the following components: the REPASt agent-based simulation code, the HLA_REPASt middleware, the RTI proxy ambassador and the Federate Ambassador Service from HLA_GRID. The REPASt agent-based simulation code must satisfy the HLA_REPASt guidelines [5]. The proxy side contains the RTI Ambassador Service and the Federate proxy ambassador. Both the RTI Ambassador Service and the Federate Ambassador Service are implemented as GRID services and are shown in shadow boxes in Figure 4.

Before the simulation starts, the RTI ambassador services will be started at the proxy side and each federate will be appointed with a RTI ambassador service. The information of the assigned RTI ambassador service will be passed to the RTI proxy ambassador. When the simulation starts, the Federate Ambassador Service of each federate will be initialised and registered with the corresponding RTI ambassador service. During the simulation, the events and state changes generated by REPASt will be monitored by the HLA_REPASt middleware, which will generate

the necessary RTI function calls for them. These RTI function calls will be passed to the RTI Proxy ambassador which will translate these function calls into invocations of the RTI Ambassador Grid service. The RTI Proxy ambassador and the RTI Ambassador Grid service run on two different machines. Finally, the RTI Ambassador Grid service will communicate with the RTIEXEC by invoking corresponding RTI function calls. The return value of the RTI function calls will be sent back as the return value of the GRID service call while the runtime exceptions of RTI function calls will be sent back by means of Apache Axis faults, which will be described later. When the RTIEXEC needs to communicate with a federate, the federate function calls will be translated into invocations of the federate ambassador service by the federate proxy ambassador. The federate ambassador service will pass the function calls to the HLA_REPASt middleware, which will then convert the federate calls into REPASt events and put them into the event scheduler.

Modifications have been made to both HLA_GRID and HLA_REPASt in order to integrate them into HLA_GRID_REPASt. Two major modifications are concerned with object encoding and remote exception

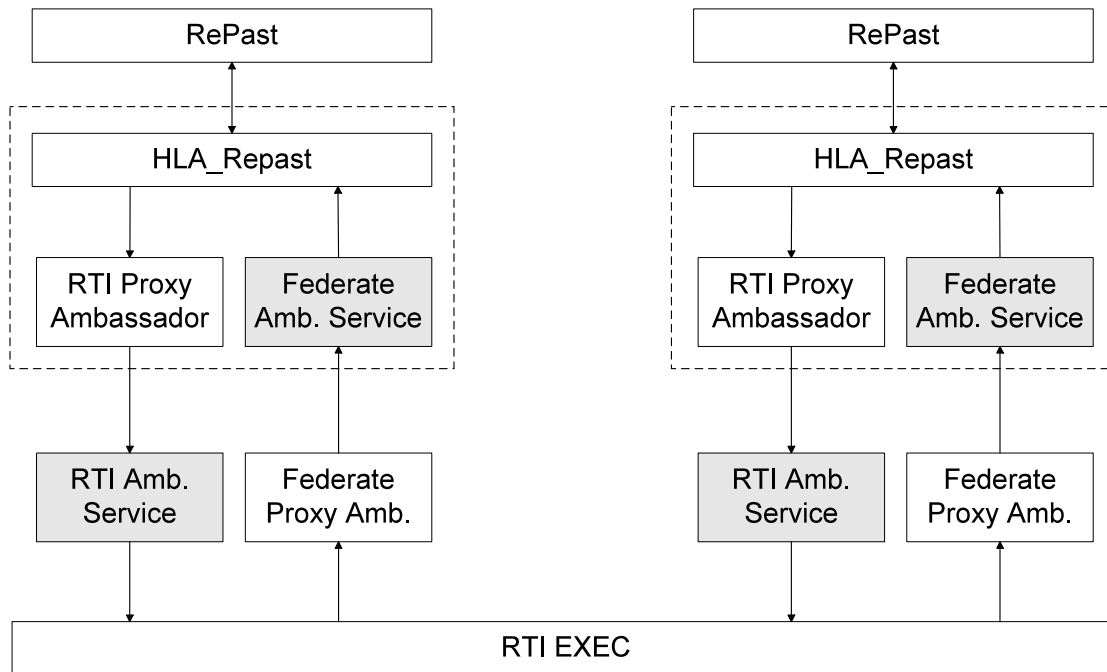


Figure 4. Structure of HLA_GRID_REPAST

handling. The former is required by the SOAP protocol while the latter is designed for improving performance.

4.1 Object Encoding

The object encoding scheme has been modified to cooperate with HLA_GRID. HLA_REPAST uses a *PublicObject* scheme to translate Java expressions into HLA function calls [5]. Java objects in the *PublicObject* scheme may be transferred between federates during a simulation. In HLA_REPAST, the Java objects are encoded into Java Byte arrays. However, Java Byte arrays cannot be directly applied within HLA_GRID, which, like other grid services, uses SOAP as the communication protocol between clients and servers. SOAP is based on XML which does not support binary data. The problem can be solved by modifying HLA_GRID to support either the *SOAP With Attachments* (SwA) proposal [1] or the *WS-Attachments* proposal [6]. Such an approach would require modifications to the Globus Toolkit and would affect the interoperability of the HLA_GRID_REPAST. Instead of modifying the Globus Toolkit, a new ob-

ject encoding scheme for Java objects is employed in HLA_GRID_REPAST as follows. First, a Java object is encoded into a Java byte array as in HLA_REPAST. Then, the base64 encoding scheme (from the Apache project) is used to encode the Java byte array into a Java String Object. Finally, a Java Byte array is generated from the Java String and passed to HLA_GRID to be used in a SOAP message.

4.2 Exception Handling

In HLA_GRID_REPAST, Java exceptions generated in Grid service calls are packed into Apache Axis Faults and passed back to the client side. In addition, remote exception handling has been introduced in HLA_GRID_REPAST. The RTI implementation used in HLA_GRID_REPAST is the DMSO RTI NG 1.3. In this implementation, Java exceptions work as return values of RTI function calls in many situations. HLA_REPAST uses heavily such functions, e.g. `unconditionalAttributeOwnershipDivestiture`. In many cases, the exceptions (return values) are discarded immediately without further handling. This behaviour causes very small over-

head when the simulation federates and the RTI ambassadors live in the same machines. This is not the case in HLA_GRID, however, where the federate simulations and the RTI ambassadors are in different machines connected via a Grid. In this case, such exceptions of RTI calls would waste substantial valuable network bandwidth. To address this issue, in HLA_GRID such exceptions are registered with the RTI ambassador services after connections between federate simulations and RTI ambassador services are established through the RTI proxy ambassador. When RTI calls generate exceptions, the RTI ambassador services will only pass exceptions that are not registered back to the federate simulations. In this way, registered RTI call exceptions can be handled remotely at the RTI ambassador service side. The performance advantage of remote exception handling is substantial and is demonstrated in the next section.

5 Evaluation of HLA_GRID_REPAST

To evaluate the robustness and performance of HLA_GRID_REPAST we have developed an agent-based federation using TILEWORLD as a test case. Experimental results have been obtained by executing simulations over a Grid in a LAN environment and a WAN environment between the UK (Birmingham) and Singapore.

Tileworld is a well established testbed for agents [8]. It consists of a grid-like environment consisting of tiles, holes and obstacles, and agents whose goal is to score as many points as possible by pushing tiles to fill in the holes (figure 5). Whenever a hole is filled, the agent who dropped the last tile will be given a score as the initial depth of the hole. The environment is dynamic: tiles, holes and obstacles appear and disappear at rates controlled by the simulation developer. TILEWORLD has been used to study commitment strategies (i.e., when an agent should abandon its current goal and replan) [7] and in comparisons of reactive and deliberative agent architectures [8]. It has also been used as a benchmark for evaluating HLA based simulations of agent-based systems using the HLA_AGENT [4] and HLA_REPAST systems.

For HLA_GRID_REPAST, a TILEWORLD simulation developed for HLA_REPAST has been used, as described in [5]. In that implementation, the envi-

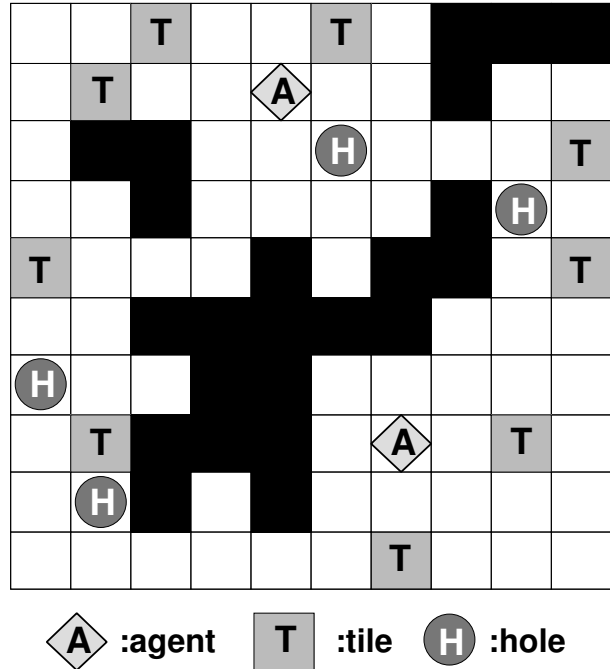


Figure 5. A 20x20 TILEWORLD

ronment of Tileworld containing the tiles, holes and obstacles of the model is simulated by a single federate while the agents of the simulation are simulated by one or more agent federates. In general, the simulation will contain one environment federate and at least one agent federate.

5.1 Performance in a LAN Environment

The LAN environment is a PC cluster consisting of one master node and 54 worker nodes³. Each node has 2G bytes of memory and two Intel Xeon 3GHz processors while connectivity is provided by a Gigabit Ethernet Switch. The RTI NG 1.3 with Java bindings is used in the experiments.

First, the performance of HLA_GRID_REPAST and HLA_REPAST with one agent federate simulating different number of agents is compared, in order to evaluate the overhead introduced by the different layers of the middleware (HLA_GRID and HLA_GRID_REPAST). In this case, the simulation consists of two federates, namely one environment federate, which simulates the TILEWORLD environ-

³<http://www.ep.ph.bham.ac.uk/cluster>

ment, and one agent federate, which hosts all agents. For HLA_REPAST, the two federates run on two separate nodes. For HLA_GRID_REPAST, in addition to the two nodes hosting the two TILEWORLD federates, two different nodes are hosting the respective HLA_GRID Proxy services for the two federates. In all experiments, the RTIEXEC is running on a separate node.

The results are shown in figure 6. For HLA_GRID_REPAST two sets of experiments have been performed, one with local (HLA_GRID_REPAST(L) in the figure), and one with remote (HLA_GRID_REPAST(R)) exception handling.

Clearly, the performance of HLA_GRID_REPAST with local exception handling is lower compared with HLA_REPAST and HLA_GRID_REPAST(R). Notably, HLA_GRID_REPAST exhibits a higher overhead when the number of agents is small.

As the number of agents increases, the total simulation time with HLA_GRID_REPAST(L) increases, albeit not substantially. This can be attributed to HLA_REPAST's heavy usage of RTI functions that generate Java exceptions which in turn are just caught and discarded. Thus, the system spends most of its time sending and receiving RTI exceptions, which have no effect on the simulation. As communication and computation tasks in HLA_GRID_REPAST are carried out concurrently by separate Java threads, the communication time and computation time overlap. In the case of HLA_GRID_REPAST(L), the frequent communication between federates and proxy services is the dominant factor of the performance of the simulation. To improve the performance of HLA_GRID_REPAST, remote exception handling has been developed as described in the previous section.

Although HLA_GRID_REPAST with remote exception handling performs much better than HLA_GRID_REPAST with local exception handling, HLA_GRID_REPAST(R) still has a large overhead compared with HLA_REPAST when the number of agents is small (less than 12 in our experiments). As the number of agents increases, the computation load becomes the dominant factor compared to the overhead introduced by HLA_GRID_REPAST(R) and the total simulation time for HLA_GRID_REPAST(R) is relatively close to that of HLA_REPAST.

Figure 7 shows the speedup of HLA_GRID_REPAST(R) in a distributed environment where 128 agents are distributed in multiple federates (1 to 16). In HLA_GRID_REPAST, each federate requires one node for the federate itself and one for the proxy services. As expected, the simulation times of both HLA_GRID_REPAST and HLA_REPAST decrease as the number of agent federates increases. However, as the computational load in each federate decreases (i.e. as the number of federates increases), the communication overhead becomes increasingly dominant. Because HLA_REPAST has less communication overhead than HLA_GRID_REPAST, it exhibits better speedup than HLA_GRID_REPAST.

5.2 Performance in a WAN environment

Experiments to evaluate the performance of HLA_GRID_REPAST were also conducted in a WAN environment between Singapore and Birmingham, UK. In these experiments, the RTIEXEC and the TILEWORLD environment federate are in Birmingham while the agent federate is in Singapore. Two HLA_GRID proxy services are also hosted in Birmingham for the environment and agent federates respectively.

As in the first set of experiments in the LAN environment only one agent federate is used while the number of agents ranges from 1 to 128. The results for a WAN environment are depicted in Figure 8 (the Y-axis in Figure 8 is in logarithmic scale)

Comparing figures 8 and 6, it is clear that simulations on both HLA_GRID_REPAST and HLA_REPAST take much longer to finish in the WAN environment than in the LAN environment. This is not surprising, as the bandwidth is much smaller and the communication latency much higher in the WAN environment. The most noticeable difference is that HLA_GRID_REPAST performs much worse than HLA_REPAST in the WAN environment. In figure 6, when the number of agents is large, the total simulation time with HLA_GRID_REPAST is close to that with HLA_REPAST. In figure 8, the total simulation time with HLA_GRID_REPAST is at least 20 times longer than that with HLA_REPAST. As the computation loads for the same number of agents

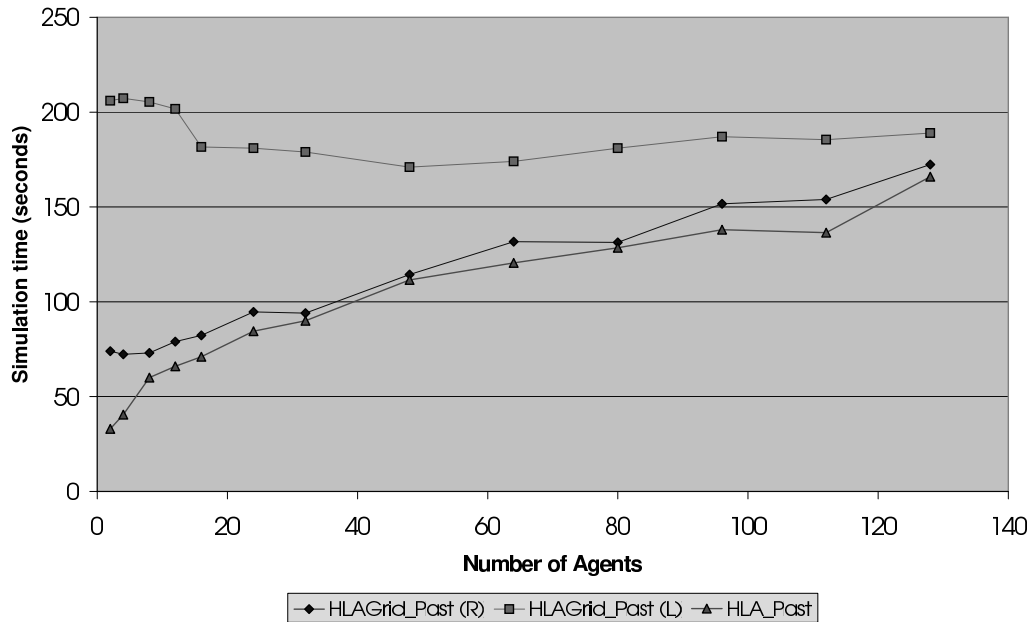


Figure 6. Performance of HLA_GRID_REPAST and HLA_REPAST with a single agent federate

are the same for both environments, this difference in simulation times between HLA_GRID_REPAST and HLA_REPAST can be attributed to the communication load which in HLA_GRID_REPAST is higher. Further experiments are currently being performed to further analyse the communication costs in the WAN environment.

5.3 Security Issues

Security and firewalls is a major obstacle for constructing large scale distributed simulations. In our experiments in the WAN environment, both sites have firewalls installed on their network boundaries and the configuration of firewalls is maintained by the respective network support personnel. It is tedious and very time-consuming to change and add firewall rules which satisfy both the requirements of the distributed simulation and the network security requirements at the different sites each time. Compared with the requirements of HLA_REPAST for firewalls, the ones of HLA_GRID_REPAST are much simpler. In the experiments of figure 8 for instance, there are two fed-

erates and one RTIEXEC. For the experiments with HLA_REPAST, there were two machines in Birmingham side and one machine in Singapore. Using HLA_REPAST together with RTI NG 1.3, the machine in Singapore had to communicate with both machines in Birmingham and therefore, there was a need to apply for two new firewall rules in Singapore. By using HLA_GRID_REPAST, the machine in Singapore only needs to communicate with the machine running the HLA_GRID proxy service in Birmingham, and thus, only one firewall rule is needed. If more federates are added in Birmingham, new firewall rules have to be added for HLA_REPAST but no more rules are needed for HLA_GRID_REPAST. An alternative solution would require the use of VPNs and IPv6 or tunnelling.

6 Conclusions and Future Work

This paper has presented HLA_GRID_REPAST, a framework for building large scale distributed agent-based simulations. HLA_GRID_REPAST combines

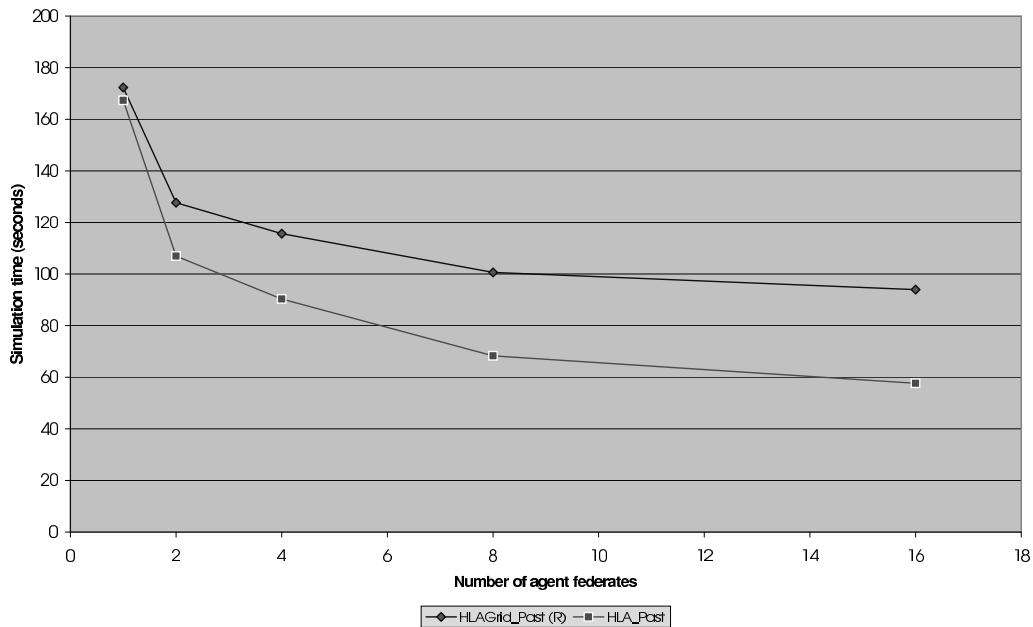


Figure 7. Performance of HLA_GRID_REPAST and HLA_REPAST with multiple agent federate

HLA_GRID and HLA_REPAST toolkits and allows writing and deploying HLA agent-based simulation over Grids, presenting federates as Grid services.

Our experiments show that the performance of HLA_GRID_REPAST is comparable to that of HLA_REPAST in a LAN environment, but in a WAN environment a substantial drop in performance is observed.

Future work will analyse the network traffic pattern generated by HLA_GRID_REPAST and HLA_REPAST and investigate the causes of the performance degradation of HLA_GRID_REPAST in a WAN environment. A large and complex case study with HLA_GRID_REPAST is also under planning. Ultimately, our vision is a "Grid plug-and-play distributed simulation system", a distributed collaborative simulation environment where researchers with different domain knowledge and expertise, possibly at different locations, develop, modify, assemble and execute distributed simulation components over the Grid; the required components reside in the Grid as Grid services which the system will automatically discover, se-

mantically match (with each other and with the simulation goals), formally verify and assemble.

7 Acknowledgements

A number of people have helped in many different ways to make this work possible. In particular we would like to thank Mike Lees, Ben Stone, Steve Pillinger, Jim Nichols, Irene Goh, and Tan Joo Sing. This work is funded by the UK e-Science Programme.

References

- [1] J. J. Barton, S. Thatte, and H. F. Nielsen. Soap messages with attachments. W3C Note, Dec. 2000.
- [2] N. Collier. Repast: An extensible framework for agent simulation. http://repast.sourceforge.net/docs/repast_intro_final.doc, June 2003.
- [3] W. Jie, T. Zang, Z. Lei, W. T. Cai, S. J. Turner, and L. Wang. Constructing an ogsa-based grid computing platform. In *International Conference on Scientific and Engineering Computation (IC-SEC 2002)*, pages 738–741, 2002.

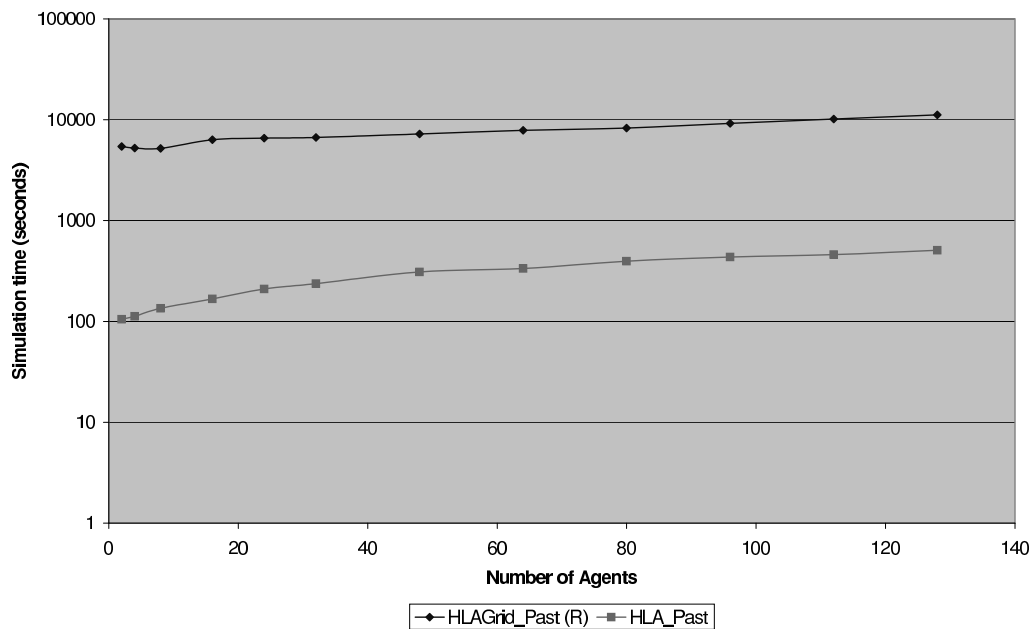


Figure 8. Performance of HLA_GRID_REPAST and HLA_REPAST with one agent federate in a WAN environment

- [4] M. Lees, B. Logan, T. Oguara, and G. Theodoropoulos. Simulating agent-based systems with HLA: The case of SIM_AGENT – part II. In *Proceedings of the 2003 European Simulation Interoperability Workshop*. European Office of Aerospace R&D, Simulation Interoperability Standards Organisation and Society for Computer Simulation International, June 2003.
- [5] R. Minson and G. Theodoropoulos. Distributing repast agent-based simulations with hla. In *European Simulation Interoperability Workshop 2004*. Paper 04E-SIW-046, June 2004.
- [6] H. F. Nielsen, E. Christensen, and J. Farrell. Specification: Ws-attachments. <http://www-106.ibm.com/developerworks/webservices/library/ws-attach.html>, June 2002.
- [7] M. E. Pollack, D. Joslin, A. Nunes, S. Ur, and E. Ephrati. Experimental investigation of an agent commitment strategy. Technical Report TR 94- 31, University of Pittsburgh, Pittsburgh, PA 15260, 1994.
- [8] M. E. Pollack and M. Ringuette. Introducing the tileworld: Experimentally evaluating agent architectures. In *National Conference on Artificial Intelligence*, pages 183–189, 1990.
- [9] J. M. Pullen, R. Brunton, D. Brutzman, D. Drake, M. Hieb, K. Morse, and A. Tolk. Using web services to integrate heterogeneous simulations in a grid environment. In *International Conference on Computational Science (ICCS)*, 2004.
- [10] J. M. Pullen, R. Brunton, D. Brutzman, D. Drake, M. Hieb, K. Morse, and A. Tolk. Using web services to integrate heterogeneous simulations in a grid environment. *Future Generation Computer Systems*, 21:97–106, 2005.
- [11] R. Tobias and C. Hofmann. Evaluation of free java libraries for social-scientific agent-based simulation. *Journal of Artificial Societies and Social Simulation*, 7(1), Jan. 2004.
- [12] Y. Xie, Y. M. Teo, W. Cai, and S. J. Turner. Service provisioning for HLA-based distributed simulation on the grid. In *Proceedings of the Nineteenth ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation (PADS 2005) (formerly called Parallel and Distributed Simulation)*, Monterey, CA, USA, June 2005.