Proceedings of the Operational Research Society Simulation Workshop 2016 (SW16)
A.Anagnostou, K. Hoad, M. Kunc eds.

WS-PGRADE WORKFLOWS FOR CLOUD-BASED DISTRIBUTED SIMULATION

Nauman R. Chaudhry, Athar Nouman, Anastasia Anagnostou, Simon J. E. Taylor

Department of Computer Science, Brunel University London
Kingston Lane, Uxbridge, Middlesex, United Kingdom UB8 3PH
nauman.chaudhry@brunel.ac.uk

ABSTRACT

Modeling and Simulation (M&S) is used for systems analysis and decision making in existing or new systems. Modeling large and complex organizations produces large-scale simulations that are difficult or impossible to run on a single computer. Such experiment execution requires high computation. Distributed Simulation (DS) allows modeling of large systems as smaller submodels that execute on different nodes of a computer network and interoperate with each other in order to compose larger systems. Furthermore, cloud computing offers on-demand access to multiple compute resources. Consequently, being able to run DS on cloud resources allows for more experimentation with large-scale simulations in a cost-effective way. However, deploying DS and in fact Cloud-based DS presents significant technical challenges. This paper proposes a framework for deploying DS on the cloud in a transparent manner using the CloudSME Simulation Platform based on WS-PGRADE workflows. A healthcare case study is used to demonstrate our approach.

Keywords: Distributed Simulation, High Level Architecture, cloud computing, MSaaS

1 INTRODUCTION

Modeling and Simulation (M&S) techniques are used to study changes in existing or new systems behaviour to help in decision making, where it is too expensive, impractical or even impossible to implement in the real world. M&S is now widely recognised in the areas of defense, computer & communication (networks), transport (traffic control system), healthcare, behavioural sciences, ecology & environment, biosciences, manufacturing & production, services (e.g., Banks) and economy (Taylor et al 2012). M&S not only requires skilled domain expertise and programming skills, but also high computational resources to simulate large and complex models. Usually, there is memory resource limitation for a single execution unit. Furthermore, the simulation execution run time could be substantial. Therefore, large-scale models could be distributed into smaller models running on several processors, which could remarkably reduce the execution time (Fujimoto, 2000). Distributing a simulation model however presents some technical challenges both in technical skills and the availability of distributed computing infrastructures. One of the technical challenges is to develop interoperable independent models running on different networked computers, while the distributed computing infrastructure has to provide a configured environment of networked computers (in some cases geographically dispersed). Having Distributed Simulation (DS) models running as a service on cloud resources significantly reduces the latter constraint and also increases reusability. The cloud computing paradigm attracts increasing numbers of M&S practitioners wishing to perform their simulations on the cloud. Developing solutions for cloud computing is also difficult without expertise due to complex technologies. The CloudSME Simulation Platform (CSSP), based on CloudBroker and WS-PGRADE/gUSE, was developed to support Modeling

& Simulation as a service (MSaaS) in manufacturing and engineering and simplify the deployment of M&S software and applications on various cloud resources (Taylor, 2014).

In this paper, we propose a framework that utilizes the CSSP to run an HLA- based distributed simulation as a service and investigate how we can automatically configure and use a DS on cloud resources. The paper first introduces the distributed simulation and cloud computing concepts. It then presents the CloudSME Simulation Platform followed by the proposed framework for cloud-based distributed simulation using WS-PGRADE workflows. A case study is then presented to show how HLA-based distributed simulation can be easily run on cloud resources i.e., the academic cloud provided by the University of Westminster (UoW).

2 DISTRIBUTED SIMULATION ON THE CLOUD

Distributed Simulation can be defined as "the distribution of the execution of a simulation program across multiple processors" (Fujimoto, 2000). DS techniques, therefore, make it possible for a single model to be divided and simulated over several processors or multiple models running in different processors to be joined together.

In DS systems, the participating simulation models are able to interoperate with each other. The simulation models can send information to and receive information from other simulations and be able to operate effectively together, i.e., sending the right information to the right destination and at the right time, also without adding prohibitive communication time overhead. Hence, data and time synchronisation is essential in DS systems.

The IEEE 1516 High Level Architecture (HLA) (IEEE, 2010) standard for DS is used to achieve interoperability and reusability between simulation models. The standard was originally developed by the US Department of Defense (DoD). IEEE 1516 HLA specifies a set of standard rules and processes to support DS.

For the DS interface and the implementation of the HLA standard, there are several Run-Time Infrastructure (RTI) implementations presently available, some are commercial RTIs, such as Pitch pRTI, MAK High Performance RTI, RTI NG Pro etc., and others are open source RTIs, such as Open HLA, CERTI, poRTIco etc. Figure 1 illustrates the HLA IEEE 1516-2010 standard, where each federate represents a simulation model and all the participating federates joined together represent a federation, communicating through the RTI.

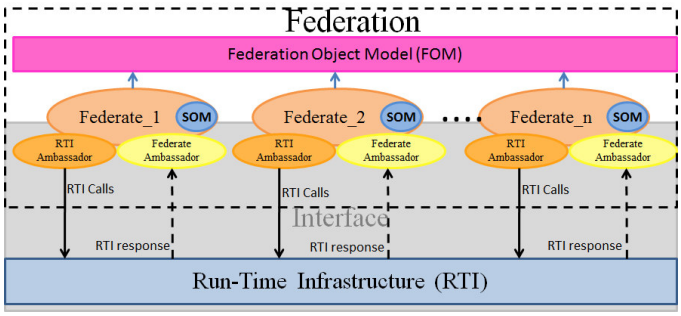

Figure 1 *High Level Architecture*

DS is typically used to model large and/or complex systems which require a significant amount of computing resources. It however requires domain specialist to not only model the system, but also in some cases manage the distributed computing infrastructure. Having access to required distributed computing infrastructure can sometimes be challenging. Therefore running DS as a Service on cloud eliminates the simulation modelers' challenges for maintaining and accessing the infrastructure.

Computing power as a utility was introduced in the 1960s (Hill et al 2013). "Cloud computing" refers to accessing internet-based computing resources and as a term first appeared in the mid-2000s. The National Institute of Standards and Technology (NIST) has made efforts to standardize the terminology of

cloud computing (NIST, 2013). Mell and Grance (2011) and NIST (2013) define cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". Cloud computing provides flexibility to the users for its computational resources, applications, access, etc. that could be tailored according to the user needs. Cloud services can also be deployed according to the security requirements of the organizational structure. Cloud computing offers three defined service models, i.e., Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS).

Cloud computing can support parallel and distributed simulation as a service by providing the required high performance computing infrastructure and its maintenance. Along with the benefits of cloud computing there also some concerns such as security of data, reliability of services and slower execution of code than code execution on cluster node (Fujimoto et al 2010). There are also difficulties for DS used with optimistic synchronization protocols. An approach was proposed to accelerate the execution speed of federates at comparable speed (Li et al 2013). A PaaS RTI-supporting architecture was proposed by combining RTI and web services to run on central servers, so as to provide DS platform to simulation users on WAN (Feng et al 2010; Zhang et al 2012). The latest work is more focused on the associated problems and usage of public clouds (Vanmechelen et al 2013; Yoginath et al 2013). A multi-agent system approach was proposed for model partitioning between the execution nodes on the cloud (D'Angelo et al 2014)

In order to support reusability and interoperability specific cloud services need to be developed to support M&S. M&S as a Service (MSaaS) is considered as a separate cloud service model as it allows users standardized access to build their own simulation models by specialized configured software to run simulation experiments (Taylor et al 2014).

3 CLOUDSME SIMULATION PLATFORM

The CloudSME Simulation Platform was developed by the Cloud-based Simulation platform for Manufacturing and Engineering project (www.cloudsme.eu) funded by the European Commission. The main aim of the project was to develop cloud-based simulation services and platforms that enable Small and Medium Enterprises (SMEs), mainly in manufacturing and engineering, to access simulation software and services, and speed-up simulation experimentation by using cloud resources and other Distributed Computing Infrastructures (DCIs) such as grids, HPC clusters, etc. CloudSME supports MSaaS by combining PaaS and SaaS and allowing simulation software and services providers to build SaaS solutions for end-users and ultimately deliver MSaaS.

The simulation software accessing mechanism should hide entirely potential heterogeneity and complexity of cloud platform, as well as permit the usage of various clouds that may use different cloud middleware. CSSP achieves this by the combination of WS-PGRADE/gUSE workflow deployment and development services (provided by MTA SZTAKI, HU) (Kacsuk et al 2012) and Cloud Broking services (provided by CloudBroker, CH). In other words, it acts as PaaS that supports simulation services deployment, as well as access, in user-transparent way, to various cloud and HPC resources.

CloudSME has developed MSaaS solutions for two types of end-users. That is, a) simulation software providers and b) end-users who use simulation for their business processes improvement or as part of their business offering. Currently, eight simulation software providers are involved in the project including 2MORO (FR), INGECON (ES), ASCOMP (CH), SIMUL8 (UK), SIMSOFT (TR), CMCL (UK), DHCAE (DE), and OUTLANDISH (UK) that offer different simulation tools. Moreover, different end-users are includes CUTTING TOOLS (UK), EUROBIOS (FR), PODOACTIVA (ES), SAKER SOLUTIONS (UK), PROYFE (ES), HOBSONS (UK), BASEPRO (IT), G-VOLUTION (UK), PROCENG (CH), TIDYBOOKS (UK), OZDEKAN (TR), GOKDOGAN (TR), and IOR (IT). Furthermore, the Repast Simphony open source simulation software was deployed as MSaaS in CSSP (Taylor et al 2014). The latter, Repast MSaaS, is used in this study for the implementation of the case

study. CloudSME is led by the Centre for Parallel Computing at the University of Westminster (UK). The cloud-based products development and end-user adoption is managed by Brunel University (UK). UNIZAR (ES), University of Westminster (UK), SZTAKI (HU), and CloudSigma (CH) provide cloud resources.

Figure 2 illustrates the CSSP components organized into three layers reflecting the cloud computing stack. These layers include the Simulation Application Layer (SaaS), the Cloud Platform Layer (PaaS), and the Cloud Resource Layer (IaaS). Moreover, the Cloud Platform layer consists of two parts, i.e., WS-PGRADE/gUSE and CloudBroker Platform (CBP).

CBP provides adapters for the available cloud resources that enable the creation, running, and managing of simulation software images on cloud virtual instances. The simulation software in order to be offered as MSaaS must have been deployed on the respective cloud resources. The deployment process varies and depends on the operating system. Generally, a simulation software executable and runtime scripts are stored in a repository and are called for instantiation and job configuration when requested. The MSaaS Application Patterns and Deployment configurations are formed by these scripts and executables. The CBP services can be accessed directly using its API or via the WS-PGRADE web-based workflow management system. WS-PGRADE can be access using two different APIs, described in detail in Taylor et al (2014).

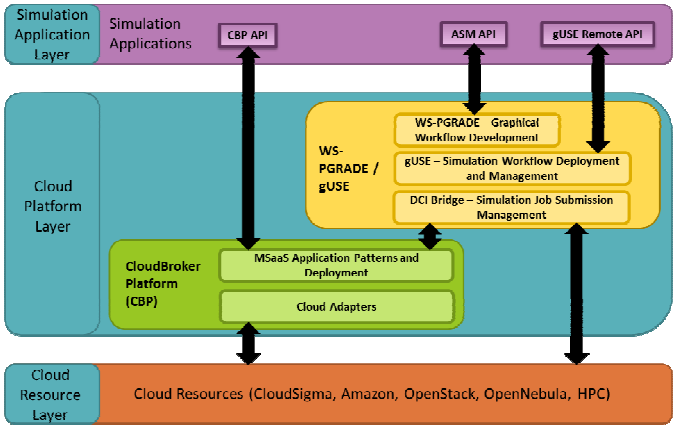

Figure 2 *CloudSME Simulation Platform*

WS-PGRADE is used for workflow creation. Workflows are directed acyclic graphs that can be configured and stored on a workflow server (gUSE). They consist of nodes, job functionality, and arcs, channels for file transfer among the nodes. A node may have one or more input and output ports (green and grey squares, respectively) and each port is denoted by an integer number. A workflow determines the sequence and dependences of a simulation run. An example can be seen in Figure 3. The specific workflow consists of three nodes that perform different jobs. *Initialize Node* has one output port (port 0) linked with the input port (port0) of the next node *Execution Node*. *Execution Node* will start performing its assigned task only after receiving the required input from *Initialize Node*. Once this task completes execution, it will generate an output file which will pass through its output port (port1) to the next node. *Results Node* receives the *Execution Node*'s output through its input port (port0) and then starts executing its task.

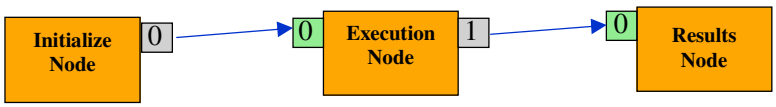

Figure 3 *WS-PGRADE generic workflow*

For flexibility. workflows are "agnostic" and therefore has the functionality to run on any DCI as long as the interface is supported by the workflow manager via the DCI Bridge. CBP is seen as a separate DCI.

4 CLOUD COMPUTING FRAMEWORK FOR DISTRIBUTED SIMULATION

This section explains the proposed framework for DS execution on the cloud. As mentioned before, it is important to access cloud resources transparently and therefore hide the underlying complexity from the end-user. To do so, the proposed framework enables the execution of DS on the cloud in a simplified way by using workflows in the CSSP. The main requirement for the implementation of the framework is the cloud deployment of the RTI and the simulation software as well as their dependencies. In our case, these are the poRTIco RTI implementation of IEEE 1516-2010 and the Repast Simphony Toolkit. Both are java-based software and therefore Java runtime must be installed too. Figure 4 illustrates the workflow in order to execute the DS which consists of four nodes plus as many model nodes as the number of federates in the federation. For example, "Initialize", "Manager", "Model-1", "Model-2", … "Model-n", "Execution" and "Collect Results" are all job nodes. Each job node has its own input and output ports. "Initialize" node contains one input port (port 0) and one output port (port 1). This job takes a text file in input port (port 0), containing the list of federates names to be part of the DS federation, as an input and pass it to "Manager" node through output port (port 1). "Manager" node has three inputs (port 0, port 1 and port 2) and one output port (port 3). This node represents the Manager federate that ensures that all federates has joined the federation when starting the execution of the DS. "Manager" node prepares the Manager Federate executable and pass it to the "Execution" job node. "Model-1", "Model-2", … "Model-n" job nodes contain two inputs (port 0 and port 1) and an output port each. Users provide the model files (model.tar file) and required data (input.tar file) in the input ports 0 and 1, respectively. "Execution" job node takes all the required files on input ports (port 0, port 1 etc.) in order to run the DS. Finally, "Collect Results" job node collects the final simulation results for further analysis and decision making. This framework also supports scalability. More federates can be added by adding Model job nodes in the workflow.

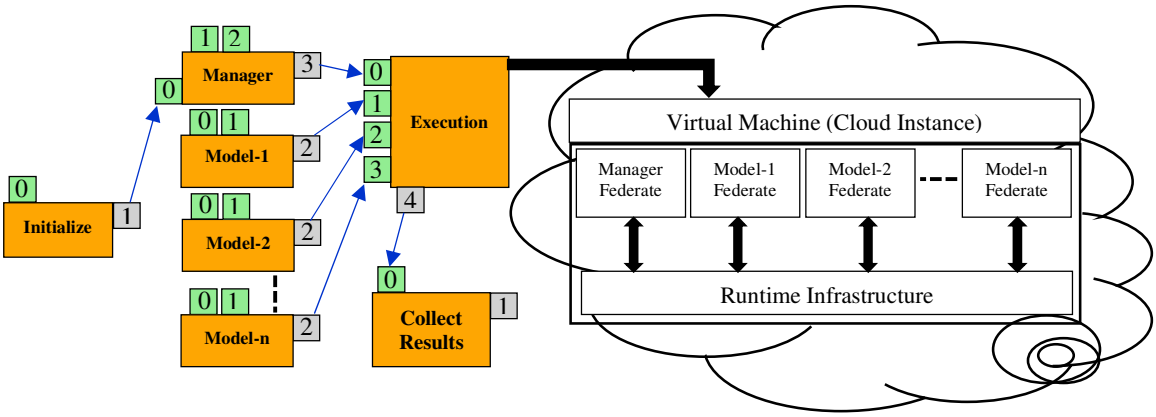

Figure 4 *Framework for Cloud-based Distributed Simulation*

5 CASE STUDY

To demonstrate the applicability of the proposed framework, we developed a simple federation of an Emergency Medical Services (EMS) model (Anagnostou et al 2013). EMS consists of two main organizations, the Ambulance Services and the Emergency Departments in the area of coverage. Each organization is modeled as an independent federate using the appropriate simulation technique. For example, the Ambulance Services model includes objects that should be able to interact with each other and the environment, namely the call operator should be able to interact with the ambulance crew and allocate an ambulance to an emergency call, the ambulance crew should be able to find the appropriate

hospital and select the best route. Therefore, for the Ambulance Service model, Agent-Based Simulation (ABS) was utilized. Emergency Departments usually are process-oriented, where the entities, in this case, patients do not interact and are not aware of the environment; they are just driven by the process through different activities. Hence, Discrete Event Simulation (DES) was selected for modeling the Emergency Departments. For demonstration purposes, the EMS federation consists of two federates; one Ambulance Service and one Emergency Department. The federates interoperate via RTI implementation that coordinates the data exchange and synchronizes the time in order to maintain causality. Emergency Departments have both ambulance and walk-in arrival points. Ambulance patients are received from the Ambulance Services federate while walk-in patients arrive locally in the Emergency Department federate. Therefore a patient agent from the Ambulance ABS federate is transferred with all its attributes to the Emergency Department DES federate and becomes an entity. Patient arrivals in a hospital affect the availability of resources. Emergency Department resource availability is updated locally in the DES federate. Emergency Department availability is communicated with the ABS federate via the RTI. The Ambulance Services federate uses this information in order to find the appropriate hospital for a patient transfer and therefore it is essential to have the most up-to-date value. The whole federation is developed with open source tools. The federates are developed in the Repast Simphony Toolkit (repast.sourceforge.net) and the RTI in the poRTIco IEEE 1516-2010 implementation (www.porticoproject.org).

5.1 Cloud Deployment

To enable the execution of distributed simulation on the cloud, we need to deploy the RTI implementation on the cloud. poRTIco, an open source, fully supported, cross-platform RTI which implements the HLA standard IEEE 1516-2010 is deployed on UoW public academic cloud (OpenStack – provided by University of Westminster, UK) supported by the CSSP. These cloud resources support only Linux applications. In order to be able to run the DS on the available cloud resource, we developed a Shell Script executable for poRTIco deployment. The other requirement was to have the relevant simulation package, i.e., Repast symphony, as well as their dependencies, namely Java runtime installation.

5.2 Workflow Creation

The workflow creation is supported by the WS-PGRADE. The first step is to create the workflow using the graph editor, a Java Network Launch Protocol (JNLP) application, which provides drag-and-drop user-friendly environment, where the structure of the workflow can be created. Figure 5 illustrates the graph editor, where the bigger squares denote jobs and the attached smaller squares denotes input and output ports (green and grey squares, respectively). Each port will be denoted by a number. Each job can have more than one input and output. Initialize, Manager, Ambulance, Hospital, Execution and Results are job nodes. Each job node has its own input and output ports. Initialize node will receive the federatelist.txt file, containing the name of federates to join the federation, on the input port (port 0). This file will be forwarded through output port (port 1) of Initialize node to Manager node on its input port (port 0). Manager, Ambulance and Hospital nodes will receive the model files (model.tar) and its parameters (batch_params.xml) for each model/federate. Ambulance node also requires data files (input.tar) and these are received through the input port (port 0). Manager, Ambulance and Hospital nodes will generate the output, which are then fed to the Execution node in its input ports 0, 1 and 2 respectively. Execution node will run the federation and generate the results file which is then transferred to Results node for further analysis.

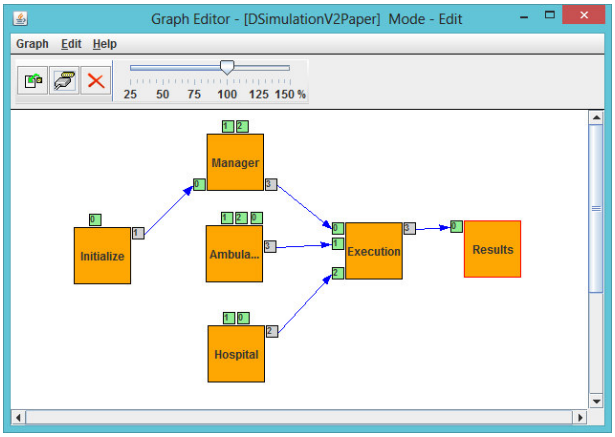

Figure 5 *WS-PGRADE graph editor*

5.3 Workflow Configuration

The graphical workflow is configured at the WS-PGRADE web portal (see Figure 6). In each of the job nodes, the first step is to select the type of DCI. As mentioned earlier, CSSP's cloud resources are managed by CB. CB account authentication is needed in order to be able to access the portal. Then the software or own executable that will execute in the workflow and the executable (shell script file) are selected. From the available cloud resources, once a selection of the instance type is made, an estimation of the job cost is displayed. Initialize.sh executable will be selected for Initialize node which will confirm that the federatelist.txt is received. Manager.sh will be selected for Manager node. Model.sh will be selected for Ambulance and Hospital modes. Finally the software that will execute the DS in the Execution node (in this case Repast_multi 1.0) and the executable (shell script file) will be selected. The next step is to configure the input and output ports. Initialize node contains one input and one output port. This job takes the text file, containing the list of federates names to be part of the federation (distributed simulation), in the input port and pass this file that is required for Manager job node through the output port. Manager node has three inputs and one output ports. This node represents the Manager federate. It will be ensured that all federates have joined the federation during the DS execution. Manager node prepares the Manager Federate executable and passes to the Execution node. Ambulance and Hospital nodes contain three and two inputs respectively and one output port. Users provide the required model.tar, batch_params.xml and input.tar on the input port 0, 1, and 2 respectively. As explained earlier, model.tar contains the simulation model source code, batch_params.xml contains the input parameters and input.tar contains the data files which required for the execution of the model. Execution job node takes all the required files in the respective input ports in order to run the DS and will export the resulting files from the job run in the output port. The output files can be downloaded after the completion of the job.

5.1 Workflow Execution

Execution job node contains three inputs, i.e. Manager, Ambulance and Hospital tar files and produce one output after execution. In this experiment, we run the distributed simulation on the single instance (see Figure 7). OpenStack Nova University of Westminster UoW has the following hardware specification for instance:

- Small (CPU: 1, Cores: 1, Memory: 2 GB)
- Medium (CPU: 1, Cores: 2, Memory: 4 GB)
- Large (CPU: 1, Cores: 4, Memory: 8 GB)
- Extra Large (CPU:1, Cores:8, Memory: 16 GB)

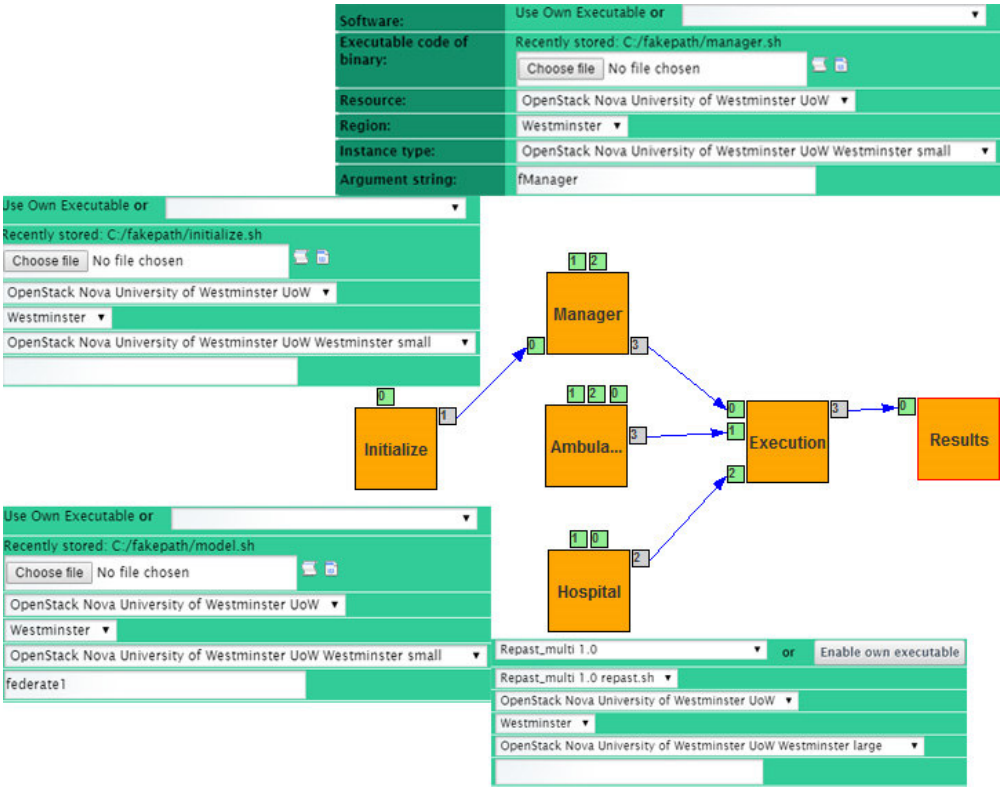

Figure 6 *Workflow Configuration*

We can run distributed simulation on any of above hardware specification instances by specifying the instance type at the time of job configuration. Multiple CPU cores will improve the performance. In this experiment, distributed simulation federation has three federates. Each federate runs on a different core. So, we will select the instance type which has three or more cores. In our case, we run this simulation on Large instance.

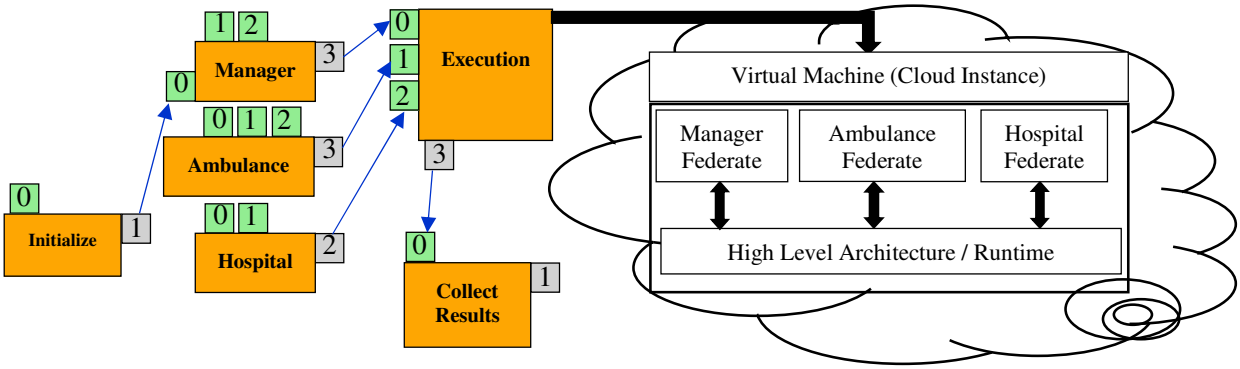

Figure 7 *Workflow execution on the cloud*

5.2 Experiment Results

The speed of the simulation execution depends on the number of federates and simulation time period. In our example, we had three federates running on three different cores, a manager federate and two Emergency Medical Services (EMS) federates, namely Ambulance federate and Hospital federate. A

smaller 24 hour simulation time was selected for our experiment. It was observed that in case of small jobs the preparation time is much higher than the actual simulation execution time. But as the simulation gets bigger or the simulation time is increased then the preparation time will be insignificant. Figure 8 presents a breakdown time of Execution node and illustrates the execution time from the job submission stage till the final stage. It was noted that in our simulation run one third of the total execution time was for the simulation run while the rest was for the preparation and finalising of cloud instance. However, as the simulation time or the complexity of the simulation will increase, the running time percentage will also increase. Although the assembling time and preparation time depends on the availability of the cloud service, but in an ideal situation it will remain approximately similar.

Status	Duration	Percent
submitted	1 second	0%
assembling	2 minutes 41 seconds	46%
preparing	39 seconds	11%
starting	22 seconds	6%
running	1 minute 49 seconds	31%
finishing	12 seconds	3%

Figure 8 *Workflow execution on the cloud*

6 CONCLUSIONS

This paper has presented a framework for running HLA-based distributed simulation using WS-PGRADE workflow on the CloudSME Simulation Platform. By doing so, it hides the deployment complexity of cloud-based distributed simulation from users. Using this framework, we are able to execute varied size federations on the cloud. This framework offers great opportunities for the distributed simulation community to make the technology widely accessible for runnung larger scale distributed simulations. Future work involves performance testing of the framework on different clouds and cloud-based clusters.

ACKNOWLEDGMENTS

Special thanks to Dr. Tamas Kiss and Mr Hannu Visti, Centre for Parallel Computing, University of Westminster for supporting in this study.

REFERENCES

Anagnostou A, Nouman A and Taylor S J E (2013). Distributed Hybrid Agent-Based Discrete Event Emergency Medical Services Simulation. In: Pasupathy R, Kim S H, Tolk A, Hill R and Kuhl M E (eds). *Proceedings of the 45th Winter Simulation Conference.* Washington, DC, USA, pp 1625-1636.

D'Angelo G, and Marzolla M (2014). New trends in parallel and distributed simulation: From many-cores to Cloud Computing. *Simulation Modelling Practice and Theory*. Vol 49: 320-335.

Feng S, Di Y, Yuanchang and Meng Z X (2010). Remodeling traditional RTI software to be with PaaS architecture. *3rd IEEE International Conference, Computer Science and Information Technology (ICCSIT)*. Chengdu, China, pp. 511–515.

Fujimoto R M (2000). *Parallel and Distributed Simulation Systems*. New York: Wiley Interscience, Inc.

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

Hill R, Hirsch L, Lake P and Moshiri S (2013). *Guide to Cloud Computing: Principles and Practice*. Springer-Verlag: London, UK.

IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Framework and Rules, IEEE Standard 1516, 2010.

Kacsuk P, Farkas Z, Kozlovszky M, Hermann G, Balasko A, Karoczkai K and Marton I (2012). WSPGRADE/gUSE generic DCI gateway framework for a large variety of user communities. Journal of Grid Computing Vol 10 Issue 4: 601-630.

Li Z, Li X, Duong T N B, Cai W and Turner S J (2013). Accelerating optimistic HLA-based simulations in virtual execution environments. *Proceedings of the 2013 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*. New York, NY, USA, pp 211–220.

Liu X, Qiu X, Chen B and Huang K (2012) Cloud-Based Simulation: The State-of-the-Art Computer Simulation Paradigm. *Proceedings of the 26th Workshop on Principles of Advanced and Distributed Simulation*. Washington, DC, USA, pp 71–74.

Malik A W, Park A and Fujimoto RM (2010). An Optimistic Parallel Simulation Protocol for Cloud Computing Environments. *SCS M&S Magazine* I(4).

Mell P and Grance T (2011). The NIST Definition of Cloud Computing. Accessed August 07, 2015. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

NIST (2013). NIST Cloud Computing Standards Roadmap. Accessed August 07, 2015. http://www.nist.gov/itl/cloud/upload/NIST_SP-500-291_Version-2_2013_June18_FINAL.pdf.

poRTIco Project. HLA RTI implementation. Accessed August 07, 2015. http://www.porticoproject.org/.

Taylor S J E, Anagnostou A, Kiss T, Terstyanszky G, Kacsuk P and Fantini N (2014). A Tutorial on Cloud Computing for Agent-Based Modeling & Simulation with Repast. In: Tolk A, Diallo S D, Ryzhov I O, Yilmaz L, Buckley S and Miller J A (eds). *Proceedings of the 46th Winter Simulation Conference*. Savanah, GA, USA, pp 192 – 206.

Taylor S J E, Kiss T, Terstyanszky G, Kacsuk P and Fantini N (2014). Cloud Computing for Simulation in Manufacturing and Engineering: Introducing the CloudSME Simulation Platform. *Proceedings of the 2014 Annual Simulation Symposium*. San Diego, CA, USA, Article 12.

Taylor S J E, Fishwick P A, Fujimoto R and Uhrmacher A M (2012). Panel on Grand Challenges for Modeling and Simulation. In: Laroque C, Himmelspach J, Pasupathy R, Rose O and Uhrmacher A M (eds). *Proceedings of the 44th Winter Simulation Conference*. Berlin, Germany, pp 1 – 15.

Vanmechelen K, Munck S D and Broeckhove J (2013). Conservative distributed discrete-event simulation on the Amazon EC2 Cloud: an evaluation of time synchronization protocol performance and cost efficiency. *Simulation Modelling Practice and Theory*. Vol 34: 126-143.

Yoginath S B and Perumalla K S (2013). Empirical evaluation of conservative and optimistic discrete event execution on cloud and VM platforms. *Proceedings of the 2013 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*. New York, NY, USA, pp 201–210.

Zhang H, Liu C and Bi X (2012). PaaS RTI-Supporting Distributed Simulation Interaction in Cloud Computing Age. *Advanced Engineering Forum* Vols 6-7: 887-894.

AUTHOR BIOGRAPHIES

NAUMAN RIAZ CHAUDHRY is a PhD candidate at Department of Computer Science, Brunel University, UK. His email address is nauman.chaudhry@brunel.ac.uk.

ATHAR NOUMAN is a PhD candidate at Department of Computer Science, Brunel University, UK. His email address is athar.nouman@brunel.ac.uk.

ANASTASIA ANAGNOSTOU is a Research Fellow at the Department of Computer Science, Brunel University London. Her email address is anastasia.anagnostou@brunel.ac.uk.

SIMON J. E. TAYLOR is the Founder and Chair of the COTS Simulation Package Interoperability Standards Group under SISO. He is the Editor-in-Chief of the UK Operational Research Society's (ORS) Journal of Simulation and the Simulation Workshop Series. He is Reader in the Department of Computer Science at Brunel University and leads the M&S Group. His email address is simon.taylor@brunel.ac.uk.