DSAAS: A PLATFORM FOR DISTRIBUTED SIMULATION AS A SERVICE

Hassan Rajaei
Fatimah Alotaibi
Rawan Alturkistani
Mashael Alzaidi
Saba Jamalian
Department of Computer Science
Bowling Green State University
Bowling Green, OH, 43403-0085
{rajaei, falotai, ralturk, malzaid,sabaj}@bgsu.edu

ABSTRACT

Advances of cloud services gave Cloud-Based Simulation (CBS) considerable attentions to obtain reliable, available, and scalable simulation models capable of executing on the Cloud. However, major challenges such as synchronization, virtualization, and multi-tenancy overhead overshadows use of CBS especially for distributed simulation. These challenges affect performance of parallel executions and thus limit the offered benefits. To overcome these issues, we propose a new platform called DSaaS: Distributed Simulation as a Service, to facilitate implementation and parallel execution of simulations models in the cloud. This platform has several layers including a middleware and an HPC platform called HPCaaS (HPC as a Service). The middleware acts as a simulation engine facilitating users to develop simulation models whereas HPCaaS makes them executable on a cloud platform. Along with their associate API, the proposed architecture enables simulation users to seamlessly interact with a cloud environment and receive benefits of the cloud for their applications.

Keywords: High Performance Computing (HPC), Cloud Computing, Cloud-Based Simulation (CBS), Distributed Simulation, Middleware.

1 INTRODUCTION

Rapid growth of cloud services led exploding migration of services and applications to the Cloud. As stated by Foster et al. (2008) “Cloud computing is hinting at a future in which we won’t compute on local computers, but on centralized facilities operated by third-party compute and storage utilities”. This paradigm shift suggests widespread use of simulation on the cloud and encourages developers to deploy their simulation models on cloud. Time saving, scalability, availability, flexibility, and cost saving are among major benefits for applications migrating to the Cloud. Nevertheless, there are challenges that hinder full adaption of cloud services for simulation. Taylor et al. (2012) identifies one major challenge as Cloud-Based Simulation (CBS). The key issues of cloud-based simulation have been demonstrated in several publications which include synchronization overhead, virtualization overhead, multi-tenancy, and network latency which causes performance degradation particularly for Parallel and Distributed Simulation (PADS) (Padilla et al. 2014; Taylor et al. 2014; Taylor et al. 2015). As a result, cloud deployment for PADS application either is none existing, or very limited to tailor-made special cases.

To overcome the above problem, we designed DSaaS: Distributed Simulation as a Service, which is a new cloud service for simulation especially targeting large simulations that need parallel executions of the simulation modules. DSaaS facilitate implementation and execution of PADS applications in a cloud environment. This new service has several layers, and is implemented on the PaaS (Platform as a Service) layer of the Cloud architecture (Mell and Grance 2011).

At the lower level of DSaaS, a specialized layer for HPC applications is designed and developed (Jackson et al. 2010; Gupta et al. 2013). This layer provides services for HPC applications and is called HPCaaS (High Performance Computing as a Service) (AbdelBaky et al. 2012). Having this service platform integrated in our solution, alleviates the CBS users from jitters a cloud environment may introduce into HPC applications. Section 3 of this paper describes the details of this layer and the jitters impact on performance.

At the second layer of the DSaaS, we developed a middleware, called CBS_MID which acts as a simulation engine targeting a smooth simulation run on a cloud computing platform. This middleware provides services for the PADS applications including conservative algorithm (Fujimoto 2000), optimistic simulations (Fujimoto 1990), and hybrid method such as LTW (Rajaei, et. al 1993). Furthermore, this engine supports simulations using High Level Architecture (HLA) (Dahmann et al 1997), as well as general Cloud-Based Simulation models. The proposed middleware provides numerous simulation services and management modules through their associate API's.

On top layer, an interface connects the middleware to the users and their applications. Using the DSaaS services and its API, generic simulation modules and tools can be built on top layer as plugins for the SaaS layer.

The DSaaS architecture enables the CBS users and their applications to seamlessly interact with a cloud environment, obtain the improved performance, and increase the scalability of their simulation models.

The rest of this paper is as follows: Section 2 presents related works. Section 3 describes the details of HPC as a Service. Section 4 presents Cloud-Based Simulation, whereas Section 5 describes the details of DS as a Service. Section 6 gives future works and concluding remarks.

2 RELATED WORK

Three main areas of related work is covered in this section. First, previous works regarding deployment of PADS applications to cloud is reviewed. Second, efforts regarding simulation as s service (SIMaaS) is described. Finally, related work regarding middleware for CBS framework is addressed.

Vanmechelen et al (2013) investigated the synchronization overhead of PADS instances on computational cores to examine how different cloud instance may affect the performance of distributed simulation. First, by using Grid Economic simulator (GES) they implemented different conservative synchronization protocols such as Standard Chandy-Misra-Bryant protocol (STD), Timeout-based Null-message Sending (TIM), Deadlock Avoidance Null-Message Sending (BLO), then they assessed and analyzed the performance for each one in two simulation models. They executed two different conservative simulation models on two different instance types of EC2. One model was a closed queuing network (CQN), and the second one an electronic auction for compute resources. In addition, the two models had differences in event arrival rates, communication patterns and computation to communication ratios. The experiment were focused on how the main strategies of conservative simulation performed on a cloud infrastructure. As a result, they found that time-out conservative protocol performed best for CQN, while Deadlock Avoidance Null-Message Sending (BLO) shows better performance for the auction model.

Malik et al. (2010) discussed that moving the optimistic synchronization approach to cloud could exhibit major problem specially on the number of rollbacks. Traditionally, Time Warp method is used to detect out of order timestamped events which are corrected by rollback method. Their experiment showed that massive rollbacks and thus high overhead could be experienced. Further, the problem could become impractical when deploying simulation models especially when having high latency in cloud networking. To overcome the issue, they proposed Time Warp Straggler Message Identification Protocol (TW-SMIP) to illustrate the interference and the latency issues. They reported that TW-SMIP protocol could decrease the communication overhead and thus reduce the number of the rollbacks on the cloud.

Research on simulation as a Service (SIMaaS) has recently gained much interest. Numerous works reported in the literature to deploy simulation in the cloud to gain high availability and accessibility, as well as on-demand resources. Despite these efforts, significant challenges that Cloud-based simulation exhibits, brings forth a crucial question: how can we make SIMaaS practical?

The research reported by Liu et al. (2012) tried to answer the above question and proposed a somewhat complete package. They first moved an existing simulation software to the cloud and studied its behavior. Then, they composed the needed requirements, and prosed SIMaaS architecture. In a form of web service, SIMaaS provides different simulation tools for clients. SIMaaS contains Cloud Manger Module (CMM) that works as a simulation operating system that has different management modules. Additionally, SIMaaS uses a web-portal and simulation service layer. Using the Service-Oriented Architecture (SOA) concept, Several services are provided by SIMaaS controlled by CMM: Modeling as a Service (MaaS), Execution as a Service (EaaS), and Analysis as a Service (AaaS). The virtualized resources of SIMaaS are offered by the Virtualized Computing Environment (VCE) module through its related API. VCE delivers computing management and makes the computing resources programmable. To measure performance loss of a simulation in the virtual environment, they conducted an experiment that shows 33.3% performance degradation compared to the physical machines.

Several surveys identified the needed features in the cloud for distributed simulation. Strassburger et al. (2008); Boer et al. (2009) argued that there is a lack of plug-and-play middleware to help interoperability, reusability, and scalability of applications on the cloud. They pointed out that interoperability should be achieved with reasonable cost. They identified the needed characteristics of a plug-and-play middleware. They emphasized first that a middleware should ensure the entire system must work without any effect in case some component changes. They added that the middleware should maintain components independency and synchronization.

3 HPC AS A SERVICE

Cloud provides utility computing through several services. HPCaaS is the service which supports HPC programs running on the cloud without facing significant performance loss. Primary motivations for these applications to move to the cloud are resource utilization, cost efficiency, flexibility, among many others. Table 1 summarizes a comparison between characteristics of an average on-premises HPC resource such as clusters or supercomputers with the one provided by a public cloud supporting HPCaaS. This table clearly shows the benefits of HPCaaS for average HPC users. For high-end HPC users, perhaps dedicated datacenters may still remain as a viable option.

3.1 Cloud and HPC

Scientific and HPC applications traditionally demand direct access to dedicated hardware and high-speed networking with low latency. Such requirements often do not match with existing virtual and multi-tenant environment of current cloud systems. We conducted numerous experiments to identify key shortcomings of a typical cloud when executing HPC applications. The results suggest that cloud networking, virtualization overhead, and multi-tenancy are among primary sources of turbulences of HPC performance in the Cloud (Gupta et al 2011) which is described in more details below.

Table 1: On-Premises HPC resources vs. HPCaaS.

Characteristic	**On-premises HPC resource**	**HPCaaS**
Scalability	Low	High
Flexibility	Low	High
Reliability	Low	High
Cost	CAPEX	OPEX
Setup Time	High	Low
Maintenance & Administration	expensive & complex	N/A
Resource Utilization	Low	High

3.1.1 Multi-Tenancy

Multi-tenancy is one of the characteristics of the cloud. It is also one of the profit making features of the cloud for service providers. It enables cloud providers to share resources between multiple tenants. Degree of multi-tenancy refers to the number of tenants sharing the same resource on the cloud. By increasing the degree of multi-tenancy, service providers are able to overprovision the resources to users. Overprovisioning allows the service providers to maximize benefits, though with the risk of reducing QoS. Nevertheless, multi-tenancy is in direct contrast with what HPC needs. HPC programs often demand direct access to dedicated hardware to maximize performance while shared resources of the cloud adversely impact performance of these programs.

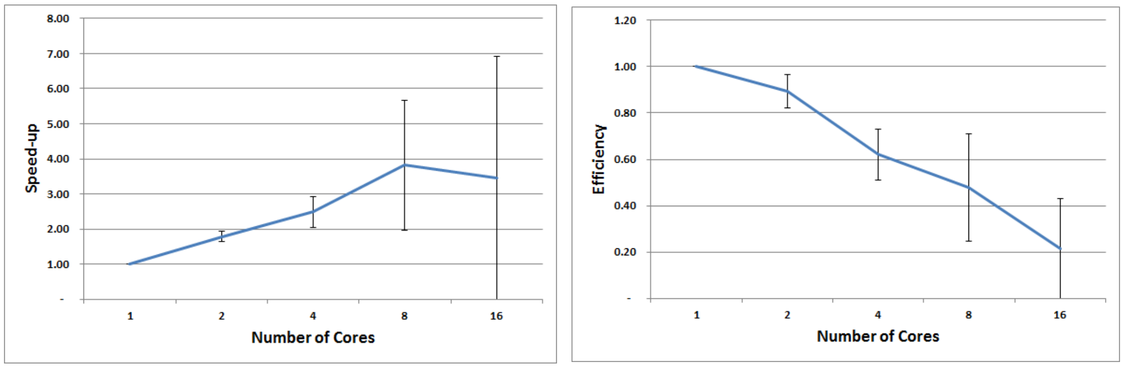

Figure 1: Speed-up (a) and Efficiency (b) of a Matrix Multiplication benchmark on Amazon EC2 instance

We conducted experiments using Matrix Multiplication benchmark on a virtual instance of Amazon EC2 public cloud. We repeated the experiments for 15 times to pin point the jitter effect as shown in Figure 1a. The error bars represent the standard deviation of the results and indicate that by increasing the number of cores, the diversity of values we get in multiple experiments, increases. Figure 1b is the efficiency achieved for the experiments and the error bars are again the standard deviations. These experiments show how performance of HPC programs on the cloud become unpredictable with potentially large fluctuations due to the shared resource and multi-tenant environment. Further, the experiments indicate that there is relatively huge gap between the average and the best performance of Matrix Multiplication benchmark on Amazon EC2. Similar experiments by others (Gupta et al 2013) confirm that multi-tenancy and network latency are major bottleneck degrading performance of HPC programs in the cloud.

3.1.2 Virtualization Overhead

Virtualization plays a key role in helping the cloud to have elasticity, resource pooling, and flexibility. Nevertheless, virtualization and in particular the hypervisor adds unwanted overhead by adding a software layer and preventing applications to have direct access to the hardware resources. This overhead is not the same for all types of hardware. For example, due to special hardware support, virtualization overhead for processors is significantly less than the overhead of network virtualization.

3.1.3 Network Bandwidth and Latency

Network and I/O resources in the cloud are shared between several tenants. Consequently, bandwidth and latency of the network may not be predictable. The bandwidth in most cases is much less than what is expected. We experimented with Point-to-Point MPI benchmarks on a 10Gig Amazon EC2 interconnect. Our results suggest that the multi-tenant environment of the cloud lowers the bandwidth. Moreover, the latency of the network of the cloud could vary. As a result, we often see performance loss of HPC programs on the cloud when not using HPCaaS support.

3.2 ASETS, our Solution

ASETS (A SDN Empowered Task Scheduling System) is an elastic HPCaaS platform developed for HPC application. Information provided by the SDN Controller enables the task scheduler to consider network properties in assigning tasks to virtual machines. We also developed a scheduling algorithm called SETSA (SDN Empowered Task Scheduling Algorithm) to run on top of the ASETS architecture (Jamalian and Rajaei 2015a, 2015b). The algorithm measures the available bandwidth of the network links and schedules tasks on the most appropriate links. This scheduling algorithm enables ASETS to utilize network bandwidths more efficiently and thus it will increase the performance of HPCaaS platform by reducing turnaround time of the submitted jobs.

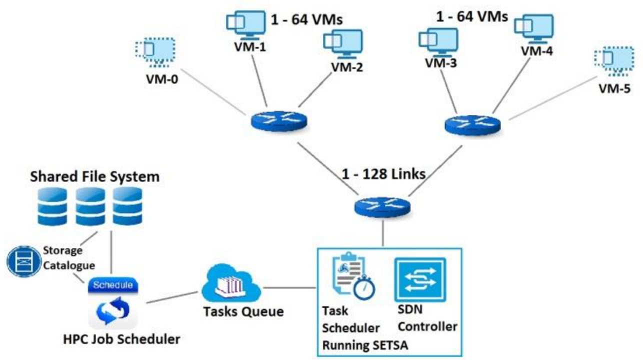

Figure 2: Overview of ASETS architecture.

3.3 Case Studies

Our empirical analysis of ASETS with multiple case studies indicated that ASETS is most beneficial in improving the performance of HPCaaS when the degree of multi-tenancy in the cloud increases (Jamalian and Rajaei 2015a). To measure the performance improvement value of the proposed system we used OpenStack integrated with OpenDaylight (Medved et al 2014) to implement ASETS in order to obtain a proof of the concept. OpenStack as a cloud operating system provides full hypervisor level access to a Cloud Computing environment whereas the OpenDaylight enables SDN to interact with OpenStack. The experiments were carried both in a private cloud as well as a public one. For the private cloud, we implemented ASETS configuration on a cluster of 6 compute nodes and for the public one we used Amazon AWS cloud.

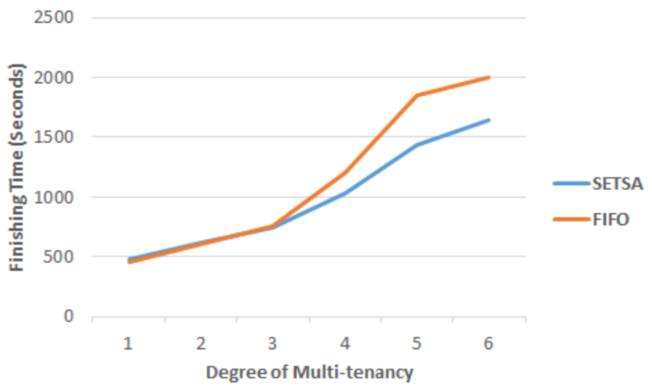

Figure 3. Performance of ASETS compared with FIFO scheduling in a private cloud.

Figure 3 shows performance ASETS in private cloud when (simulated) degree of multi-tenancy increases, whereas Figure 4 shows Assets performance in Amazon EC2 cloud. As Figure 4 illustrates, the SETSA scheduling algorithm of ASETS has unwanted overhead when using smaller scales of Virtual Machines (VM). Nevertheless, with larger number of VMs significant performance improvement is observed for ASETS over traditional FIFO task scheduling. As the HPCaaS platform scales with additional VM, the number of network links and available bandwidths increase, allowing SETSA to have larger variety of choices to redirect tasks data. This experiment was limited to 32 VM. It is speculated that if the number of virtual machines increase (e.g. x100 times or more) with a larger network-intensive task, the fluctuations of the network links correspondingly increase and SETSA will perform much better.

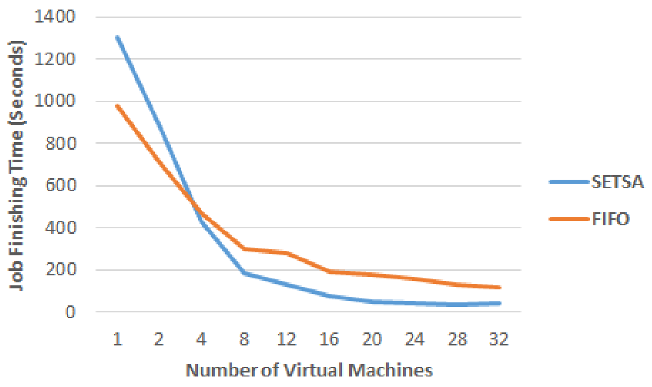

Figure 4. Performance of ASETS compared with FIFO scheduling in Amazon public cloud.

4 CLOUD-BASED SIMULATION (CBS)

Computer simulation helps to study the behavior of a system, including interactions, complexity, and expected outcomes. As the complexity of the target systems increases, predictability of the system behavior becomes vital both for the system designers and the users. To study in depth the behavior, not only we need detailed simulation models, but also computing powers to quickly offer the needed data and tools for analysis and interpretations of the collected data. Cloud services appears as a viable method to deliver simulation services for analysts as well as non-experts users.

4.1 Why Cloud-Based Simulation

It is well-known that simulation is a powerful tool for study and analysis of the behavior of complex systems. It is also well-known that good simulation tools are expensive and often have steep learning curves. Availability of the tool when it is needed, is a third obstacle that makes simulation to remain in the expert domain. Furthermore, scalability of the modules and computational power of the simulation scenarios add other dimensions to the limitations of this powerful method.

To alleviate some of the above issues, we need to look at the promises of cloud computing which promotes pay-as-you-go mothed along with high availably, reliability, and scalability of resources as well as non/or low maintenance cost of the used resources.

Comparing the above services with some of the existing tools, few of which cost tens of thousands of dollars for a particular module, it is safe to say that Simulation-as-a-Service will become a trend. The provided services could include not only a tool to develop models and execute them on the cloud, but also provide easy to understand tutorials, how to build base model, lectures and videos, as well as diverse case studies, powerful analysis tools and presentations, customized services, just to name a few. While most of these benefits could currently exist in multiple specialized tools, the new cloud-based trend can bring much more diversities to simulation users and communities and thus open the doors to wider use of this powerful technique.

4.2 Prospective Types of CBS

Simulation has wide variety of techniques, and thus cloud-based simulation services can potentially cover all of these branches. Nevertheless, in this research we focus only on Discrete Event Simulation (DES) which represents chains of events in the system, where each event can cause other changes at certain event time, and thus the simulation outcome will be given through ordered chains of the generated events (Fujimoto 2000). Furthermore, we focus on Parallel Discrete Event Simulation (PDES) and its three associate branches: Conservative, Optimistic, and Hybrid protocols (Figure 5) in DSaaS and CBS. Needless to say that sketched here, can be extended and be applied to other CBS domains.

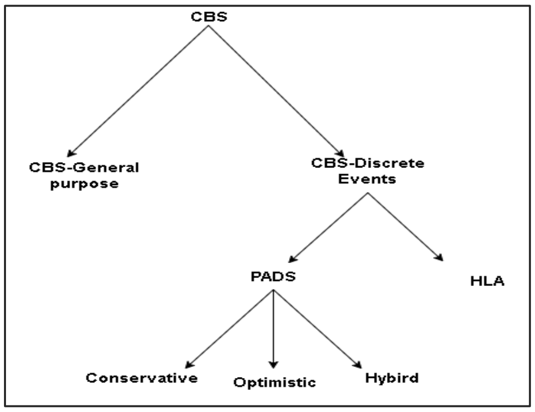

Figure 5: Illustrating types of CBS focusing on branches addressed in this paper

4.3 Distributed Cloud-Based Simulation

Sequential DES (i.e. running on single CPU) albeit can give good estimate of the system understudy, it has multiple limitations. Prime examples include scalability of the model, detailed and modular plugins, and computing power. Parallel and Distributed Simulation (PADS) addresses some of these issues and provides mechanism to distribute the model across multiple nodes, with specific algorithms to synchronize the events (Fujimoto 2000). The simulated system can be executed on broad span of interconnected multiprocessors.

PADS has two main synchronization protocols; conservative and optimistic, but the hybrid of the two is often supported. The system must maintain, however, the same results as its sequential counterpart, while distributing the discrete events among multiple processors. In the conservative protocol, the simulation engine blocks the logical processes when their local simulation clocks could become out of timestamp order, whereas the optimistic method allows those process to proceed but when error is detected, that process rollbacks to correct the error.

The blocking and rolling back of PADS processes are major obstacles in porting them to clouds. Apparently a local data center could better benefit PADS program rather than cloud since the cloud platform would add additional hazards, such as networking delay and others as mentioned earlier, and hence significantly degenerate performance. Clearly, not every user can have access to a dedicated HPC data Center. As a result, the need for a distributed cloud-based simulation becomes clear. As an HPC application domain, benefits of PADS include faster executions and ability to simulate large distributed systems. The HPCaaS platform described earlier bridges the gap and let the PADS program enjoy benefits of the cloud.

Cloud services give an illusion of unlimited resources, and thus serve well to distributed simulation. A distributed cloud-based simulation service can furnish the users all benefits of the cloud while obtaining as well faster execution of the module when the user desires, and pay for only that service. As mentioned earlier, and confirmed by others, e.g. D'Angelo(2011), there are serious hazards to address before we can comfortably move PDES to cloud. A distributed simulation-as-a-service can alleviate the problem.

5 DISTRIBUTED SIMULATION AS A SERVICE (DSAAS)

The three well-known platforms of a cloud according to the NIST definitions (Mell and Grance 2010), include Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). This section describe mapping of the DSaaS layer, its architecture, the Middleware, and the API interfaces to cloud layers.

5.1 Mapping DSaaS Layer to Cloud Architecture

DSaaS is implemented at the PaaS level and on the top of the utility services of the IaaS layer. The platform utilizes an HPCaaS layer (described in Section 3) and its main objective is to make the HPC programs run smoothly on the cloud. This service alleviates the PADS users from the jitters of the clouds and furnishes the tool they need to focus developing their simulation model to be executed in a potentially endless computing resources when they desire. Figure 6 illustrates the mapping view of DSaaS to the cloud architecture.

The middleware works as a simulation engine of DSaaS, and connects the PADS programs to the HPCaaS layer. The engine provides main simulation services in building modules to launch, execute, and interconnect the simulation model in cloud environment. As a simulation engine, the middleware interacts with the simulation programs via its API. It is in charge of retrieving simulation objects, updating the state of the simulation, tracking events for check-pointing, and so on. What is left is example models at the application level (i.e. SaaS) to demonstrate how the DSaaS architecture can be used. This part of system is left to be accomplished as the future work.

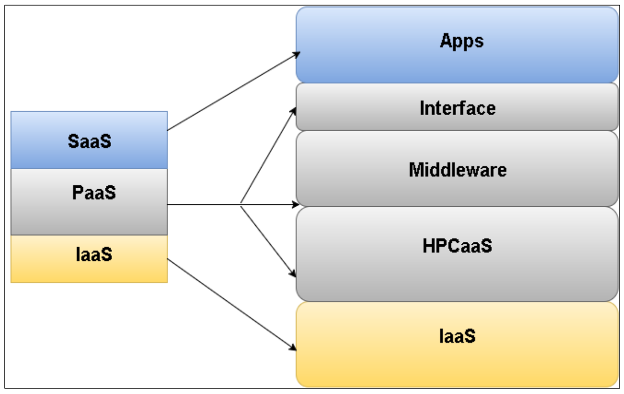

Figure 6: Overview for DSaaS mapped to the three Cloud platform

5.2 Architecture of DSaaS

The architecture of DSaaS is shown in Figure 7 and explained in this section. The top layer has the interface, with its the associated API routines for both conservative and optimistic synchronization methods. The API for the hybrid and other methods is left for future work. The next layer is the middleware (CBS_MID) which is a major layer of the DSaaS platform due to extensive simulation services it provides as well as its interconnections and integration role to the application programs. CBS_MID is the simulation engine and the core of the simulation process that launches and executes the simulation models using the provided management modules. CBS_MID engages the simulation instances seamlessly with the HPCaaS layer (ASETS). The latter layer prepares the PADS programs for smooth execution on the cloud. Using DSaaS and CBS_MID, synchronization of a distributed simulation is managed transparently from the users and frees them from the underlying complex details associated with both parallel processing and cloud hazards.

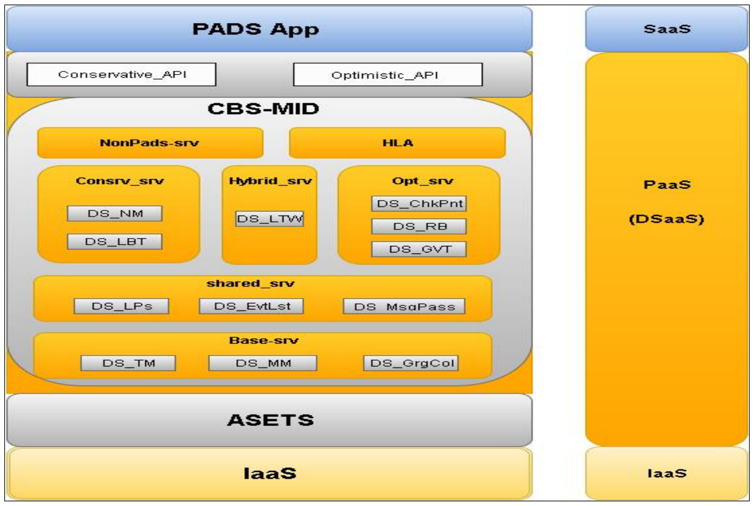

Figure 7: The architecture of DSaaS integrated in a cloud platform

5.3 CBS_MID Services and Management Modules

The middleware services are classified based on the types of protocol the users prefer having them for their execution models. Table 2 summarize these services. As seen in this table, wide variety of DES/PDES can be furnished. Implementation of these services may add minor challenges which a typical simulation engine could encounter and it is not addressed here, rather the architecture of CBS basis is the focus of this paper. The framework presented in this paper, however, can be used for plugins particular module, for example HLA simulation.

Table 2: Services of CBS_MID.

Service Name	Types	Services and Management Details
Base-srv	Base services	**Time management:** Deals with time synchronization of the distributed simulation
		Memory management: Allocate and release memory
		Garbage collection: Releases memory no longer used by simulation objects
Shared-srv	Shared services	**Logical processes services:** Creates & manages LPs.
		Event list services: Queue containing future events
		Message passing: Manages communications between simulation entities
Opt-srv	Optimistic Sim	Roll back, check pointing, and Global Virtual Time. Supports features of the optimistic synchronization
Conservative -srv	Conservative	Null massages, and Lower Bound Time Stamps Supports features of the conservative synchronization
Hybird-srv	Hybrid services	**Local Time Wrap (LTW)**: a hybrid methods of optimistic and conservative approach
HLA-srv	HLA	Supports High Level Architecture simulations
NonPads-srv	Non-PADS	Supports sequential simulations

The services supported by the middleware are called by the Application Program Interface (API), developed for conservative and optimistic protocols. These APIs hide the synchronization details and makes developing of PADS application easier. The APIs are categorized into groups such as DS-LP, DS-NM, DS-Detect, DS-EventList, DS-LBTS, and so on. While basic service routines and their related API are developed, the complete set of interface routines and their implementations are left as future work.

6 CONCLUDING REMARKS AND FUTURE WORK

Cloud-Based Simulation, specially its distributed counterpart, appears to become a trending move as newcomers of cloud services. While sequential CBS can become an easy deployment target, the same cannot be through for the distributed and parallel simulation. The latter is in the HPC domain where multiple research indicate performance hazards such as networking delay, shared resource overhead, and several others, could diminish the gained benefits. To overcome these issues, we proposed DSaaS, which is a special service dedicated to assist parallel discrete event simulation on the cloud. The proposed system has multilayered architecture which include an HPC-as-a-Service layer, a Middleware simulation engine, and an interface layer. The implementation and testing of the HPCaaS layer is accomplished earlier. The design of the DSaaS is completed and reported here, whereas the implementation and testing of this architecture along with a few use-cases are left for the future work.

REFERENCES

AbdelBaky, Moustafa, Manish Parashar, Kirk Jordan, Hyunjoo Kim, Hani Jamjoom, Zon-Yin Shae, Gergina Pencheva et al. 2012. "Enabling high-performance computing as a service". *IEEE Computer Society Press.* Los Alamitos, CA, pp. 72-80.

Boer, Csaba A., Arie de Bruin, and Alexander Verbraeck. 2008. "A survey on distributed simulation in industry".*Journal of Simulation* 3, no. 1 .pp. 3-16.

Dahmann, Judith S., Richard M. Fujimoto, and Richard M. Weatherly.1997."The department of defense high level architecture". In *Proceedings of the 29th conference on Winter simulation*, pp. 142-149. Atlanta, Georgia.

D'Angelo, Gabriele. 2011."Parallel and distributed simulation from many cores to the public cloud." In *High Performance Computing and Simulation (HPCS), 2011 International Conference on*, pp. 14-23. Istanbul , Turkey,

Foster, Ian, Yong Zhao, Ioan Raicu, and Shiyong Lu. 2008. "Cloud computing and grid computing 360-degree compared". In *2008 Grid Computing Environments Workshop*, pp. 1-10.

Fujimoto, Richard M. 1990. "Parallel discrete event simulation". *Communications of the ACM* 33, no. 10 .pp.30-53. New York, NY.

Fujimoto, Richard M. 2000. *Parallel and distributed simulation systems*. NY, New York: John Wiley& sons.

Gupta A. and D. Milojicic, "Evaluation of HPC Applications on Cloud," in Open Cirrus Summit (OCS), 2011 Sixth, pp. 22-26, 2011.

Gupta, Abhishek, Laxmikant V. Kale, Filippo Gioachin, Verdi March, Chun Hui Suen, Bu-Sung Lee, Paolo Faraboschi, Richard Kaufmann, and Dejan Milojicic. 2013. "The who, what, why and how of high performance computing applications in the cloud". In *Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science*.

Jackson, Keith R., Lavanya Ramakrishnan, Krishna Muriki, Shane Canon, Shreyas Cholia, John Shalf, Harvey J. Wasserman, and Nicholas J. Wright. "Performance analysis of high performance computing applications on the amazon web services cloud".2010. *IEEE Second International Conference on Cloud Computing Technology and Science*, pp. 159-168.

Jamalian, S. and Rajaei, H. "ASETS: A SDN Empowered Task Scheduling System for HPCaaS in the Cloud," in Cloud Engineering, 2015, and IC2E IEEE International Conference on, 2015.

Jamalian, S. and Rajaei, H. "Data-Intensive HPC Tasks Scheduling with SDN to Enable HPC-as-a-Service " in the proceedings of IEEE Cloud 2015. July 2015, NY, and IEEE Computer Journal

Liu, Xiaocheng, Qiang He, Xiaogang Qiu, Bin Chen, and Kedi Huang. 2012. "Cloud-based computer simulation: Towards planting existing simulation software into the cloud". *Simulation Modelling Practice and Theory* 26 .pp.135-150.

Medved J., R. Varga, A. Tkacik and K. Gray, "OpenDaylight: Towards a Model-Driven SDN Controller architecture," in 2014 IEEE 15th International Symposium on, pp. 1-6, 2014.

Malik, Asad Waqar, Alfred J. Park, and Richard M. Fujimoto. 2010"An optimistic parallel simulation protocol for cloud computing environments". *SCS M&S Magazine* 4, pp. 1-9.

Mell, Peter, and Tim Grance. 2011. "The NIST definition of cloud computing”. Gaithersburg, MD, National Institute of Standards & Technology .

Padilla, Jose J., Saikou Y. Diallo, Anthony Barraco, Christopher J. Lynch, and Hamdi Kavak. 2014. "Cloud-based simulators: making simulations accessible to non-experts and experts alike". In *Proceedings of the 2014 Winter Simulation Conference*, pp. 3630-3639. Savannah, Georgia IEEE.

Rajaei, Hassan, Rassul Ayani, and Lars-Erik Thorelli. 1993."The local Time Warp approach to parallel simulation". In *ACM SIGSIM Simulation Digest*, ol.23,no.1,pp. 119-126.

Strassburger, Steffen, Thomas Schulze, and Richard Fujimoto. 2008. "Future trends in distributed simulation and distributed virtual environments: results of a peer study". In *Proceedings of the 40th Conference on Winter Simulation*, pp. 777-785. Miami, Florida. Winter Simulation Conference,

Taylor, Simon JE, Azam Khan, Katherine L. Morse, Andreas Tolk, Levent Yilmaz, Justyna Zander, and Pieter J. Mosterman. 2015. "Grand challenges for modeling and simulation: simulation everywhere—from cyberinfrastructure to clouds to citizens". *Transactions of Society for Computer Simulation and simulation international 2015 vol. 91*.

Taylor, Simon J. E., Richard Fujimoto, Ernest H. Page, Paul A. Fishwick, Adelinde M. Uhrmacher, and Gabriel Wainer.2012."Panel on Grand Challenges for Modeling and Simulation". *In Proceedings of the 2012 Winter Simulation Conference (WSC)*. pp. 232, Berlin, Germany.

Taylor, Simon JE, Tamas Kiss, Gabor Terstyanszky, Peter Kacsuk, and Nicola Fantini. 2014. "Cloud computing for simulation in manufacturing and engineering: introducing the CloudSME simulation platform". In *Proceedings of the 2014 Annual Simulation Symposium*, pp. 12. Tampa, Florida. Society for Computer Simulation International.

Vanmechelen, Kurt, Silas De Munck, and Jan Broeckhove. 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* 34.pp.126-143.

AUTHOR BIOGRAPHIES

To be added later