Journal of Computer and Communications, 2014, 2, 22-31
Published Online June 2014 in SciRes. http://www.scirp.org/journal/jcc
http://dx.doi.org/10.4236/jcc.2014.28003

Scientific Research

Migrating a Desktop Simulator of a Chemical Process to the Cloud

Salaheddin Odeh, Mohamad Shaban, Ahmed Qutob

Faculty of Engineering, Al-Quds University, Jerusalem, Palestine
Email: sodeh@eng.alquds.edu, shaban_mohd@hotmail.com, aqutob@eng.alquds.edu

Received 27 April 2014; revised 25 May 2014; accepted 2 June 2014

Abstract

This paper shows how a desktop simulation can be migrated into its cloud equivalence using Windows Azure. It is undeniable that simulators are expensive and cost-intensive regarding maintenance and upgrading, and thus, it is not always feasible to buy such a simulator. Therefore, it will be of great significance if we have an approach, which provides simulators with services through the Internet with the aim of making them accessible from anywhere and at any time. That is, researchers and developers can focus on their actual researches and experiments and the intended output results. The cloud simulation infrastructure of this contribution is capable of hosting different simulations with the ability to be cloned as cloud services. The simulator example used here mimics the process of a distillation column to be seen as a widely used plant in several industrial applications. The cloud simulation core embedded in the cloud environment is fully independent from the developed user-interface of the simulator meaning that the cloud simulator can be connected to any user-interface. This allows simulation users such as process control and alarm management designers to connect to the cloud simulator in order to design, develop and experiment their systems on a "pay-as-you-go" basis as it is the case of most cloud computing services, aimed at providing computing services as utilities like water and electricity. For coding convenience, Windows Azure was selected for both developing the cloud simulation and hosting it in the cloud because of the fact that the source code of the desktop simulator is already available in C# based on dot Net technology. From a software technical point of view, UML graphical notations were applied in order to express the software requirement specifications of the distributed cloud simulation, representing a widespread technology in the object-oriented design and analysis.

Keywords

Cloud Computing, Process Simulator, Object-Oriented Software Engineering, UML, User-Interface Design

How to cite this paper: Odeh, S., Shaban, M. and Qutob, A. (2014) Migrating a Desktop Simulator of a Chemical Process to the Cloud. *Journal of Computer and Communications*, **2**, 22-31. http://dx.doi.org/10.4236/jcc.2014.28003

1. Introduction

Simulations imitate real phenomena using a set of mathematical formulas. In essence, simulations play an important role in many fields such as academic research, e-learning and industrial development [1]. Simulations with real-time response have important industrial applications, especially where the penalty for improper operation is costly as it is the case in many critical systems, for example nuclear power plants, aero planes, chemical plants, to name but a few. Unfortunately, this advantage comes at the cost of more complex simulation systems characterized as often complex, time consuming, error-prone, requiring complex hardware, which are usually very expensive [2].

Currently, most of the simulation programs are still deployed as desktop applications in the form of packaged software requiring regular upgrades and maintenance [3]. A key observation is that the user needs a certain simulator for research and experimenting purposes regarding number of uses are varied: while someone needs it only once such as the case in one-time projects, others' work is fully dependent on such a simulator as it is the case in training and education as well as in user-interface development and design of process control and alarm management systems [4]; the rest need it for a period, for example in some research projects. In this sense, it is not always feasible to buy a simulator since, as stated earlier, such simulators are expensive and cost-intensive regarding maintenance and upgrading. Therefore, it will be of great significance if we have an approach which provides simulators with services through the Internet with the aim of making them accessible from anywhere and at any time. Moreover, the system must be realized in such a way that it must on the one hand allow us to realize the software to be customizable, configurable, high efficient, flexible, agile, real-time, cheap, high available, and reusable. It is indubitable that customers desire to use the software under a usage-based payment model accomplished by hosting simulation software on a remote machine. Essentially, such an approach allows simulation users to obtain access to this machine to carry out their experiments or research, meaning that they can focus on their actual experiments and the intended output results. These objectives can be achieved in numerous ways. An obvious way is the usage of cloud computing, which will be discussed at length in later sections. This approach also has major drawbacks such as there is no guarantee that the remote simulation will be available 24 hours a day/7 days a week and is dependent on other distributed ingredients since some time one simulator may depend on another simulator that maybe not immediately available.

This paper presents a cloud simulation infrastructure capable of hosting different simulations with the ability to be cloned as cloud services. The services of cloud computing can be classified with regard to their common concepts as follows: it is service oriented, from the hardware to the application level, such as Infrastructure/Hardware as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). The technical system used for testing our approach and its appropriateness for complex technical systems is a process simulator of a distillation column, which is a chemical process to separate a chemical mixture of the two substances benzene and Toulon.

The cloud simulation platform is a broad array of web-based services aimed at allowing users to obtain a wide range of functional capabilities on a "pay-as-you-go" as it is the case of most cloud computing services aimed at making computing services as utilities like water and electricity [5]. With our approach, it is possible to provide a complex simulator as a cloud service. Once a user posts a request, a virtual machine is assigned to him, and the simulation will be started on that particular machine with the ability that the cloud system is capable to handle many client requests at the same time so that each request is independent from each other. The user remotely interacts with the cloud simulation through a Web-based user-interface without the necessity to install any software.

The simulator example used here is to mimic the process of a distillation column (see **Figure 1**), which is a widely used plant in the petroleum, chemical, petrochemical and pharmaceutical industries. In essence, its main purpose is the development of new products as well as for the recovery and reuse of volatile liquids [6]. The plant of the distillation column consists of different subsystems (parts) such as a column for separating a mixture, a mixture flow part, a light component and a heavy component section. This process separates an ideal mixture, and it contains several technical components such as a column, a reboiler at the bottom of the column, a condenser, a reflux drum, pumps and several control units managed by operators. Various process variables such as operating pressure, temperatures inside the distillation column, liquid levels etc., play a central role in achieving the degree of separation required during the operation of monitoring and controlling the distillation column process.

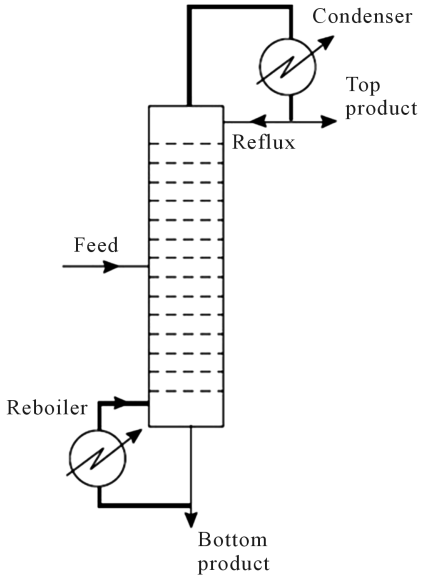

Figure 1. A simplified scheme of a distillation column [6].

In section "Cloud Computing", the different approaches of cloud computing to put our system in the proper place in the cloud computing landscape are reviewed. Section "Windows Azure: The Microsoft Cloud Platform" discusses the technology of windows azure as one of the powerful platforms suitable for deploying cloud-based applications. Based on the object-oriented software engineering approaches, we show the detailed design and implementation of the subsystems of the cloud-based simulation and the integration of the whole system in section "Object-oriented migration of the non-cloud simulator to the cloud". As the Web-based user-interface plays a central role in mediating the physical (real) experiment, section "User-Interface of the cloud simulation" discusses the functionality of the user-interface in addition to various aspects of human-computer interaction, which must be taken into account while developing a user-centered user-interface. The contribution finalizes with a brief conclusion.

2. Cloud Computing

According to Lin *et al.* [7], cloud computing is a business and technology concept in the beginning stages of existence with different meanings and perspectives. Application and IT users consider it as IT as a service (ITaaS) for delivering of computing, storage, and applications over the Internet from centralized data centers. On the contrary, Internet application developers deal with it as an Internet-scale software development platform and runtime environment. Infrastructure providers and administrators view it very differently, namely, it's the massive, distributed data center infrastructure connected by IP networks [7]. Virtualization is the technology on which cloud computing is based with the pay-as-you-go manner for the services. It can be categorized into private and public clouds [8]. Recently, these concepts have been applied in different fields such as social networks, business applications, and content delivery networks [9]. In general, all cloud services share some common characteristics. First, they have an open standard interface to achieve interoperability, e.g., WSDL, XML. Second, they can be placed (published) in an Internet-searchable repository. Third, users have the possibility to automatically discover (search) the services. Fourth, they offer the possibility of being remotely invoked via a standard protocol, e.g., SOAP-remote procedure call. Fifth, services are platform-independent, meaning that they can be written in any language such as Java, C#, etc. and finally, every piece of program can be wrapped as a service.

Due to the fact that cloud computing is an innovation of traditional computing model, it is very important to correctly understand the cloud's architecture. **Figure 2** summarizes the architecture of the current cloud computing, which is divided into two layers: the bottom resource and the upper service layer. While the bottom structuring the foundation is based on virtualized resources in the form of storage and computing, the upper service layer provides the three specific services: SaaS, PaaS and IaaS. SaaS allows the user to rent or use the software over a network instead of buying the software and running it on a specific computer with the key advantage that

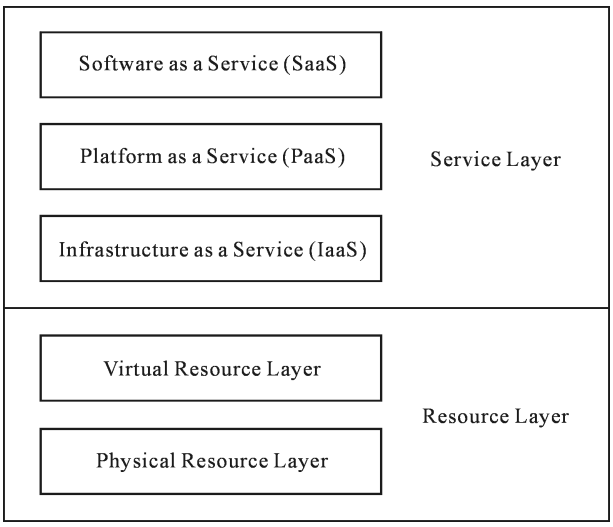

Figure 2. Cloud computing architecture.

SaaS fastens the delivery of applications and the new functionality to the users [10] [11]. To this end, this will save some cost on servers and software not only for the users, but also for the providers as well due to the fact that they need to maintain one program [12]. As examples of SaaS are GoogleDocs, Microsoft Office Web Apps and Acrobat.com. They are aimed at improving internal resource productivity without the necessity to manage its own resources, rapid delivering of new applications and functionality with immediate reach to all users connected to a network, and providing better government services with standardization in services for all users. PaaS is the middle bridge between hardware and application; it offers platform and development environment. This means that the system capacity is scalable and flexible dynamically as needed to a higher abstraction level. The end users can write their own code and PaaS providers upload that code and present it on the Web. For users, PaaS lowers the total cost of owning and manages all lower abstraction resources like hardware. Examples of PaaS are Amazon Elastic Computing (EC2), Google App Engine, Windows Azure, SalesForce, GoGrid and Rackspace. All resources needed are located at a single datacenter, which serves all end users. IaaS introduces servers, networks, memory, CPU and storage to the users on demand instead of owning them. For example, users can rent capacity for storage and server instances when they need them. For the providers, virtualization allows offering unlimited instances of servers to customers and makes cost-effective use of the hosting hardware [13]. Examples of IaaS are GOOGLE, IBM, GoGrid, Eucalyptus, Terremark Worldwide, Amazon S3, and Rackspace.

Recently, X as a Service with the meaning that everything as a service or anything as a service, has entered the vocabulary of many papers, where X is software, hardware, platform, data, business, and so on [14]. More recent developments include, for example, robot as a service [15], simulation as a service [16], cloud database as a service [17], collocation as a service [18], application software as a service [19], computer resources as a cloud lab Service [20] etc. Although there are many examples of simulation approaches based on the cloud technology, in the following, we will discuss that proposed by Härri *et al.* [16]. In their paper, an approach to evaluate the intelligent transportation system (ITS) application is suggested. The ITS will get benefit from cloud computing in reducing the high demand in processing power that is needed for the transportation simulation. The system was implemented using a Web-server as front-end, a controller as back-end, and high capacity computing (HCC). Kernel-based virtual machines (KVMs) was employed on the HCC and each virtual machine represents a fully simulation environment. The front-end server provides a simple Web interface for users to configure and upload the simulation scenario, and thus, it is the interface between the user and the HCC. The back-end controller is the core of the simulation as a service (SaaS) consists of a controller that connects to an HCC platform. Once the controller receives the requests from the front-end server, it distributes the jobs to the required and available HCC machines and sends the result back to the front end.

3. Windows Azure: The Microsoft Cloud Platform

Due to the vast improvement in cloud computing and increasing of client demand for cloud services, several

cloud center companies came to light such as Amazon Elastic Computing (EC2), Google App Engine, Windows Azure, SalesForce, GoGrid and Rackspace. Earlier we have developed a conventional simulation of the distillation column with the help of visual studio 2010 on the basis of Microsoft windows and dot Net. The mathematical model of this simulation originally created by Settgast [21] [22] is a simplified version of a more complicated process simulator [23], which is used as a training tool for process operators.

One reason why our decision fell on Windows Azure which is very suitable for Microsoft Visual Studio is that the source code of the non-cloud version of the distillation column is already available in C# which was developed using dot Net technology. It includes the Windows Azure SDK, allowing extending Visual Studio 2010 to enable the creation, configuration, building, debugging, running, packaging and deployment of scalable web applications and services on Windows Azure [24] [25]. Visual Studio 2010 includes a local fabric with the ability to simulate the behavior of a real Windows Azure fabric, consisting of a computing and a storage emulator. For making cloud-based applications possible, developers must create a Windows Azure role which is an individually scalable component running in the cloud where each instance of a role corresponds to a virtual machine (VM) instance. There are two types of such a role, namely, Web and worker role. A Web role corresponds to a Web application and runs on an Internet information service. It deals with the users' input and accepts incoming HTTP or HTTPS requests. Oppositely, a worker role acts as a background application that runs arbitrary dot Net code [26].

As we mentioned before, Windows Azure comes under the PaaScategory and the users hire a platform to deploy their applications without any need for configuring the infrastructure giving them the ability to get rid of the limitations of the SaaS approach. Windows Azure provides a distributed operating system, where users can deploy and run their applications without any transaction with the Windows interface, that is, no access is necessary to the underlying virtualization environment or operating system details like network interface, IP configuration, and disk management. One major benefit of this approach is helping developers to begin creating their applications almost immediately, since the vendor provides and manages everything, from the network connectivity to the runtime. However, there are also many other advantages [26], for example, completely abstracting the physical components of the system, which gives customers the freedom to select the provided computer components with different level of service level agreement (SLA), providing redundancy by storing data as replicas on three nodes to guarantee scalability and fault-tolerance, applying load balancers to distribute computing jobs in a completely transparent way, and allowing customers to choose among a range of virtual machine types that can be described using computer components such as CPUs, RAM, Instance storage disk space and I/O performance.

4. Object-Oriented Migration of the Non-Cloud Simulator to the Cloud

4.1. Design, Development and Implementation

In the object-oriented design and analysis, use cases and scenarios for graphical modeling of the application and the solution domain are important ingredients of the requirement specification. They present essential stages in the object-oriented software engineering process. A key observation of object-oriented software engineering is that it is an iterative process, that is, activities can occur in parallel and more than once. Diagrams such as classes and objects are well-known presentations for modeling the aforementioned domains that are included in the unified modeling language (UML), which is a notation that resulted from the unification of OMT (Object Modeling Technique) [27] [28], and has been designed for a broad range of applications because of the fact that it provides constructs for a broad range of systems and activities, e.g., distributed systems, analysis, system design, deployment. In this approach, emphasis is also placed on expressing the functional requirements for the system using the UML graphical notations. Graphical models are most useful when you need to show how state changes or where you need to describe a sequence of actions. Currently, use-case descriptions and sequence diagrams are commonly used [29]. Use-cases are a scenario-based technique for requirements elicitation and have now become a fundamental feature of the UML notation for describing object-oriented system models. They identify the type of interaction and the actors involved. Where interactions are represented as named ellipses, actors are identified through stick figures. Sequence diagrams exhibit the actors involved in the interaction, the objects they interact with, and the operations associated with these objects. Objects and actors are aligned along the top of the diagram. Labeled arrows show operations and their sequences that are ordered from top to bottom. A sequence diagram is to read from top to bottom to obtain the order of the actions that take place [30].

To migrate the non-cloud simulation to the cloud using Windows Azure and Visual Studio 2010, the following steps are necessary: In the first place, a Windows Azure project using visual studio 2010 with a Web Role that must be created. After that, the source code must be locally written and tested using Windows Azure local fabric. Finally, the developer deploys the application to the Windows Azure environment. After having done so, the simulation application is hosted in the Windows Azure environment, allowing users to access it through the Internet using a Web browser. The use-case diagram shown in **Figure 3** exposes what takes place in the Windows Azure cloud environment once a user starts the Web-based cloud simulation. As shown in this figure, when the user activates the Web-based cloud simulation homepage, an HTTP request will be sent to the simulator hosted in Windows Azure, which will then ask for username and password for billing issues. In a further step, a Web role takes this request and cooperates with the worker role to assign an instance to this request, causing the simulator to be loaded to one of the virtual machine available, which in turn starts to run. If there is no VM machine available the request waits in a queue until one VM will be free. During the interaction with the Web-based cloud simulation, the user has the freedom to pause the simulation, and then to continue the simulation session at a later time. It is worth mentioning that an important issue of the cloud environment is that if one instance is broken for any reasons, Windows Azure automatically creates a new instance instead of the broken one without that the user notices the failure, leading to make the cloud system failure transparent, which according to Tanenbaum and Van Steen [31] means that a user does not notice that a resource fails to work properly, and that the system subsequently recovers from that failure.

For determining the necessary classes of the cloud simulation, a UML technique designated as Class Responsibility Collaborator analysis (CRC analysis) using cards was applied [32]. CRC (Class-Responsibility-Collaborator) Card Modelling is a simple powerful object-oriented analysis technique, which often includes the users, analysts, and developers in a modelling and design process, and as such brings together the entire development team to form a common understanding of an object-oriented development project. A CRC model is a collection of cards (usually standard index cards or larger) that are divided into three sections: Class, Responsibility and Collaborator. A class represents a collection of similar objects that are things of interest in the system being modelled, and can be a person, place, thing, or any other concept important to the system at hand. The class name appears across the top of the CRC card. A responsibility is anything that the class knows or does. These responsibilities are things that the class has knowledge about itself, or things the class can do with the knowledge it has. A collaborator is another class that is used to get information from, or perform actions for the class at hand. It often works with a particular class to complete a step (or steps) in a scenario. The collaborators of a class appear along the right side of the CRC card.

Based on the dynamic of continuous distillation columns, a CRC analysis of the cloud simulation was carried

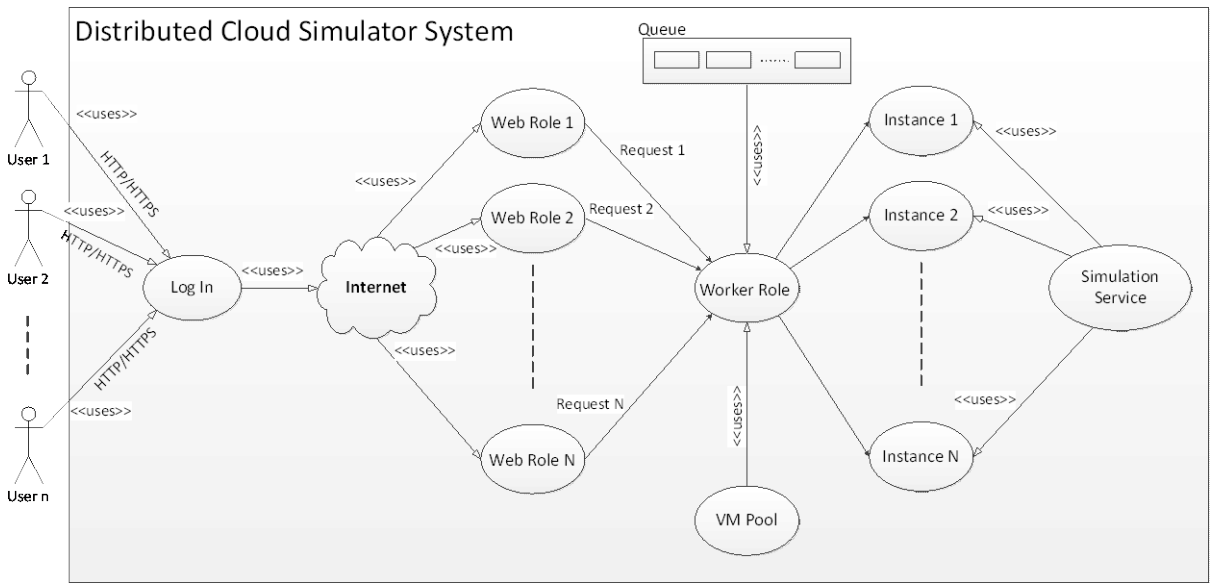

Figure 3. Use case diagram of the Web-based cloud simulation.

out, in which all the necessary CRC steps were applied. As a result, we acquire the core classes: user-interface, variable, relations, web role and worker role. As an example, we will show the user-interface class to demonstrate the methodology. While doing this, an emphasis is placed on the class methods and the collaborator class. As an example, the user-interface class has class methods such as *SimInterface*, *Main method*, *Compute* etc. and collaborator methods such as *relations*. The development of the system follows the prototyping approach because of the fact that it is suitable to resolve requirements and design uncertainties as it is the situation in most of the research projects [33]. Software is developed in a series of increments with each increment including new system functionality. The software components were incrementally built within an iterative system and software engineering process. The stages specification, design, development, and testing within the iterative design process are not chained, but rather interleaved and concurrent [30].

The libraries of the visual programming environments such as dot Net used to create windows applications are more powerful to its corresponding Web-based tools. To develop the system, various development tools are used such as Visual C# as part of Visual Studio 2010 pro., Microsoft Windows Server 2008 SP2, Windows Azure Tools VS2010, Azure Storage Explorer 4, Windows Azure SDK-x86, Windows Azure Management Tool and HTTP Analyzer v6. The dot NET technology allows quick creation through drawing and placing of graphical objects on the user display. Dot Net technology includes graphical user interface tools with rich libraries for user interface components, enabling data to be displayed in many forms.

4.2. Interaction with the Cloud Simulation Environment

The workflow of how to start and use a cloud simulation instance hosted in Windows Azure shows **Figure 4** consisting of three main steps. First, the user requires the cloud environment through a conventional Web browser to obtain a cloud service.

This causes a Web-role to send the user a request prompting him to enter his login information. Once the user logs in, the load balancer checks the availability of a free virtual machine; if there is a free VM in the VM pool, the load balancer assigns one to the requested user; otherwise, this request will be pending in a queue until one VM becomes free. Afterwards, an instance of the simulator will be loaded into that VM machine and the user can start interacting via a Web browser with it. As stated earlier, the user can temporarily stop his simulator session and store its context in order to be continued later. At the end of using the simulation, the user logouts and this leads Windows Azure to terminate the cloud session in a such a way that, on the one hand, the assigned VM machine will be returned to the VM pool, and on the other, it calculates the costs of using the cloud simulation

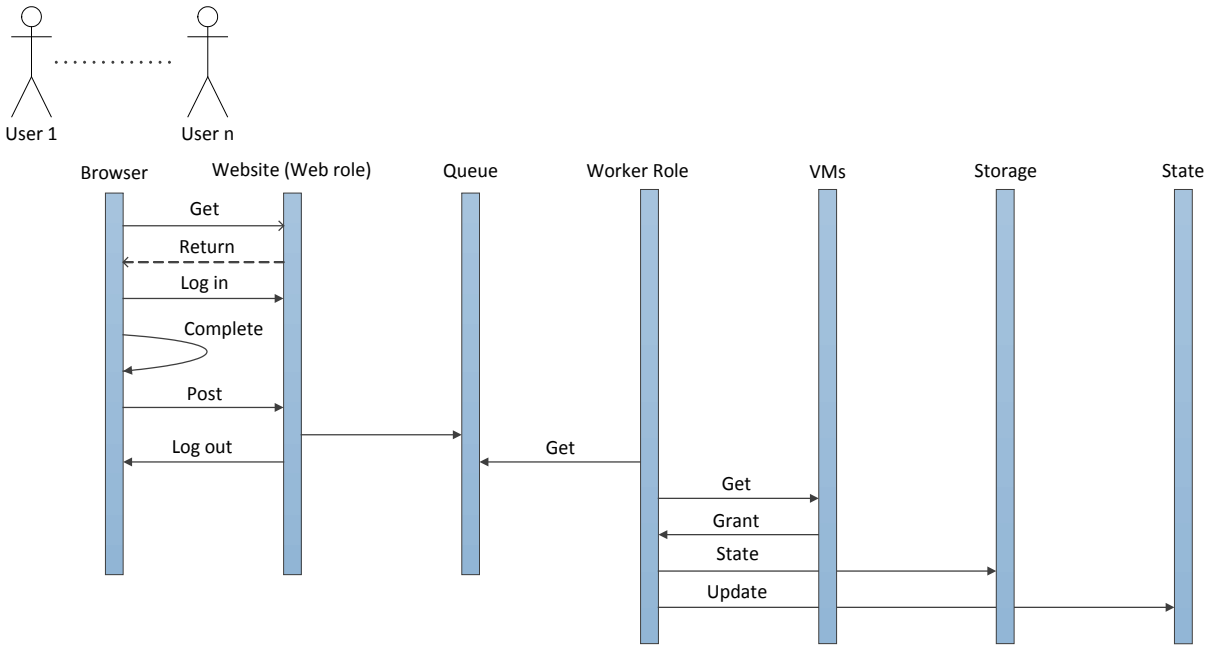

Figure 4. Sequence diagram of the Web-based cloud simulation showing starting and using a cloud simulation instance.

in dependence on time and resources used. In light of distribution transparency, each user works in an independent way from the other users and as such no interference between user's data and information.

5. User-Interface of the Cloud Simulation

It is necessary to clarify some term definitions related to human factors that must be taken into account when designing graphical user-interfaces for monitoring technical systems. This will not only simplify distinguishing between the terms used in this contribution, but it will mediate the concept as well. From the ergonomic perspective, user-interface designers must distinguish between aspects of perceptive and cognitive ergonomics. The cognitive ergonomics relates to reasoning, memory and knowledge. Here, we are more concerned with perceptive ergonomics, focusing on designing issues such as color, shape form, dimension and allocation, highlighting and so on [34]. **Figure 5** shows the implemented Web-based user-interface of the cloud simulation consisting of many visual attributes that help user to operate and follow the simulator moment by moment.

This implemented user-interface is fully independent from the simulator core and its role in order to provide users with the ability to interact with the simulator. This refers to the fact that the simulator core runs in the Azure environment and therefore it can be connected to any user-interface. One of the major reasons why this is of great significance is that user-interface designers for e-learning training systems or process control and alarm management developers can design and develop their systems as desired and they have the possibility to connect to the simulator to test their creations.

6. Conclusion

After having completed this project, we came to the conclusion that cloud computing offers the promise of outsourcing the task of providing and managing execution platform to users; it offers the possibility of making a simulation technology much more readily accessible to the simulation community. Such simulator as cloud service can help in having high efficiency, flexibility, higher availability and reusability. Moreover, it can process diversified requests dynamically and respond to the requests in real-time with the ability that it can be presented, customized and configured as users' desire. In this approach, no up-front commitment by users is necessary, for

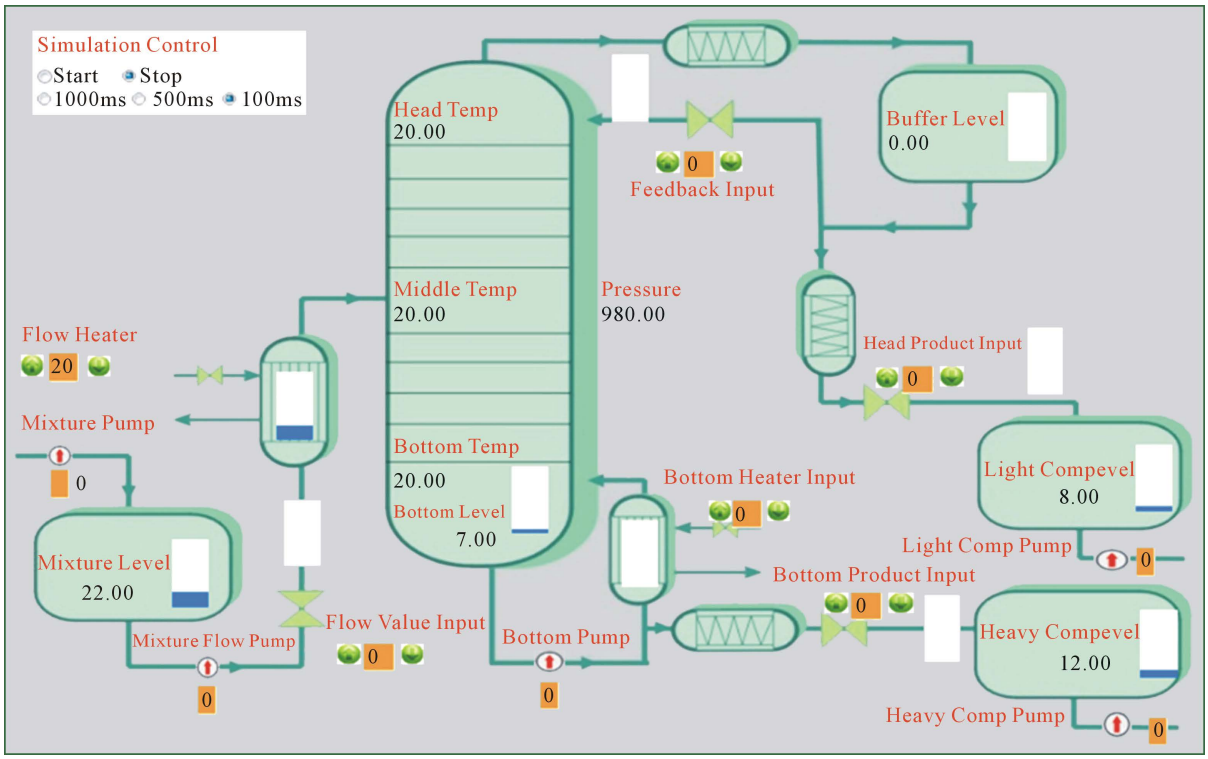

Figure 5. Web-based user-interface of the cloud simulation of a distillation column process.

they only have to pay for use of the resources on a short-term basis leading to the fact that simulation as cloud service reduces operational costs for hardware, shipping, set-up time, travels, and staffing. Simulation as cloud service can be applied for software development such as user-interface design. It can be very helpful for training and e-learning, and it provides publish and share simulation services.

References

[1] Zabre, E. and Román, R. (2008) Evolution, Tendencies and Impact of Standardization of Input/Output Platforms in Full Scale Simulators for Training Power Plant Operators. *World Academy of Science, Engineering and Technology*, **2**, 951-959. http://waset.org/Publications/evolution-tendencies-and-impact-of-standardization-of-input-output-platforms-in-full-scale-simulators-for-training-power-plant-operators/11622.

[2] Guo, S. (2012) Simulation Software as a Service and Service Oriented Simulation Experiment. Ph.D. Dissertation, Georgia State University, Atlanta. http://www.cs.gsu.edu/xhu/papers/Dissertation_SongGuo_Final.pdf.

[3] Liu, W., Du, Z., Chen, Y., Chai, X. and Wang, X. (2010) On an Automatic Simulation Environment Customizing Services for Cloud Simulation Center. *Proceedings of the Fifth IEEE International Symposium on Service Oriented System Engineering* (*SOSE*), Nanjing, 4-5 June 2010, 214-221. http://dx.doi.org/10.1109/SOSE.2010.43.

[4] Ali, S. (2003) Approximative Process Visualization Based on Qualitative Knowledge and Fuzzy Logic. In: Borys, B.-B. and Wittenberg, C., Eds., *From Muscles to Music*: *A Festschrift to Celebrate the* 60*th Birthday of Gunnar Johannsen, Kassel*, Kassel University Press, Kassel, 199-209. http://www.uni-kassel.de/upress/online/frei/978-3-933146-87-8.volltext.frei.pdf

[5] Yamazaki, T., Ikeno, H., Okumura, Y., Satoh, S., Kamiyama, Y., Hirata, Y., Inagaki, K., Ishihara, A., Kannon, T. and Usui, S. (2011) Simulation Platform: A Cloud-Based Online Simulation Environment. *Neural Networks*, **24**, 693-698. http://dx.doi.org/10.1016/j.neunet.2011.06.010.

[6] Sinnott, R. and Towler, G. (2012) Chemical Engineering Design: Principles, Practice and Economics of Plant and Process Design. Butterworth Heinemann, New York.

[7] Lin, G., Fu, D., Zhu, J. and Dasmalchi, G. (2009) Cloud Computing: IT as a Service. *IT Professional*, **11**, 10-13. http://doi.ieeecomputersociety.org/10.1109/MITP.2009.22.

[8] Peng, J., Zhang, X., Lei, Z., Zhang, B., Zhang, W. and Li, Q. (2009) Comparison of Several Cloud Computing Platforms. *Proceedings of the Second International Symposium on Information Science and Engineering* (*ISISE*), Shanghai, 26-28 December 2009, 23-27. http://dx.doi.org/10.1109/ISISE.2009.94.

[9] Vecchiolal, C., Pandey, S. and Buyya, R. (2009) High-Performance Cloud Computing: A View of Scientific Applications. *Proceedings of the* 10*th International Symposium on Pervasive Systems*, *Algorithms and Network I-SPAN* 2009, *IEEE Comupter Society*, Kaohsiung, 14-16 December 2009, 4-16.

[10] Mutavdzic, R. (2010) Cloud Computing Architectures for National, Regional and Local Government. *Proceedings of the* 33*rd International Convention*, *MIPRO* 2010, Opatija, 24-28 May 2010, 1322-1327.

[11] Wei-Tek, T., Xin, S. and Balasooriya, J. (2010) Service-Oriented Cloud Computing Architecture. *Proceedings of the Seventh International Conference on Information Technology*: *New Generations* (*ITNG*), Las Vegas, 12-14 April 2010, 684-689. http://dx.doi.org/10.1109/ITNG.2010.214.

[12] Zhang, S., Zhang, S., Chen, X. and Huo, X. (2010) Cloud Computing Research and Development Trend. *Proceedings of the Second International Conference on Future Networks*, *ICFN*'10, Sanya, 22-24 January 2010, 93-97. http://dx.doi.org/10.1109/ICFN.2010.58.

[13] Barnatt, C. (2010) A Brief Guide to Cloud Computing: An Essential Guide to the Next Computing Revolution, Constable & Robinson Limited, London.

[14] Schaffer, H.E. (2009) X as a Service, Cloud Computing, and the Need for Good Judgment. *IT Professional*, **11**, 4-5. http://dx.doi.org/10.1109/MITP.2009.112.

[15] Chen, Y., Du, Z. and García-Acosta, M. (2010) Robot as a Service in Cloud Computing. *Proceedings of the Fifth IEEE International Symposium on Service Oriented System Engineering* (*SOSE*), Nanjing, 4-5 June 2010, 151-158. http://dx.doi.org/10.1109/SOSE.2010.44.

[16] Harri, J., Killat, M., Tielert, T., Mittag, J. and Hartenstein, H. (2010) DEMO: Simulation-as-a-Service for ITS Applications. *Proceedings of the* 71*st IEEE Vehicular Technology Conference*, Taipei, 16-19 May 2010, 151-158.

[17] Mateljan, V., Cisic, D. and Ogrizovic, D. (2010) Cloud Database as a Service. *Proceedings of the* 33*rd International Convention*, *MIPRO* 2010, Opatija, 24-28 May 2010, 1185-1188.

[18] Ishakian, V., Sweha, R., Londono, J. and Bestavros, A. (2010) Colocation as a Service: Strategic and Operational Services for Cloud Colocation. *Proceedings of the* 9*th IEEE International Symposium on Network Computing and Appli-*

cations (*NCA*), Cambridge, 15-17 July 2010, 76-83.

[19] Hou, Z., Zhou, X., Gu, J., Wang, Y. and Zhao, T. (2010) ASAAS: Application Software as a Service for High Performance Cloud Computing. *Proceedings of the* 12*th IEEE International Conference on High Performance Computing and Communications* (*HPCC*), Melbourne, 1-3 September 2010, 156-163.

[20] Odeh, S. and Al-Khatib, Y. (2012) Computer Resources as a Cloud Lab Service. *Proceedings of the* 2012 *IEEE Global Engineering Education Conference* (*EDUCON*), Marrakesh, 17-20 April 2012, 674-678. http://dx.doi.org/10.1109/EDUCON.2012.6201121

[21] Settgast, D. (1993) Qualitative Simulation einer Destillationskolonne (Studienarbeit). Unpublished Graduation Report (in German), Human-Machine Systems Engineering, University of Kassel, Kassel.

[22] Settgast, D. (1993) Analyse von Prozessführungsproblemen bei einer Destillationskolonne (Studienarbeit). Unpublished Graduation Report (in German), Human-Machine Systems Engineering, University of Kassel, Kassel.

[23] Gilles, E.D., Holl, P., Marquardt, W., Schneider, H., Mahler, R., Brinkmann, K. and Will, K.H. (1990) Ein Trainingssimulator zur Ausbildung von Betriebspersonal in der Chemischen Industrie (in German). *Sonderheft Atp-Namur Statusbericht*'90, 261-268.

[24] Windows Azure. http://www.windowsazure.com/en-us/

[25] Redkar, T. (2010) Windows Azure Platform. Apress, New York.

[26] Brunetti, R. (2011) Microsoft Windows Azure Step by Step. Microsoft Press, Sebastopol.

[27] Blaha, M. and Rumbaugh, J. (2004) Object-Oriented Modeling and Design with UML. 2nd Edition, Prentice Hall, Upper Saddle River.

[28] Booch, G., Maksimchuk, R., Engle, M.W., Young, B.J., Conallen, J. and Houston, K.A. (2007) Object-Oriented Analysis and Design with Applications. 3rd Edition, Addison-Wesley Professional, New York.

[29] Stevens, P. and Pooley, R. (2006) Using UML: Software Engineering with Objects and Components. 2nd Edition, Addison-Wesley, Harlow.

[30] Sommerville, I. (2011) Software Engineering. 9th Edition, Addison-Wesley, New York.

[31] Tanenbaum, S. and Van Steen, M. (2007) Distributed Systems: Principles and Paradigms. 2nd Edition, Prentice Hall, Upper Saddle River.

[32] Beck, K. and Cunningham, W. (1989) A Laboratory for Teaching Object-Oriented Thinking. *Proceedings of the OOPSLA*'89 *Conference on Object-Oriented Programming Systems, Languages and Applications*, ACM New York, **24**, 1-6.

[33] Faison, T. (2002) Component-Based Development with Visual C#. M & T Books, John Wiley & Sons, New York.

[34] Anderson, J.R. (2009) Cognitive Psychology and Its Implications. 7th Edition, Worth Publishers, New York.

Scientific Research Publishing (SCIRP) is one of the largest Open Access journal publishers. It is currently publishing more than 200 open access, online, peer-reviewed journals covering a wide range of academic disciplines. SCIRP serves the worldwide academic communities and contributes to the progress and application of science with its publication.

Other selected journals from SCIRP are listed as below. Submit your manuscript to us via either submit@scirp.org or Online Submission Portal.