Author's response to reviews

Title: The Cloud Paradigm Applied to e-Health

Authors:

Jordi Vilaplana (jordi@diei.udl.cat)
Francesc Solsona (francesc@diei.udl.cat)
Francesc Abella (abella@gss.scs.es)
Rosa Filgueira (rosa.filgueira@ed.ac.uk)
Josep Rius (jrius@diei.udl.cat)

Version: 4 Date: 21 December 2012

Author's response to reviews: see over
Abstract
Point-by-point response to the reviewer’s comments. All the changes in the manuscript have been highlighted in blue.

1 Reviewer 1

“*The manuscript presents a cloud based platform for the e-Healthcare field to enhance the results of diagnosis. The objective is to improve treatments by extracting statistical and useful data from thousands of patients with different diseases. The cloud system modeling approach applied to medical field was also proposed in order to take advantages of cloud computing such as scalability, maintainability and quality of services (QoS). To improve the QoS, the queuing system has been applied to model the performance of the cloud system.*

The proposed platform consists of two layers, front end and back end. In the back end, the platform consists of three servers, primary, specific, and control. The primary servers are responsible for performing most of the computations. Specific servers are responsible for the communication with the database. Control servers are responsible for creating and removing virtual machines according to the tasks load status. The system uses two queues to manage tasks: one for the main servers and the other for specific servers. The main purpose of the queues is to store tasks that are waiting to be processed. All tasks are served according to first-come-first-serve and are managed by the scheduler.

1. Major Compulsory Revisions: ”

1.1 Comment 1

“(1)- The proposed queuing system modeling and cloud system architecture are interesting. The manuscript insists that the architecture of the cloud system can be applied to the e-Health field. However, the manuscript didn’t specifically describe how the proposed system can benefit in e-Health field. For example, there is no evidence of how the cloud platform can enhance the results of therapies and how the platform can improve current treatments by extracting statistical and useful data from thousands of patients.”
We have described how the proposed system can benefit the e-Health field. Now you can find the following paragraph at the end of the conclusions section.

1.1.1 Solution
The following paragraph has been included to the manuscript.

The proposed system can improve the e-Health field by providing a model to support medical software, saving resources and enhancing the control and management of the patients. Pay-per-use service would lower overall costs. The proposed system would also allow tendencies that could be used to improve the current treatments to be generated and analysed. Also, having an electronic clinical history would save paper, physical space and would improve the efficiency of those who need to access it. The design can easily satisfy the needs of e-Health related applications without major changes, allowing the construction of web-based applications that implement all the needed medical workflows. The proposed system guarantees high scalability, ensuring that when the system requirements grow, the underlying platform will be able to handle the situation. Also, the proposed system suggests the usage of a large shared infrastructure, which would result in many hospitals and treatments having homogeneous data that would facilitate correlations and data mining.

1.2 Comment 2

"(2)- The waiting time, as a QoS criterion, is measured carefully in well-explained metrics that are also adopted in the proposed well-structured algorithm. The First-Come-First-Serve (FCFS) is a common approach to process tasks in the queue. However, it happens frequently in the medical workflow that there may have multiple tasks that depend on each other. In such a case, a deadlock may occur. The pros and cons of using FCFS are recommended to clearly specify.

It is well known that deadlocks are a potential weakness for a First-Come-First-Serve queue. However, in this manuscript we are proposing a generic system which will have web-based servers at its top. For this reason, all the medical workflows would be implemented at a software level, and all the tasks arriving in our model are supposed to be web requests. The approach of a FCFS queue has been used in order to simplify the model analysis.

1.2.1 Solution
For these reasons we do not explicitly address the deadlock issue in our manuscript. We have added a reference to this situation at the Cloud Architecture subsection of Background section.

As we are proposing a generic system, medical workflows will not be implemented as part of our model. Instead, these medical workflows will be implemented via software. All arriving tasks in our model will consist of web
requests, avoiding deadlock situations that could otherwise arise when using a FCFS queue policy.

1.3 Comment 3

“(3)- The proposed queuing system model is able to create/delete virtual machine dynamically based on average waiting time, but the manuscript did not describe how to create/delete the VM and which type of VM should be created/deleted.”

We have explained how OpenStack allows us to create/delete virtual machines dynamically.

1.3.1 Solution

We have added the following paragraph to the OpenStack section of the manuscript.

OpenStack offers a set of APIs (Application Programming Interface) that allow to interact dynamically with the installed OpenStack platform. Using these APIs, it is possible to authenticate and interact with the system from the command line or programmatically. For example, in Python we have available the python-nova client API [1, 2] available, where the nova boot and nova delete commands allow us respectively to boot a new server and immediately shut down and delete a server dynamically.

1.4 Comment 4

“(4)- Most of the figures are not clearly described. More detailed explanations to the figures are recommended.”

We have added more detailed descriptions for most of the figures.

1.4.1 Solution

We have slightly modified the description for Figure 1.

PaaS allows users to create their own cloud applications, providing all the execution and compilation of software as well as operating systems.

We have rephrased the description for Figure 2.

Design of the proposed cloud architecture. User requests from multiple devices go through a HTTP interface to the cloud system. A First-Come-First-Serve scheduler distributes all these requests to the Front-end nodes, which forward these to the Back-end nodes. The Back-end nodes process the requests and compute the expected user result, accessing the system database if needed. In the Back-end, there are also control nodes that monitor the state of the system, and are able to create or destroy virtual machines according to that state.

We have added a more detailed description for Figure 3.
The first $M/M/m$ queue models the access to the primary servers, and is always accessed when new requests enter the system. The second $M/M/m$ queue models the access to the database cluster, which is accessed based on a probability depending on the Back-end nodes.

We have added a more detailed description for Figure 4.

The first one models the access to the primary servers, and the second one models the access to the database. $\lambda$ is the request arrival rate. There is a probability $d$ of accessing the second queue, and a probability $(1 - d)$ of exiting the queueing system without going through the second queue.

We have extended the description for Figure 5.

Representation of an $M/M/m$ queuing system with $m$ servers and two density functions. The average arrival rate of the requests is represented by $\lambda$. The total number of servers goes from one to $m$, each one having a service rate represented by $\mu$.

We have extended the description for Figure 6.

The state space records the number of customers in the queueing system. The values $\lambda$ and $\mu$ represent the arrival and service rates of requests.

We have extended the description for Figure 7.

It shows that increasing the number of servers significantly decreases the resulting waiting time.

We have extended the description for Figure 8.

It shows that increasing the number of servers significantly decreases the resulting waiting time.

We have extended the description for Figure 9.

Graph plotting the waiting time ratio between waiting times of the first and second queues with one and a hundred servers, for different $\rho$ values.

1.5 Comment 5

“(5)- The proposed platform uses two queues, one for the computation and the other for the database. The manuscript should explain how the two queues will be managed and how they can coordinate with each other.”

We have detailed how the two queues interact with each other from the queueing theory perspective.
1.5.1 Solution
We have added the following paragraph to the Modelling section.

Our two queues can be analysed independently, and they form an open Jackson network. The interconnection and behaviour between the queues is ruled by Burke’s [9] and Jackson’s theorems. Burke states that we may connect many multiple-server nodes together in a feedforward network and still preserve the node-by-node decomposition. Jackson [10, 11] states that to calculate the total average arrival rate we must sum the arrivals from outside the system plus arrivals from all internal nodes.

1.6 Comment 6
“(6)- In the testing, many variables were assumed. There is no clear evidence on the implementation parameters or the assumptions.”

Our manuscript did not specify clearly why some of those variable values were chosen. We have justified why we assumed a value for $d$ (database access rate), $W_{\text{min}}$ (minimum response time) and $W_{\text{max}}$ (maximum response time).

1.6.1 Solution
We have added the following paragraphs to the manuscript:

The mean access rate to the database $d$ can widely vary from one application to another. We have assumed a 0.9 value due to our experimental application making requests to the database for 90% of the user requests. We also did some testing with slightly modified values of $d$ and proportional results were obtained.

According to the guidelines stated by Shneiderman [5, 4, 6], a system’s response time should be appropriate to the task that is being performed. For typing, cursor motion and mouse selection, they define an interval of between 50 and 150 milliseconds, and a value of 750 to 1000 milliseconds for simple and frequent tasks. The customers of our system will be performing simple and frequent tasks due to the interaction with a web-based application. For these reasons, we assume a $W_{\text{min}}$ value of 150ms and a $W_{\text{max}}$ value of 750ms. These values could be modified to analyse other kinds of system where mean acceptable response times could be different.

1.7 Comment 7
“(7)- In page 3, PaaS does not provide hardware capability. PaaS provides all execution and compilation of software as well as operating systems.”

We have clarified our explanation, which was poorly written and gave the impression that with PaaS customers could manage the underlying hardware.
Our point was that with PaaS, customers do not manage or control the underlying cloud infrastructure, such as network, servers, operating systems or storage, but have control over the deployed applications [3].

1.7.1 Solution

We have rewritten the PaaS explanation in page 3:

And as a combination of both, Platform as a Service (PaaS), which means offering the capability to deploy applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure, but has control over the deployed applications [3].

1.8 Comment 8

“ (8)- To make the terms consistent, using either “primary server” or “main server” is recommended, but not both. ”

We have made the terms consistent, always using the “primary server” term.

1.8.1 Solution

The term “main server” has been changed to ”primary server” throughout the manuscript.

1.9 Comment 9

“ (9)- Many sentences need to be rephrased and the manuscript contains many spelling and grammar errors: for example, the third paragraph in page 4, “, this is cloud systems systems made up by hundreds”. Typos, such as “e-Healt” in the 2nd paragraph of page 3. The 1st paragraph in page 6, “manly” should be “mainly” and “tack-ing” should be “taking.” ”

The entire manuscript has been checked. These and some other typos/mistakes have been addressed.

1.10 Comment 10

“ (10)- Not all references were cited in the text. ”

All the references have been revised and updated.

1.11 Comment 11

“ (11)- No all references are scientifically credited to be used like Wikipedia. ”

The manuscript has been updated and all the references to Wikipedia have
been removed. Product webpage references, like VMWare, that are not scientifically credited, have been kept as a guidance for a more detailed product description.

2 Reviewer 2

“Minor Essential Revisions.
Considerations for authors: ”

2.1 Comment 1

“* Abstract has to be update in order to clarity that the "results" paragraph is where this work will be used and the expected impact. The "results" paragraph seems to me the summary of the main contributions. "It is though to include.." -δ "It is thought to include.." ”

We have reviewed and updated the abstract and corrected the stated typo.

2.1.1 Solution

The following paragraph from the Abstract section has been rephrased in the manuscript.

Detailed results for the model formed by a Jackson network of two M/M/m queues from the queueing theory perspective are presented. These results show a significant performance improvement when the number of servers increases.

2.2 Comment 2

“* ”In Preliminary concepts and related work” authors said that PasS is a combination of software and service but it not clarify that there are three layers: Infrastructure as a Service (IaaS) where raw virtual machine are offered (e.g. dual core with 1 GB of RAM virtual machine), then on top Software as a Service (SaaS) where virtual machine with some kind of software installed is offered (then it combines both, e.g. the former with Linux and Apache2 plus Mysql) and finally the Platform as Service (PaaS) where virtual machine with software that are accessed remotely is offered (e.g. the former that provides a WebMail service) In the last one the users see the services but they don’t care about Infrastructure and software needed. So PaaS joins IaaS plus SaaS in order to provide a service entry point (normally by Web) so users only works with the services directly. ”

We have rephrased the paragraph in order to clarify the descriptions.

2.2.1 Solution

The following paragraph of the manuscript has been rephrased.
A cloud system is a network of computer servers that are offered under demand as a service, and they are designed to be scalable and flexible. Cloud systems can be served in three different ways (see Fig. 1). The first layer is Infrastructure as a Service (IaaS), which means offering hardware, storage and physical devices over the Internet; The second layer is Software as a Service (SaaS), which means offering software and hosted applications over the Internet; And as a combination of both, Platform as a Service (PaaS), which means offering the capability to deploy applications created using programming languages, libraries, services, and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure, but has control over the deployed applications [7, 3]. In our case, we are interested in modelling a private cloud system, maintained by one organization/institution, of the SaaS kind, which mainly provides software services to its members or end users, clinicians and patients.

2.3 Comment 3

"* The second paragraph of "Preliminary Concepts and Related Work" explain that in a previous work the cloud system is modelled by M/M/m distribution. Also that M/M/m/m+r has been used by authors and G/M/m - M/G/m - G/G/m could be used. The authors should clarify better how we can detect the use of M/M/m or M/M/m/m+r models. (For future works, it could be an adaptive system that can select the proper model on the fly) */

We have clarified when could we use M/M/m or M/M/m/m+r queue types.

2.3.1 Solution

The following paragraph has been included in the "Preliminary Concepts and Related Work" section in the manuscript.

M/M/m/m+r models can be more suitable when we have a known finite buffer for arrivals. M/M/m models are useful when these maximum connections are unknown or not relevant, and the resulting analysis is not as complex as in the M/M/m/m+r models.

The following paragraph has been added in the "Future work" section.

In our case, if patients can enter the system, a M/M/m system could be used, as we would have no clear reference to the maximum number of users in the system. In the other case, if patients can’t enter the system, we could take the M/M/m + r/K approach because we would have a more specific set of customers. We would want to create an adaptive system that could select the best model for each situation.

2.4 Comment 4

"* It just a question of taste but I prefer ”Front-end and Back-end” rather than ”Front-and Back-end” */
We have modified the phrase.

2.4.1 Solution
The following phrase has been modified in the manuscript.

The architecture of our cloud platform consists of two main parts: Front-end and Back-end [...] 

2.5 Comment 5
“* Page 5, in the Back-end section it said ”…the server and its virtual machines…” so for fault tolerant reasons, some extra servers and virtual machines may be around. Otherwise ”the server” becomes a single point of failure. ”

The model is designed to be used with different physical servers. For testing purposes there is only one physical server with multiple virtual machines in it. We have rephrased the paragraph to state that our model is designed to be used with multiple physical servers.

2.5.1 Solution
The following paragraph has been rephrased in the manuscript.

The Back-end functions include management of the job queue, the servers and their virtual machines and the storage servers with their database system.

2.6 Comment 6
“* Page 6, line 6 on ”Methods, system analysis and design” we found ”(Section ), ” with the missing number of the section. ”

We have corrected the mistake.

2.6.1 Solution
We have changed the incorrect reference.

Taking into account the cloud architecture (Section Back-end)

2.7 Comment 7
“* On page 7, ”Fig. 6shows” (white space missing) and the same problem on section ”Result and discussion” with ”Fig. 7show”, ”Fig. 8show”, ”Fig. 7by” ”

We have added the missing space in all the cases.
2.8 Comment 8

“* Page 7, ”Taking into account” -đ ”Taking into account” Page 8, ”...when the averaged waiting time...” -đ ”...when the average waiting time...” ”

We have corrected the stated sentences.

2.9 Comment 9

“* In the ”Future work” section there are the word ”Furthermore” many times used. For the second one it is better to use ”Moreover”, and for the last one ”In this way, by knowing the hospital users...” ”

We have modified the section in order to avoid the repeated usage of the word ”Furthermore”.

2.9.1 Solution

The following section has been rephrased in the manuscript.

As explained above, we would like to extend Algorithm 1 to determine the value of $T$ based on $\rho$. We would like to run more tests in order to explain how fast can $W$ (waiting time) change and the proposed system reaction to these changes. Furthermore, it would be of great interest to incorporate mechanisms for deciding the type of virtual machines that should be created/deleted (primary or specific servers). Moreover, we would like to replace both queues with a more realistic $M/M/m/m + r/K$ model, with $m$ servers, $m + r$ user connections (the maximum number of users in the system, that is, users receiving the service, being at most $m$, plus users who are waiting, at most $r$), and a maximum number of $K$ users as presented in [7]. In our case, if patients can enter the system, a $M/M/m$ system could be used, as we would have no clear reference to the maximum number of users in the system. In the other case, if patients can’t enter the system, we could take the $M/M/m + r/K$ approach because we would have a more specific set of customers. We would want to create an adaptive system that could select the best model for each situation. As future work, we also plan to develop an application by using OpenStack, which will emulate the requirements of the Tobacco Control Unit in Santa Maria Hospital (Lleida, Spain), using real data based on user numbers and requirements. We have already implemented a preliminary prototype [8]. The aim of this work would be to estimate the computing resources that such a Tobacco Control Unit would require. In this way, by knowing the hospital users, we will design a cloud system applied to e-Health in a specific hospital. This application should be extended to emulate the behaviour of the system assuming the scalability of the system by increasing the number of hospitals. We would also like to extend the scalability tests to more than one hundred servers. We would like to test up to one million servers in order to verify the scalability of the system.
2.10 Comment 10

" * "Future work" should include tasks like: + Not only up 100 servers but 1,000,000 servers in order to demonstrate the real scalability. + It would be nice to explain how fast can W (the main waiting time) change and the proposed system react without problems (check + create the virtual machine warranty that Wmax ¡ W) "

We have extended the "Future work" section.

2.10.1 Solution

The following paragraph has been included to the manuscript.

We would also like to extend the scalability tests to more than one hundred servers. We would like to test up to one million servers in order to verify the scalability of the system.

The following paragraph has been included to the manuscript.

We would like to run more tests in order to explain how fast W (waiting time) can change and the proposed system reaction to these changes.

2.11 Comment 11

" * The Authors information have to be reviewed: - "in the same university" - "in the same University" - "e-Health and parallel simulation" - "e-Health, and parallel simulation" - "His research interest include are" - "His research interests include" - "he is a research associated at" - "he is an associate researcher at" - "His main research interest are information governance and privacy..." - "His main research interests are information governance, privacy..." In general, to many currently (at the present time, at present, nowadays, now, etc.) "

We have reviewed and improved the author’s information.

2.12 Comment 12

" * Figure 4 label ends in "," rather than "," "

The mistake has been corrected.

References


