Survey Paper On Impact Of Cloud Computing on Conventional Software Engineering

Rohan Garg
Computer Engineering Department,
Institute of Technology, Nirma University, Ahmedabad, India
14bce029@nirmauni.ac.in

Prof. Kruti Lavingia
Computer Engineering Department
Institute of Technology, Nirma University, Ahmedabad, India
kruti.lavingia@nirmauni.ac.in

***Abstract*— Since the conception of computers, a varieties of softwares have been created. Each of these follow definite standards and are developed using practices evolved through the years. With the advent of Cloud Computing , it is the demand of time that focus shifts from the conventional software engineering methodologies to the new age techniques which are suitable for the same. Here we analyze the challenges faced by the traditional development techniques and present new software engineering requirements for clouds. We also describe one of the requirement engineering processes proposed for SaaS Cloud Environment and introduce the Triple-I model to mitigate the challenges.**

Keywords - Cloud, Software Engineering, SaaS, Requirement Engineering

I. INTRODUCTION

The field of Computer Science and Engineering has shown immense growth in the past few years. One of the most technologically advanced and believed to be technology of the future is Cloud Computing. Cloud Computing[7] itself is classified in three categories:

- Software As A Service (SaaS)
- Platform As A Service (PaaS)
- Infrastructure As A Service (IaaS)

In the present age, the word Cloud goes hand in hand with Big Data and is used for cross continental businesses and their integration, compute intensive scientific and engineering processes, real time data intensive and monitoring systems such as disaster management, climate predictions etc. The wide reach of cloud and the vast variety of services offered by it is possible because of features[7] like elasticity, on demand resource provisioning, fault tolerance , optimization, transparency etc majority of which is to be managed by softwares . Human intervention would not only be prone to errors but would be time consuming too.

So there needs to be a systematic and formal approach towards developing softwares that not only fulfill the required functionalities mentioned above but also enhance its functioning thereby making it attractive for the customers and leading them to become the technology of future. It would surely be unwise to assume that the elastic, unpredictable, dynamic and on demand nature of the cloud could be satisfied by the traditional software engineering approaches. Software engineering for cloud would require innovative approaches to take into consideration utility based engineering.

II. CHALLENGES FOR CONVENTIONAL MODELS

A. Mobility

As compared to a decade ago, the number of connected devices today is much more and it will continue to grow with the advancing times and upcoming technologies such as IoT. This means that data and the processor will be co-located rather than at separate locations.

B. Sharing

With the increasing connections amongst the devices, the data which earlier used to be static and unused for a longer time has today become dynamic. Its form changes with every new interaction and ownership is changed across devices and users. Its location too changes in the same way.

C. Multitenancy

The degree of controllability by a single entity is of little significance. With multiple processes modifying the data stored at different locations , the access delays increase. Also data is shared between instances in multiple locations.

D. Parallelization

Traditional software models allow work to be completed sequentially. With the advent of distributed computing and large scale systems, it becomes necessary to divide the work in chunks and perform parallel processing to complete it faster.

E. Big Data

The huge and voluminous amount of data is being generated by the computing devices today. This data differs in structure. The velocity at which it is incoming and the amount of processing it requires completely changes the requirements of a software to be developed.

F. Clock Speed

As per Moore's Law the computing power becomes 2x every 18 months. The evolution of amount and type of incoming is faster than that. Once a zenith is reached for processing power, engineers have no option but to utilize the existing resources for the fulfillment of tasks. Same is true in case of budget constraints too.

G. Interaction with Providers

Till date, the requirement gathering had involvement of users and software developers. However for clouds, software development will have to consider heterogeneous platforms, distributed web services and multinational enterprises whose geographical locations should not affect the cloud and its services in a major way. This is possible only if Cloud Service Providers(CSPs) are included in the development process. Questions on cost, component reusability, quality assurance etc. can be answered only by them. The reusability of web services might reduce the amount of code to be written but the complexity would increase for it is to be integrated with other services and documentation may or may not be available. Only coding and testing can be done without interaction with CSP. Maintenance phase would have involvement of CSP as well because it is on their infrastructure that developments will be rolled out. The amount of interaction between the CSP and the engineer would depend on type of cloud : Public, Private or Hybrid. Software engineer will have to solve the conflicts in case of data lock-in involving the CSPs to provide the customer with best results.

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

III. REQUIREMENT ENGINEERING PROCESS FOR SAAS

The following specific modifications have been proposed along with inclusion of CSP as a stakeholder:

A. Architecture Requirement

The architecture should support multi-tenancy. This means that multiple instances of modified data by different users should maintain their states. Consistency and correctness of data should not be affected. The various parameters such as security, privacy, availability, data recovery, sustainability etc will also have to be considered as a part of the Service Level Agreements(SLA). This means that with each different type of service the architecture would slightly differ.

B. Behavioural Requirement

Based on the customization and extension of services demanded by the user the model would change. The on line system should be able to handle the change in the design, performance, security, data migration and its governance and maintain the locality of data as well. This would happen when the system is functioning that is, operational.

C. Manangement Requirement

The pay-as-you-go feature is what makes cloud draw customers to itself. In order to achieve it, specific requirements of billing, monitoring of resources, penalty calculation, virtualization, migration of virtual machines and management of ongoing processes needs to be considered.

D. Implementation Requirements

Along with the CSP, the third party comes in to picture. Here Cloudonomics which deals with implementation costs in clouds as well as maintenance of cloud, its upgrade and testing need to be taken care of. Also if any standards or practices that are required by third to be maintained on cloud need to be considered.

E. QoS and Security

Quality of Service generally includes the non functional requirements such as fault tolerance, reliability, ease of use and interoperability. The security requirements other than usual malware filters, access control and code reviews would include authentication control , identity establishment, user authorization etc.

F. Compliance Requirements

Cloud computing requires servers and data centers to be placed at distant places geographically. The requirement here would be to ensure that all the standards and legal rules and regulations of data governance, security , data migration, SLAs etc are in accordance with the laws and legal frameworks of the particular region while making sure that the functionalities are least affected and customer enjoys all the benefits.

G. User Based to Service Based Requirements

With numerous CSPs in the market and growing number of customers with their varied needs, the need for standard designed specifically for SaaS arises. In a traditional software development cycle, Requirement Elicitation, Analysis, Specification and Verification phases are included where user is involved. To understand the infrastructure of cloud, maintenance demands and management etc CSP will be required. Based on this involvement itself and a few additional steps in CMMI level 2 a significant improvement is seen in reduction of project risk, cost of development, development time and higher return on investment.

IV. THE TRIPLE I MODEL

The European Commission Cloud Computing Experts Working Group proposed the 'Information-Incentive-Intention' model to address the challenges of future IT.

The relationship between the data and the operations that handle it define the software constraints. They define the reusability, adaptability, execution and its distribution. With increased cohesion between the data and the operations , the software becomes difficult to maintain and upgrade.

A. Information:Data Abstraction

Information is processed data. While the data is in raw form, the information should be able to represent the data in any format desired. This can be done by maintaining metadata which is adaptable in nature which can help in various interpretations of the data. Representation of data in form of adaptive information would help to mold the data in a desired shape, decomposing it and even construct the information again from partial data. This shows that even if encoding of data is important, while developing applications for cloud, the specific data structures and operations need to be made adaptive to provide interoperability.

B. Intention:Code Abstraction

Quite similar to the data abstraction, the code which acts on it for processing must be adaptable too. It should be transformable and reusable on any machine irrespective of the underlying architecture. traditional development involved considering the types of machines used by client and then development of code specific to those machines which reduced the burden of developers and provided optimization limited to the clients machines. It is to be understood that the definition or the intention of code is different form its execution. The mapping should not be tedious or else it would affect the performance. The intentions of different codes can be combined and then put into application.

C. Incentive:Goal Maintenance

This phase deals with achieving the business goals. The Incentive step would ensure that optimal reusability , the performance is bound to suffer after a certain limit. With the increasing need for adaptability and re-composition of the code , the complexity to translate those requirements into concrete implementations also increases. Hence while considering the re-usage of code, a number of hidden factors, heuristics and inaccuracies should be considered so that the business goal does not take a hit.

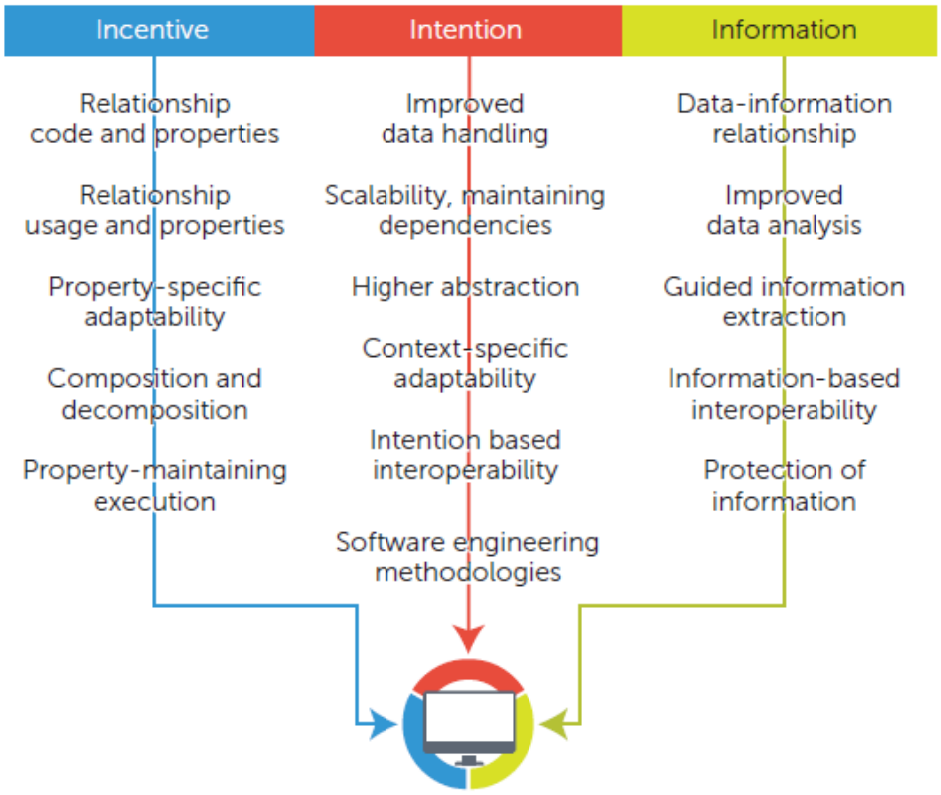

Figure 1. The Software Engineering Roadmap as proposed by the Experts Group[6]

D. The Roadmap to Realization

The Triple I model is just conceptual in nature. Implementing it would mean that the entire traditional software engineering would have to be rethought. this would evidently lead to an economic upheaval, where the systems and softwares we use today are based on the traditional approach. The process needs to be smoothened by:

1. Developing the current IT to be change ready.

2. Gradually moving to the newer models. To ensure the same , the experts group and software engineering experts came together and developed a well defined and clearly expressed Road Map.

V. CONCLUSION

It is clearly seen that with the growing technology, a number of changes are required in the software engineering processes. The conventional techniques though developed through years of implementation and have been made full proof are of little or no significance in the contemporary times. A deeper insight and understanding of the current IT and its paradigms leads to development of unconventional models whose implementation becomes an issue. The solution would be gradual shift from traditional to the non traditional engineering methods and reap the benefits of advanced technology.

REFERENCES

[1] A. Tariq, S. A. Khan and S. Iftikhar, "Requirements Engineering process for Software-as-a-Service (SaaS) cloud environment," 2014 International Conference on Emerging Technologies (ICET), Islamabad, 2014, pp. 13-18.

[2] R. Guha and D. Al-Dabass, "Impact of Web 2.0 and Cloud Computing Platform on Software Engineering," 2010 International Symposium on Electronic System Design, Bhubaneswar,2010, pp. 213-218.

[3] R. Bahsoon, N. Ali, I. Mistrik and T. S. Mohan, "The Fourth IEEE International Workshop on the Future of Software Engineering for/in the Cloud 2014 (FoSEC 2014)," 2014 IEEE World Congress on Services, Anchorage, AK, USA,2014, pp. 232-233.

[4] R. Bahsoon, N. Ali, I. Mistrik and T. S. Mohan, "The IEEE Services Visionary Track on the Future of Software Engineering for/in the Cloud," 2015 IEEE World Congress on Services, New York City, NY, USA,2015, pp. 29-30.

[5] R. Bahsoon, N. Ali, I. Mistrík and T. S. Mohan, "The IEEE Services Track on Software Engineering for/in the Cloud," 2016 IEEE World Congress on Services (SERVICES), San Francisco, CA, 2016, pp. 97-98.

[6] L. Schubert and K. Jeffery, "New Software Engineering Requirements in Clouds and Large-Scale Systems," IEEE Cloud Computing, vol. 2, no. 1, 2015, pp. 48-58

[7] Buyya, Rajkumar, James Broberg, and Andrzej M. Goscinski, eds. Cloud computing: Principles and paradigms. Vol. 87. John Wiley & Sons, 2010