Journal of Computer Science 10 (12): 2408-2414, 2014
ISSN: 1549-3636

doi:10.3844/jcssp.2014.2408.2414 Published Online 10 (12) 2014 (http://www.thescipub.com/jcs.toc)

FEATURE-BASED ANALYSIS INTO THE TREND OF SOFTWARE TECHNOLOGIES FROM TRADITIONAL TO SERVICE ORIENTED ARCHITECTURE AND SAAS CLOUD

Atieh Khanjani, Wan Nurhayati Wan Ab. Rahman and Abdul Azim Abd Ghani

Department of Software Engineering and Information System,
Faculty of Computer Science and Information Technology, Universiti Putra Malaysia,
43400 UPM Serdang, Selangor, Malaysia

Received 2014-04-04; Revised 2014-05-06; Accepted 2014-07-25

ABSTRACT

There are issues of confusion between the concepts of Service Oriented Architecture (SOA) and Software as a Service (SaaS) which affect on the benefits they offer like cost reduction and agility. To solve this problem, the paper aims to explore the concepts of SaaS and SOA in order to give better understanding of these two technologies. Since SaaS cloud is getting more popular day by day and many companies are shifting to apply SaaS solutions, this motivates us to make a clear understanding of the concepts of SaaS delivery model and SOA architecture. Therefore, in this research we have reviewed the concepts and features of both SaaS cloud and SOA and then compared them with the traditional on-premise software.

Keywords: SOA, SaaS, Software, on-Premise, on-Demand, Business, Cloud Computing, Service Oriented

1. INTRODUCTION

The origin of computing resource sharing back to 1960's when the idea of "computation should be organized as a public utility" has been proposed by John McCarthy in 1961. The idea in fact, addressed the concept of cloud computing-a shared resource of computing power. Anyway, while the idea has been around for some time, the web-based technology required to support Software as a Service (SaaS) matured heading into the early 2000 when Sales force company started offering SaaS services like Customer Relationship Management (CRM).

SaaS became a commonly accepted business model after the first SaaS Conference to be offered by SDForum in 2005. According to IBM and Microsoft definition of SaaS as two major SaaS providers in 2007, SaaS functionality is delivered by a subscription model over the Internet that customer rents a total solution remotely delivered. SaaS is impacting software industry and increasingly popular to the users for its simple deployment. SaaS allows providers to serve and support a lot of users only through a single version of product with pay as you go licensing model (Chou and Chou, 2008). A survey on more than 1000 enterprise software decision makers has been performed in June 2010, by Forrester. The results showed that SaaS combined with buyers'desire for solutions that allow them to save costs, deploy quickly and avoid long-term lock-in, is definitely continuing to gain in popularity (AFST, 2014).

Forrester's research also shows that the global SaaS market is expected to surpass US$130 billion by 2020 from US$21.2 billion in 2011 (Collier, 2012). Furthermore, more than 20% of all distribution and manufacturing software was SaaS-based in 2012. This number is expected to grow to 45% within the next 10 years given what companies say they would be willing to

Corresponding Author: Atieh Khanjani, Department of Software Engineering and Information System Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor, Malaysia

consider. In contrast, traditional on-premise deployments are far less utilized than they were in the past; a few years ago the percentage of use was in the 90%, while today it stands at just 50% (Burgess, 2013).

With more applications being delivered via the cloud, there is set to be an explosion of SaaS providers. According to Martin Thompson, 2014 is the year that enterprises begin to claw back control from overbearing software publishers and sloppy licensing and audit practice (CW, 2014). Moreover, according to Gartner research SaaS will grow $22.1bn in market place for 2015 (McLellan, 2013). The revenue of SaaS subscription model is projected to be a quarter of the total new software revenue by 2016, which is a jump from the current 5 to 15% of the total spending in software during the same year (e-Core, 2009).

The rest of paper is organized as follows; section two explains SaaS versus on-premise traditional software. Section three, elaborates service oriented architecture. In section four, we have described the differences and similarities of SOA and SaaS. Section five explains the results and discussion of the paper and finally we have concluded our work in section six.

2. SAAS VERSUS ON-PREMISE TRADITIONAL SOFTWARE

Traditional on-premise software model in which the software is purchased under a lisence for installation on personal computers known as software as a product (TechTarget, 2010). It can be expensive to run so that only large companies may well be able to afford such issues and not smaller businesses which don't have the necessary skills or resources to perform such systems effectively. Therefore, SaaS model came to overcome the problems of running systems due to they are running by the suppliers' side at the hosted data centers without having any requirement for software or hardware on user side. For running a SaaS software we need only a good internet connection and a web browser like Google Chrome, Firefox, Internet Explorer or Safari (Jaffe, 2013). **Figure 1** indicates the representation of SaaS as a utility.

SaaS as the best-known and commonly heard branch of cloud computing aims to provide on demand and plug-in platform like using public utility like electricity. The other technologies which before SaaS were used like 'Application Service Provider' (ASP) are single tenant, more like on-premise (in terms of licensing and architecture) and hosted multiple client-server applications, while SaaS provide a 'multi-tenant' model in which subscribers access the same code base, with their data and any customizations kept separate. Because ASPs are originally constructed as single-tenant applications, their ability to share processes and data with other applications was restricted, so they may offer a few economic advantages rather than their locally installed counterparts (Al-Sahhar, 2009).

Moreover, SaaS is a centralized, configurable, scalable and multi-tenant system which uses meta-data to provide support for different customers with different requirements at the same time. SaaS is changing the way in which people construct, sell, purchase and utilize the software that's why it has significant impact on software industry. On the technical side, the SaaS provider deploying patches and upgrades to the application transparently and delivering access to end users over the Internet through a browser or smart-client application. Providers might even provide tools that allow users to change the workflow, data schema and other aspects of the application's operation for their use. **Figure 2** indicates a sample SaaS architecture.

In fact, SaaS redefines the software deployment model from packaged applications to a dynamic Internet based service relationship which effect on relationship and value proposition between software vendors and both service user and provider. **Table 1** illustrates the differences between on-demand and on-premise software paradigm. The relationship between customer and provider in SaaS is ongoing and much higher than traditional one as the provider offer continuous relevant improvements in the services and features as well. As mentioned before, SaaS is maintained and supported by providers which let the companies relieve a significant amount of time and money spent on technical support. In addition, SaaS providers roll out software updates with new features on a frequent monthly basis as part of a license agreement. Therefore, SaaS usually does not need additional fees for software updates and users can always have the access to the newest version of software with latest features and capabilities.

Meanwhile, upfront costs are considered as a definite drawback of traditional software paradigm as software purchases require a commitment for the entire solution, making first-year costs two to three times more than a SaaS solution. In addition, upgrades or hardware purchases may also be necessary which adds to the costs of ownership. Furthermore, traditional software needs

new buy-in when new versions are released. Using traditional software is necessary to have several versions behind the system upgrades because each upgrade may expose a new bug or database issue, or even make the system down totally. In SaaS solutions if a problem occurs for a customer and then fixed by provider, the problem automatically has been fixed for every one without any upgrade fee (Alsahhar, 2009).

Fig. 1. On-demand SaaS (Derek, 2011)

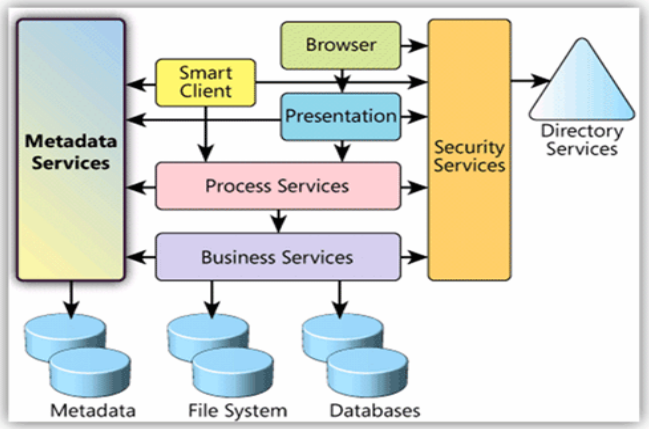

Fig. 2. Sample SaaS Architecture (Bedin and Moinuddin, 2007)

Table1. On-demand Vs. on-premise

Parameters	On-demand	On-premise
Development	Short, continues cycle	Longer cycle, big bang
Delivery	Hosted	Installed
Licensing	Subscription (inclusive)	Perpetual license and maintenance
Allocation	Expensed	Capitalized
Cost	Pay as you go	Upfront capital costs for hardware, software licensing, lab space
Additional cost	Configuration	Installation, maintenance, customization and upgrades
Updates	Shorter and frequent	Larger, less frequent /new buy in
Profits	Ongoing	Initial sale
Hardware	Software and hardware provided by provider side	Customer needs to provide hardware and platform himself to run app
Platform	Single platform (Bedin and Moinuddin, 2007)	Multi version
Data sharing	Easy	Hard or almost impossible
Life style	Internet-based subscription	Install, upgrade and maintain
Tenancy	Multi-tenant (Alsahhar, 2009)	Single tenant
Customer-vendor relationship	Frequently	One or a few times for upgrade only (Claybrook, 2012)
Scalability	Adjust SaaS subscription	additional in-house server (hardware) and software licensee

Compare to traditional software paradigm there are some advantages mentioned for SaaS as follow:

- Easier administration
- Compatibility (having the same version for all users)
- Automatic updates and patch management
- Lower cost
- Better load control techniques
- Easier collaboration
- Quickly deploy
- Global accessibility
- High availability
- High accessibility (access from everywhere through the Internet)
- Reduce power consumption through resource sharing (SaaS, 2010; Santosh, 2014; Merker, 2009)

However, there are also some disadvantages for SaaS as follow:

- New and maybe immature
- Security and customizability concerns
- Need a good connection and network (Santosh, 2014)

Even though SaaS might be new and immature yet but it is soon going to open its place in the industry according to the mentioned researches. However, there are some security concerns for SaaS and since the security is one of the most important issues for customers SaaS vendors employ some of the highest security measures available for their solutions including high encryption, authentication and access features. Moreover, many providers provide flexibility for the users to maintain their own data base server with seamless integration to the SaaS solution. Furthermore, some institution do not trust SaaS provider with their full control over the data and consider it as a risk since the institution loses control.

In terms of customization of SaaS even though at the first days of emerging SaaS customization was limited. However, users has now ability to customize the user interface to change the appearance of the program, as well as modify specific areas, such as data fields, to alter what data looks like. By growing the SaaS markets and getting more matured, providers are investing more in the development to provide more customization and flexibility that companies are accustomed to with on-premise software.

In addition, SaaS depends on a good internet connection and if not a good connection or when disconnected there is a problem for using the service. However, on-premise software also is subject to electrical outages, hardware failures and a range of other risks. But some SaaS vendors put this opportunity to use the functionality in an offline way to help the user keep working in case there is no Internet. Once a solid connection is available again, all the data is synced to the system. At last but not least today, nearly every type of core business function from human resources to enterprise resource planning services is available via SaaS (Derek, 2011).

3. SERVICE-ORIENTED ARCHITECTURE (SOA)

From object oriented in 1980s and component-based development models in 1990s, we came to service orientation model now, which keeps the component-based advantages such as self-description, dynamic discovery, loading and encapsulation. However, in service orientation we have a big change in terms of invoking the objects and passing methods for messages among services. This provides interoperability and adaptability benefits, as these autonomous messages can be sent from one service to the others without considering how the service handles those messages as they are intelligent enough to self-govern the part of their own logic.

Service orientation provides an evolutionary approach to building distributed software that facilitates loosely coupled integration with its inherent scalability and resilience. In fact, Service Oriented Architecture (SOA) presents solutions as services in an architectural model. Services in SOA form a group of independent functions which are more related to the service interface rather than relating to each other. The service in SOA can run on different operating systems, write in any programming languages (java, C#) and various locations (Rahmansyah and Gaol, 2013).

By having these services, SOA's purpose is to be very productive, efficient and agile, surpassing the other existing technology solutions. There are certain standard and common features that are maintained by the services, but they have the ability to be extended and evolved independently. It's possible to combine services so that other services can be created. Only service descriptions make services away of the other services, so that means that they should be thought of as loosely-coupled (AI, 2014). SOA provides support for realizing the advantages of computing and principles that are service-oriented. In SOA architecture, the client switches to another tasks while is waiting for the answer but in traditional client-

server architectures the client should wait idly for its turn to be served (Bedin and Moinuddin, 2007).

4. SAAS AND SOA

The idea of using SaaS, first popped up in the late 1990s in order to allow sharing end user licenses in a way that reduced cost and also shifted infrastructure demands from the company. In the other words, SaaS is able to rent the software usage hosted by a third party instead of buying additional hardware or software to support (Farhat, 2013).

Every service in the cloud including SaaS services is provided to the customers through web services only. Therefore, web services plays a vital role in cloud computing (Nadanam, 2012). In fact, web services are one of the core technological concepts in cloud computing which is rapidly taking shape and growing even though the revolution of cloud may take a decade to fully unfold. Web services make the communication between applications easier so that a software client like web browser can access several applications over one network (Marston *et al*., 2011).

Web services are self-describing and modular components that can be advertised, published, located and discovered through the Internet by using such a standards and protocols as Simple Object Access Protocol (SOAP), Universal Description Discovery and Integration (UDDI) and Web Service Description Language (WSDL). WSDL is a simple XML document which contains a set of machine-readable description of a web service (like operations, messages, protocols for binding these messages and also a network endpoint specification).

A discovery mechanism that helps customer to discover their desire web service is a global and public registry of web services, called UDDI. It is aimed to collect information about the business services in a structured approach and applied for both publishing and discovering the information through provider and client. The information can be classified and found by the using standards taxonomy. Moreover, UDDI provides a schema to define the rules for communicating with the registry by provider and users. SOAP is a sending message protocol to communicate between the applications through the Internet. SOAP is an independent platform- language communication protocol based on XML and using HTTP which can be run on different operating systems (Windows, Linux) with various technologies and programming languages. Since SOAP is not built with programming languages (like Java, C#) and it is only combination of XML and HTTP, so the simplicity of that is one of the advantages to choose it compared to the similar solutions.

Access to services over the Internet has relied on the interaction between a web server and a browser through HTTP protocol. The programmatic way to access to the services over the Internet, called Web services (Khanjani and Wan ab. Rahman, 2013). Web services make the realization of SOA applications possible (Al-Baltah *et al*., 2014). SOA is a design style to build SaaS application using web services. SOA as an architecture can be implemented by web service. The other technologies that can use SOA are corba, REST and (Jabr and Al-omari, 2010; Sabasti *et al*., 2013).

SOA provides a tool to deploy and quickly re-configure as business conditions change the applications and databases owned by a company. Recently, major software providers, deliver products with a SOA design and implementation. SaaS provides a readily accessible means to out-source the applications and databases of a business. According to (Natoli, 2008) for using the benefits which both technologies offer, it is better to incorporate them since it makes sense to in-source as well as out-source some aspects of the business including IT. It is not necessary that SaaS rely on SOA but if the SaaS functionalities use SOA architecture and deploy the SOA, then, the benefits of leveraging these services in business will be huge. Building SaaS on top of the SOA is preferable and makes the application easier to scale. (Nassif and Capretz, 2013).

Table 2 indicates a summary of similarities and differences between SOA and SaaS.

Table 2. Similarities and differences of SOA and SaaS

Similarities	Differences	
	SaaS	SOA
Both reduce costs	Software delivery model	Software design model
Provide more agility	Tactical	Strategic
Consider as IT solution	How doftware deliver	How software structure
Both provide service	Provides business service	Provides small isolated processes as a service
	Offer service to user	Offer service to other applications

5. RESULTS AND DISCUSSION

The first question arise when thinking about SOA and SaaS is that, can we compare these two technologies with each other?

The results of the research indicate that SOA is a manufacturing model which deals with designing and building software by applying the service oriented principles, while SaaS is a model for sales and distribution of software applications. In simpler terms, SaaS is a means of delivering software as services over the Internet to its subscribers, while SOA is an architectural model in which the smallest unit of logic is a service. So, SOA (an architectural strategy) and SaaS (a business model) cannot be directly compared. However, to get the maximum benefits of cost reduction and agility, it is highly recommended that enterprises integrate SOA and SaaS together (Indika, 2011).

One of the resemblances between SOA and SaaS is that both technologies make important effects on cost reductions to better serve existing markets. Online retailers like Amazon.com are filling a huge demand that traditional retailers cannot serve cost-effectively. SaaS customers usually visit, subscribe, pay, customize and use the service all without provider intervention (Bedin and Moinuddin, 2007).

According to (Mohana and Thangaraj 2013), SaaS offers reliable access to software applications to the end users over the Internet without direct investment in infrastructure and software. SaaS is designed to be run for thousands of different customers on a single code while traditional software solution was to be run an individual company in a dedicated instantiation of the software. SaaS can be seen as for “business” of SOA (Santosh, 2014). In fact SOA and SaaS cloud are converging but SOA is an underlying architecture pattern to build the software on it; while SaaS is kind of service delivery model that we use.

SOA in fact decomposes the assets of IT and make them back up as a set of services and then these services can be configured and even reconfigured to business solutions through using processes as the configuration layers (Rahmansyah and Gaol, 2013). The most important reasons we use SOA is agility and reuse. Besides, with emerging cloud computing services can be now outside or inside the enterprise. Since most of the cloud services are used by APIs or services, so cloud is always service-oriented and need data integration which is a core component of SOA and SOA needs data integration strategy but data integration does not require SOA. In fact data integration is a plan or enabling technology for SOA like an engine for a car. So the relationships between them are getting clear now. Data integration is the core mechanism of cloud as well or can become even more important.

SOA and SaaS complement each other. If we consider SOA as a building foundation of architecture, then SaaS (in over all cloud computing) is a set of architectural options that data and processing may reside, when it matches the requirements of the architecture (Linthicum, 2010).

6. CONCLUSION

The people who works with SOA concepts when they come to the SaaS cloud, they misunderstand the concepts (SaaS, 2009). According to (Tang, 2011) compare to object oriented design which was based on reusability for function and SOA architecture based on reusability for business, SaaS is based on the reusability for service. SaaS is supported by SOA. SaaS is software delivery model while SOA is software construction model. This study has been reviewed the concepts of SaaS an SOA and also compared them in de tails with traditional software. Future researches may focus more on the characteristics of the services used in SaaS and SOA, their quality of service and also comparison between other new technologies based on the services.

7. ACKNOWLEDGEMENT

We kindly appreciate Associate Professor Dr. Abu Bakar Md. Sultan and Professor Dr Hamidah Ibrahim, the Deputy Dean of Research and Graduate Studies and the top management of Faculty of Computer and Information Technology, Universiti Putra Malaysia for paying the fee.

8. ADDITIONAL INFORMATION

8.1. Funding Information

We have used the faculty's money/budget for this research and it is not by the grant.

8.2. Author’s Contributions

The main author roughly contributed to the

preparation, writing, editing, development and publishing of the manuscript and co-authors contributed to the development by giving comments/suggestions and publishing the manuscript.

Atieh Khanjani: Writing the draft, editing and correcting.

Wan Nurhayati Wan Ab. Rahman: Result and discussion, comparing SOA and SaaS.

AbdulAzim Abd. Ghani: Title, Abstract, Comparing SaaS and On-premise.

8.3. Ethics

We do not encounter any ethical issue and we have properly reference by citing relevant papers.

9. REFERENCES

Alsahhar, O., 2009. Device Test engineer at AIRCOM international. Telecommunications, University of Texas.

AFST, 2014. SaaS Applications: Security and Interoperability. AFS Technologies.

AI, 2014. SOA Vs. SaaS-what's the difference? Apprenda Inc.

Al-Baltah, I.A., A.A. Abdul Ghani, W.N.W.A. Rahman and R. Atan, 2014. Semantic conflicts detection of heterogeneous messages of web services: Challenges and solution. J. Comput. Sci., 10: 1428-1439. DOI: 10.3844/jcssp.2014.1428.1439

Bedin, W. and M. Moinuddin, 2007. An overview of software as a service in retail.

Burgess, R., 2013. Why the manufacturing industry is shifting to SaaS cloud hosting.

Chou, D.C. and A.Y. Chou, 2008. Software as a Service (SaaS) as an outsourcing model: An economic analysis.

Claybrook, B., 2012. On-premises Vs. SaaS: Making the choice.

Collier, M., 2012. SaaS enablement.

CW, 2014. Predictions 2014: How CIOs plan to spend their 2014 IT budget. Computer Weekly.

Derek, S., 2011. What is SaaS? 10 frequently asked questions about software as a service. Software Advice™, Inc.

e-Core, 2009. Cloud and SaaS solutions. e-Core.

Farhat, T., 2013. Software-as-a-Service (SaaS) as-a-secure-service. Slideshare.

Indika, 2011. Difference between SaaS and SOA. Difference Between.com.

Jabr, M.A. and H.K. Al-omari, 2010. E-learning management system using service oriented architecture. J. Comput. Sci., 6: 285-295. DOI: 10.3844/jcssp.2010.285.295.

Jaffe, L., 2013. SaaS-Software as a Service, 1-4.

Khanjani, A. and W.N.W.A. Rahman, 2013. Concepts and derivatives of web services. J. Comput. Eng., 12: 74-78.

Linthicum, D., 2010. Understanding the intersection between SOA, data integration and cloud computing.

Marston, S., Z. Li, S. Bandyopadhyay, J. Zhang and A. Ghalsasi, 2011. Cloud computing-the business perspective. Decision Supp. Syst., 51: 176-189. DOI: 10.1016/j.dss.2010.12.006.

McLellan, C., 2013. Saas-pros-cons-and-leading-vendors. Oracle Service Cloud.

Merker, L., 2009. Considering enterprise software as a service? University Business.

Mohana, R.S. and P. Thangaraj, 2013. Machine learning approaches in improving service level agreement-based admission control for a software-as-a-service provider in cloud. J. Comput. Sci., 9: 283-1294. DOI: 10.3844/jcssp.2013.1283.1294.

Nadanam, P. and R. Rajmohan, 2012. QoS evaluation for web services in cloud computing. Proceedings of the 3rd International Conference on Computing Communication and Networking Technologies, 26-28, IEEE Xplore Press, Coimbatore, pp: 1-8. DOI: 10.1109/ICCCNT.2012.6395991.

Nassif, A.B. and M.A.M. Capretz, 2013. Offering saas as SOA services. Innovat. Adv. Comput. Inform., Syst. Sci. Eng., 152: 405-414. DOI: 10.1007/978-1-4614-3535-8

Natoli, J., 2008. SOA and SaaS, What's the difference? Intel.

Rahmansyah, R. and F. Gaol, 2013. Service oriented architecture governance implementation in a software development project as an enterprise solutions. J. Comput. Sci., 9: 1638-1647. DOI: 10.3844/jcssp.2013.1638.1647.

SaaS, 2009. SaaS: The future of flexible software model.

Sabasti, I., Siluvai, P. Jawahar, V.and Kumar, S.2013. A framework for simple object access protocol messages to detect expansion attacks for secure webservice. J. Comput. Sci., 9: 308-313. DOI: 10.3844/jcssp.2013.308.313.

Santosh, T., 2014. Software as a service.

Tang, G., 2011. The SAAS architectures and design on the five layers driving model. Manage. Eng., 2: 61-65.

TechTarget, 2010. Software as a Service (SaaS). TechTarget.