

# Software As A Service (SaaS)

Rosnita Binti Abdul Kahar  
Faculty of Information Science and Technology  
Universiti Kebangsaan Malaysia  
Email: akrosnita@gmail.com

## Abstract

*Software as a Service (SaaS, typically pronounced 'sass') is a model of software deployment whereby a provider licenses an application to customers for use as a service on demand. This paper will discuss about the benefits, issues and challenges. The author also mention about the software type, vendor and customers industry that using SaaS.*

## 1. Introduction

When the Internet burst upon the scene in the early 1990s, the concept of software as a service (SaaS) seemed an idea whose time had come. It got hyped along with everything else about the Internet and reached a massive peak of inflated expectations in early 2000 as venture capitalists funded dozens of nearly identical companies that provided various SaaS offerings [1]. As venture funding dried up in mid-2000, the cracks in the SaaS model began to appear.

SaaS is getting a lot of attention these days. The concept of SaaS is not new and has existed for a while. It has been referred to by other names such as Application Service Provider (ASP), Managed service provider (MSP) [2], [3], on-demand services, business process outsourcer (BPO), outsourced IT models [2], cloud computing, utility computing etc [3].

Software as a Service (SaaS, typically pronounced 'sass') is a software thats developed and hosted by the vendor and which the end user customer accesses over the Internet. Unlike traditional packaged applications that users install on their computers or servers, the SaaS vendor owns the software and runs it on computers in its data center. The customer does not own the software but effectively rents it, usually for a monthly fee. SaaS is sometimes also known as hosted software or by its more marketing-friendly cousin, on-demand [4].

SaaS software vendors may let the application to download by the customer on their devices and disabling it after use or after the on-demand contract expires. The on-demand function may be handled internally to share licenses within a firm or by a third-party application service provider (ASP) sharing licenses between firms [1].

## 1.1. SaaS and SOA

Considerable confusion arises in distinguishing between software as a service (SaaS) and service-oriented architecture (SOA). The difference between SaaS and SOA is that the former is a software-delivery model whereas the latter is a software-construction model [5]. SOA basically involves exposing functionality from distributed systems in the form of stateless functions; this is similar to other distributed system architectures such as CORBA and DCOM [3].

SaaS deployments are revenue generating businesses targeted directly at end users, whereas SOA deployments are usually created within IT environments and the services are exposed to other applications as opposed to end users [3]. Author [3] also said that SaaS and SOA are very complementary in nature. In fact, they can't exist without each other. They can be seen in figure 1.

What are the key elements of a SaaS platform? Every SaaS platform has to have a few core things in place, these are: multi-tenancy, ordering and provisioning, user authentication and authorization, service catalog and pricing, service monitoring, SLA management, usage metering, billing, invoicing and payments. Besides these core components, a SaaS platform also needs to support the usual business functions such as marketing, lead tracking, sales, customer support, revenue and financial management, partner settlement, business intelligence etc [3].

Now, let's take a look at the key elements of a SOA platform. A typical SOA platform deployment consists of service producers and consumers from across the enterprise. Service producers publish services via the SOA platform, which get consumed by multiple service consumers. There has been a lot of focus on the technical aspects of a SOA platform e.g. service bus, communication protocols (e.g. SOAP), service interface definitions (e.g. WSDL), service discovery (e.g. UDDI) etc. The importance of service monitoring, management and governance is also well understood, but this is not enough. In a typical large enterprise, the service producers and consumers could be applications or systems belonging to different departments, organizations or even subsidiaries within the enterprise. In such environments, services cannot be produced and consumed informally without proper service management in place since

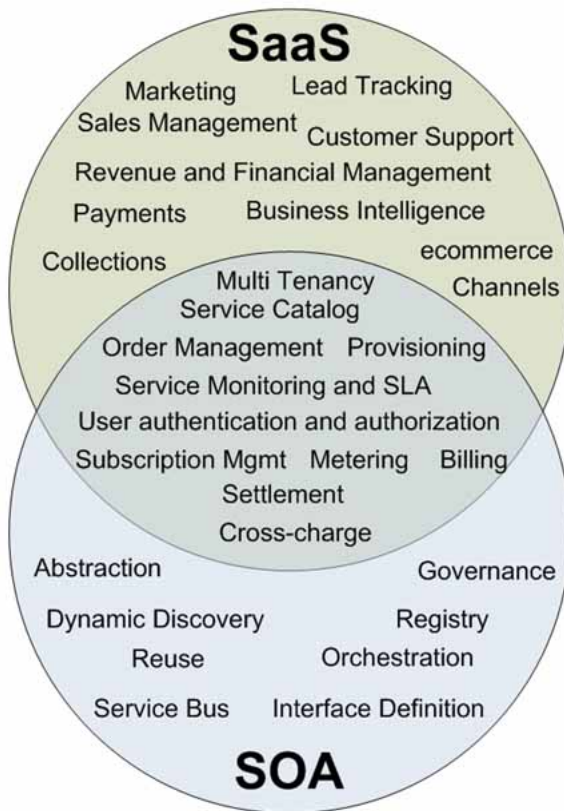


Figure 1. SaaS Vs SOA [3]

there is a cost associated to hosting and exposing a service by the service producer. In order to derive this cost, the total cost of operations or ownership (TCO) needs to be taken into account besides the cost to create the service. Also, there are security concerns around publishing the services openly. This leads to the need for service catalog management, provisioning, authentication, authorization, usage metering and cross-department charging. As highlighted before, these are also the core elements of a SaaS platform. So as an enterprise SOA deployment matures, it is suddenly in need of the core functions of a SaaS platform [3].

Let's take a look at the flip side of this. Every SaaS platform needs to support the ability to add new service offerings and modify existing offerings in the service catalog with minimal changes to the core platform components. These additions or modifications should not lead to creation of a whole new SaaS platform for every service. Instead, all the basic functionalities of the SaaS platform such as ordering and provisioning, authentication and authorization, service catalog and pricing, metering, billing and invoicing, payments etc should be reused for multiple service offerings. Such reuse necessitates the need for a SOA platform. Further, use of a SOA platform enables other advantages such as a more flexible and plug-n-play architecture leading to lower overall cost of ownership [3].

It is probably quite intuitive that most complex architectures including SaaS architecture will benefit from SOA capabilities, but a SOA platform needing SaaS capabilities is not that intuitive. There has been a lot of hype around SOA for a while but most SOA deployments in large enterprises have either not been successful or have not provided the expected ROI because the SaaS elements are missing in these deployments. In order to realize the full benefits of large-scale SOA deployments, it is essential to have a SaaS like service management functionality in place. This is where SOA and SaaS together can enable the concept of "IT as a service" and help take IT to the next natural step in its evolution [3].

## 1.2. How is SaaS different from an ASP?

SaaS evolved from the application service provider (ASP) model. When ASPs sprang up in the 1990s, they offered essentially the same thing SaaS vendors offer today: hosted applications delivered over the Internet. The problem ASPs ran into was that they tried to be all things to all people, and they buckled under the weight of their own infrastructure. In trying to serve the unique needs of each of their customers, ASPs lost the economies of scale that were necessary for them to provide their services in a cost-effective manner [4].

Today's successful SaaS vendors, such as Salesforce.com, LeanLogistics and Ketera, have solved the scalability and reliability problems that dogged the ASPs and ultimately led to their downfall. Instead of trying to be all things to all people, they offer one-size-fits-all solutions. That is, all customers of a SaaS vendor use the same software. The underlying code is the same for all customers and cannot be customized. Any features or functionality that the SaaS vendor adds to the software based on a customer's feedback becomes available to all customers. This multi-tenancy approach differentiates SaaS vendors from the original ASPs and from other vendors of hosted, on-demand software and gives SaaS vendors the economies of scale they need to offer their software cost-effectively and make upgrading their customers to new versions of the software a relative cinch [4].

## 2. The Benefits

For many companies large and small, SaaS is the best way to roll out new technology. The staying power of SaaS has arisen for several reasons:

### 2.1. Low cost of entry.

Instead of paying lots of money to roll out complex technology across the entire company, customers can roll out just one test department of, say, 20 people. The risk is

very low if it fails, and the company doesn't have to involve its busy IT staff [1].

SaaS implementations are also cheaper because companies don't have to buy additional hardware or infrastructure to make the software work, so there are no capital expenditures with SaaS [4].

## **2.2. Less Time Deployment.**

SaaS deployments usually take less time [2] than on-premise software implementations simply because the customers not installing software on every user's computer [4]. Realistically, implementations can take between three and six months depending on the size and complexity of the implementation. Because SaaS is easier and quicker to implement than traditional software, the customer can achieve ROI faster and provided the users adopt the software [4].

## **2.3. The onus is on the vendor.**

If the vendor's software is broken, the vendor won't be getting money from any customer for long. The vendor is motivated to fix the problem [1].

## **2.4. The vendor works for the buyer.**

Customers don't have to rely on their IT departments to install an application. Everything is running securely at the vendor's location [1].

The advantage in this for the vendor is that it spends less time managing compatibility and upgrades across several versions of the software. It also spends less to support customers, as they all use the same version and they don't run it on their own equipment [2].

## **2.5. Easier Access to Current Technology.**

This is because changes are made just to the one code base [2].

## **2.6. Fewer Bugs.**

This is also an advantage for the vendor because having one code base reduces the complexity that can lead to errors [2].

## **2.7. Less-risky investment.**

Instead of spending \$60,000 all at once, for example, customers pay for the software monthly. The monetary risk is lower and less scary [1].

## **2.8. Vendors must provide a secure data environment, or they're out of a job.**

Most vendors understand that data must be backed up religiously, and security is the top priority. Customers' IT departments are typically pulled in many directions and can't be as focused on one technology. Customers can assume their data security is probably safer when it's hosted [1].

## **3. The Vendors**

The example of successful companies are Salesforce.com Inc. [1], [4], Journyx, Webex, GotToMyPC, Concur, RightNow Technologies Inc [1], LeanLogistics [4], Cisco WebEx [3] and Katera [4].

## **4. The Examples**

Some examples of the SaaS are [2]:

- Customer Relationship Management(CRM) Software
- Human Resource
- Procurement
- Document Management

The rank of application used as SaaS depicted in Figure 2. It shows that the largest application using SaaS is Customer Relationship Management (about 18 %), followed by Human Resource (15 %) and Procurement Software (12 %).

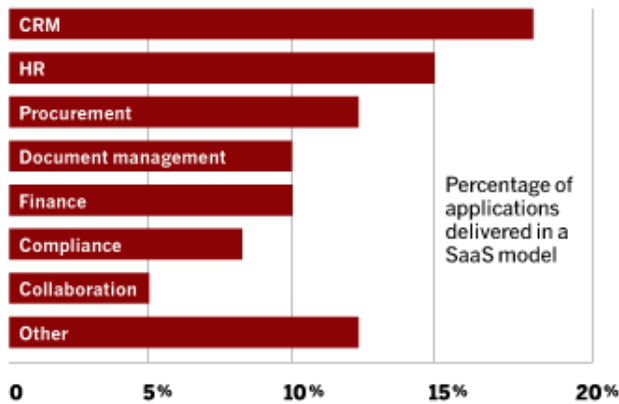
The figure also shows the industrial sectors that used the application. The highest sector is technology industry (18 %), followed by both financial services and utilities (15 %) and the media (12 %). This figure shows that Government is in the sixth place by vertical market.

Figure 2 shows that SaaS is best known in the CRM space, thanks to Salesforce.com's aggressive marketing and its ease of use for salespeople compared to CRM offerings from companies such as Oracle and SAP, aided by the fact that sales groups often have discretion as to the applications they use, notes AMRs Boi. But SaaS also is widely used in the human resources and procurement spaces, both of which have a history of being served by outside firms in a service bureau model. Examples include Concur Technologies expense and travel management software, SuccessFactors employee performance management software, ADP's benefits management software and Ariba's procurement software. SaaS applications also can be found in a wide variety of specialty areas, such as Web analytics, container allocation analysis for shippers and help desk management all of which have histories of being handled by outside service bureaus. Applications in these spaces typically rely on batched data exchange and widely deployed, standard interfaces to internal applications, making SaaS an easy fit, notes Tier1 analyst Mankowski [2].

## SaaS Usage TODAY

The usage of applications delivered as a service fall mainly in three areas: CRM, HR and procurement.

### BY APPLICATION



Technology companies are the biggest users of the SaaS model, followed by financial services and utilities.

### BY VERTICAL MARKET

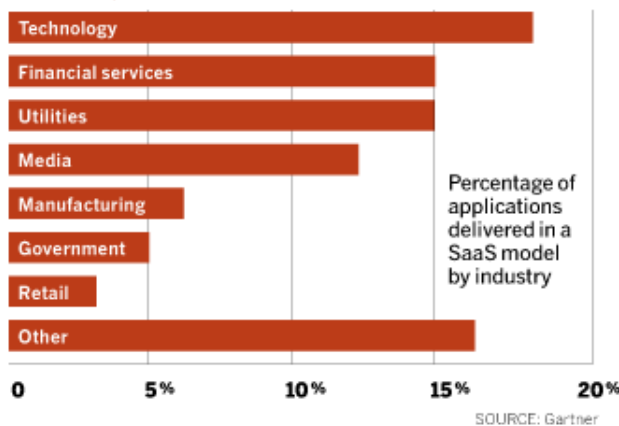


Figure 2. SaaS Today [2]

A third area that has seen broad SaaS adoption is Web conferencing, offered by WebEx, Citrix Online and Adobe. Applications such as Web conferencing and surveying work well with a SaaS approach because they let IT offer users functionality without having to invest in expertise and operations [2].

## 5. Issues and Challenges

### 5.1. Security

The customers have to concerns about keeping their data in a SaaS vendor's systems because they have no direct control over those systems - and because of the horror stories that emerged from the ASP implosion of 2000 and 2001. When those ASPs went out of business, many of them left their customers without access to their data, and some ASPs even sold the data they hosted [4].

### 5.2. Server and Software Maintenance

How and how often the data will backup is an issue to the customers. How long the backup's kept and are the backup stored off-site also the questions that the provider must answer [1]. Others questions are: Is the data center at computer's location staffed 24 hours a day, seven days a week? and What staff is on-site?

### 5.3. Building the Contract

This experience taught customers valuable lessons in negotiating contracts: Always make sure they'll have access to the data and the software or source code in the event the service provider goes out of business [4].

### 5.4. SaaS is not Unique

Because SaaS designed for use by hundreds of other companies, the customer have to consider whether they want to rely on it [2]. The SaaS also make sense if the process is not complex. The customer also have to realized that if they want to success with SaaS they have to accept that everything cannot be customized and it's not unique to specific organization.

With SaaS, there's just one code base for the software, used by all customers, in what's called a multitenant architecture. While the software might be configurable by users to their individual needs, the code itself is the same for all and is not customizable for any individual customer. Any enhancements made based on one customer's requests immediately become available to all customers. So the application isn't one that differentiates the company competitively, such as improving customer service or enabling higher margins relative to competitors [2].

### 5.5. Integration

The convenience of using SaaS applications, especially when adoption is driven by business users, can mask a significant IT challenge. That challenge is integration, both with other enterprise applications and with data sources [2].

## 6. Key Criteria To Apply SaaS

In [4] there are key criteria to decide SaaS implementation to the organization:

- ~ the type of process or function for a SaaS solution
- ~ the extent to the need to customize the SaaS solution
- ~ the extent to which it needs to be integrated with other systems (both internal and external)
- ~ the maturity of the application

There are also questions suggested by [1] that one must ask when dealing with reputable vendor to choose SaaS as a solution:

- ~ How well has the vendor integrated its service operation into its core business?
- ~ How many clients do they have on the SaaS solution today?
- ~ Is the software producer hosting the software for the vendor or have they outsourced that to a completely separate company that basically acts like reseller?
- ~ Where is the computer that will be use to serve the application?
- ~ What data do they backup and how often?
- ~ Is the data center at the computer's location staffed 24 hours aday, seven days a week?
- ~ What physical security measures are in the data center?
- ~ Do they have a disaster recovery plan if your data center becomes unavailable?
- ~ Are servers dedicated to each customer, or do multiple customers share single server?
- ~ How many Internet Service Providers (ISPs) do they buy connectivity from?

## 7. Conclusion

The conclusion from the foundation is using the SaaS will benefit both the customer and vendor. Some of the benefits are it will cut cost for the customers because they do not have to build their own data center and the cost of application development will much more reduced [2]; and also for the vendor for building one data center and standard software for all customers.

There are also issues and challenges such as security, maintenance and uncustomizable of the software to the customers.

Knowing the benefits, issues and challenges so that the organization can refer to the list of things to consider for choosing SaaS as a solution.

## Acknowledgment

The authors would like to thank to Mr Zamri Murah for this Latex application and his kindness teaching all the superb things.

More thanks to Pn Noraidah be the friend and the study team, working hard hand in hand, sharing the knowledge and the effort to be closer to the destiny.

## References

- [1] C. Finch, *All Your Money Won't Another Minute Buy: Valuing Time as a Business Resource*. Lulu. com, 2007.
- [2] G. Gruman. (2007) The Truth About Software as a Service (SaaS). [Online]. Available: [http://www.cio.com/article/109706/The\\_Truth\\_About\\_Software\\_as\\_a\\_Service\\_SaaS\\_](http://www.cio.com/article/109706/The_Truth_About_Software_as_a_Service_SaaS_)
- [3] V. Singla. (2009) The Overlapping Worlds of SaaS and SoA. [Online]. Available: <http://cloudcomputing.sys-con.com/?q=node/1047073>
- [4] M. Levinson. (2007) Software as a Service (SaaS) Definition and Solutions. [Online]. Available: [http://www.cio.com/article/109704/Software\\_as\\_a\\_Service\\_SaaS\\_Definition\\_and\\_Solutions](http://www.cio.com/article/109704/Software_as_a_Service_SaaS_Definition_and_Solutions)
- [5] P. Laplante, J. Zhang, and J. Voas, "What's in a Name? Distinguishing between SaaS and SOA," *IT Professional*, vol. 10, no. 3, pp. 46–50, 2008.