Benefits, Dimensions and Issues of Software as A Service (SAAS)

Dr. Raj Kumar Narwal
Department of Economic and Statistical Analysis, Government of Haryana
Panchkula, India

Dr. Sumitra Sangwan
Asstt. Professor, N.M.Govt. College, Hansi, Hisar, India

I. INTRODUCTION

Software as a service (SaaS, typically pronounced [sæs]) is software that is deployed over the internet. With SaaS, a provider licenses an application to customers as a service on demand, through a subscription or a "pay-as-you-go" model. Saas is also called "software on demand." SaaS vendors develop, host, and operate software for customer use. Rather than install software on site, customers access the application over the Internet. The SaaS vendor may run all or part of the application on their hardware, or may download executable code to client machines as needed—disabling it when the customer contract expires. The software can be licensed for a single user, or group of users.

The world has seen numerous so-called disruptive technologies come and go. Some have had a profound impact on how we run our businesses and go about our daily lives and many have not. Some were long lasting and others were gone in a flash. Software as a Service (SaaS) is proving to have great potential to impact our lives in almost every way. Take for example a small business in Kansas that now has immediate access to a global market by listing its products on eBay. If we talk about a medium scale industry that, due to the cost of the software license and required infrastructure, could never afford a true sales force automation tool, but recently because of the presence of website like salesforce.com which support best-of-breed CRM system for $59.00 per user per month, with no upfront cost Families can now share photos with friends across the country with services like flickr. The list is increasing day by day to accommodate more number of such sites.

II. SAAS APPLICATIONS BENEFITS

SaaS applications are expected to take advantage of the benefits of centralization through a single-instance, multi-tenant architecture, and to provide a feature-rich experience competitive with comparable on-premise applications. A typical SaaS application is offered either directly by the vendor or by an intermediary party called an aggregator, which bundles SaaS offerings from different vendors and offers them as part of a unified application platform.

In contrast to the one-time licensing model commonly used for on-premise software, SaaS application access is frequently sold using a subscription model, with customers paying an ongoing fee to use the application. Fee structures vary from application to application; some providers charge a flat rate for unlimited access to some or all of the application's features, while others charge varying rates that are based on usage.

III. LOWER IT COSTS

When you subscribe to a SaaS application, you avoid the overhead associated with implementing conventional software. A typical software implementation involves purchasing and maintaining servers, housing them securely, and installing and maintaining the software. This requires the time and effort of experienced IT personnel and deflects the efforts of employees at a number of levels away from the core mission of your organization. According to Phil Wainewright, analyst with Summit Strategies and founder of ASPNews.com and Loosely Coupled, the cost of implementing conventional enterprise software is four to five times the cost of the original license.

IV. ECONOMIES OF SCALE

Subscription costs for Service software applications reflect the economies of scale achieved by "multitenancy." Multi-tenancy means that many customers run their applications on the same unit of software. For example, in a SaaS website calendar application, customer data is all stored in one database. This makes the overall system scalable at a far lower cost. If you're concerned that multi-tenancy means losing control of your data see will I lose control of my data?

V. PAY AS YOU GO

When you subscribe to a Service software application, you pay a monthly or annual subscription fee. Compared to a traditional software license, this subscription payment structure works to your advantage. An on-going monthly expense is easier to incorporate into your budget than a large one-time outlay. You can cancel or change your subscription at any time without losing a large initial investment.

VI. SAVE TIME

Because you eliminate many of the typical implementation tasks associated with licensed software and because the software is already up and running on the Service software vendor's data center, deployment time tends to be much shorter with a SaaS application than a traditional one. Rebecca Wettemann, vice president of Nucleus Research, compares deployment times for customer relationship management (CRM) applications: one to three months for a web-hosted application; 18 months for a licensed application.
Focus technology budgets on competitive advantage rather than infrastructure when you subscribe to a web-hosted application, you free your organization from supporting high cost, time-consuming IT functions, including:

- Purchasing and supporting the server infrastructure necessary to install and maintain the software in-house.
- Providing the equipment redundancy and housing necessary to ensure security,
- Reliability and scalability.
- Maintaining a labor-intensive patch and upgrade process.

VII. GAIN IMMEDIATE ACCESS TO THE LATEST INNOVATIONS

With traditional licensed software, you typically have to wait for the next release to benefit from the latest innovations or to move your organization to a new browser or operating system. Given the cost and complexity of moving to a new version, it may not even be practical to upgrade each time a new release becomes available. With a Service software subscription, on the other hand, you benefit from innovations on an on-going basis. As soon as a new or improved feature appears in the application, you can begin using it.

VIII. JOIN A COMMUNITY OF INTEREST

Purchasing a traditional software license is very much an individual affair. When you subscribe to a SaaS application, however, you become a member of a community that has the application at the center. As Wainewright points out, Service software changes the relationship between software vendors and customers. In a service environment, argues Wainewright, there is "a convergence of interest between customer and vendor that's more intimate than that expressed in the world of conventional on-premises applications."
The intimacy results because:
SaaS vendors constantly monitor how their customers are using the application.
Customers easily benchmark themselves against their peers.
Awareness of how customers in the aggregate are using an application presents the vendor with a constant supply of metrics it can't ignore. Typically, this awareness translates into improved usability, performance, and functionality.

IX. DIMENSIONS:

As far as the functionality of the software's are concerned, in the "pure" form of Service software, a provider hosts an application centrally and delivers access to multiple customers over the Internet in exchange for a fee. In practice, however, the defining characteristics between an on-premise application and a Service software application are not binary, but are graduated along three different dimensions: how software is licensed, where it is located, and how it is managed. Each of these traits can be visualized as a continuum, with traditional on-premise software on one end and pure SaaS at the other. In between are additional options that combine aspects of both.

X. LET US DISCUSS THESE DIMENSIONS IN DETAIL.

Licensing:

In software on-premise applications typically are licensed in continuously, with a single up-front cost for each user or site, or (in the case of custom-built applications) owned outright. SaaS applications are licensed with a usage-based transaction model, in which the customer is only billed for the number of service transactions used.

Another model related to it requires customer payment a flat fee per seat for a particular time period - such as a month or a quarter and is allowed unlimited use of the service during that period.

Location:

SaaS applications are installed at the Service software hoster's location, while on-premise applications are, of course, installed within your own IT environment. In between is the appliance model, in which the vendor supplies a hardware/software component as a "black box" that is installed at your location, instead of the vendor's. An example of an appliance in this sense would be a device that includes a logistics application with a cached and periodically updated database. A shipping company might provide such a device to its large customers, so they can query the device for shipping information instead of hitting the shipping company's servers with thousands of individual queries a day.

Management:

Traditionally, the IT department is responsible for providing IT service to users, which means being familiar with network, server, and application platforms; providing support and troubleshooting; and resolving IT security, reliability, performance, and availability problems. This is a big job, and some IT departments subcontract some of these management responsibilities to third-party service providers that specialize in IT management. At the other end of the spectrum, SaaS applications are completely managed by the vendor or SaaS hoster; in fact, the implementation of management tasks and responsibilities is opaque to the consumer. Service-level agreements (SLAs) govern the quality, availability, and support commitments that the provider makes to the subscriber.

XI. ISSUES RELATED WITH THE PURCHASE OF SOFTWARE :

For any given application or function, you can determine your SaaS readiness by plotting your organization's needs and expectations on each continuum. Finding the right place on each continuum involves taking a number of considerations into account, each of which ultimately boils down to a tension between control and cost. Some of these considerations include the following:

XII. POLITICAL ISSUES

Sometimes, the decision can be short-circuited by resistance from within an organization, if important people insist that certain functionality remain internal, under the control of IT; other considerations therefore become unimportant. Test-drive deployments (see the previous subsection titled "Managing the Risks of Software Acquisition") might sometimes help convince risk-averse managers to approve pilot projects.

XIII. TECHNICAL ISSUES.

Service software applications typically provide some flexibility for customer configuration, but this approach has its limitations. If an important application requires specialized technical knowledge to operate and support, or requires customization that a SaaS vendor cannot offer, it might not be possible to pursue a SaaS solution for the application. Another factor to consider is the type and amount of data that will be transmitted to and from the application on a regular basis. Internet bandwidth pales in comparison to the gigabit Ethernet links commonly found in enterprise LANs, and data transmissions that take a few minutes to transfer between servers in your server room might take hours to transmit to and from a Service software application located across the country. Because of this, it might make sense to consider a solution that takes network latency into consideration.

Financial Issues

Consider the total cost of ownership (TCO) of a Service software application, compared to that of an equivalent on-premise application. Although the initial cost of acquiring software capabilities through SaaS is normally lower than that of on-premise applications, the long-term cost structure is less certain. Factors that can affect the TCO of a SaaS application include the number of licensed users; the amount of custom configuration you will have to perform to integrate the Service software application with your infrastructure; and whether your existing data centers already provide economy of scale, thereby reducing the potential cost savings of Service software.

XIV. LEGAL ISSUES

Some industries are subject to regulatory law in different parts of the world, which imposes various reporting and recordkeeping requirements that your potential Service software solution candidates cannot satisfy. Consider the regulatory environments in all the different jurisdictions in which your organization operates and how they affect your application needs. In most of the situations, technical and financial considerations also may

have legal issues, such as whether candidate Service software providers will be able to meet your internal standards for data security and privacy in order to avoid legal exposure. Consider any legal obligations you have toward customers or other parties, and whether SaaS will allow you to continue to meet them.

XV. EFFECT OF SAAS ON INFORMATION TECHNOLOGY

In order to study the basic effects of service architectures on information technology a certain procedure is followed. Let us describe that process in little bit detail. First a decision to start about the SaaS is developed. After you've made the decision to pursue SaaS, the next step is to prepare for the transition by assessing how the deployment will affect your existing information technology assets and by taking steps to ensure that the transition can be handled smoothly.

There are many challenges faced by SaaS. The challenges and their solutions are primarily as follows.

1. Find a leader having good understanding about the SaaS functioning: One that sees customer relationships as ongoing service opportunities, not as one-time transactions. Making sure the leader supports this approach is critical, especially if he or she has a traditional software background.
2. Consider the customer lifecycle for specific organisation. Find out its stages. Determine the key metrics for each stage and build your company's goals around them. For example, if your lifecycle stages range from acquisition to enablement to maturation, the corresponding metrics could range from # of leads generated by free trial to login rates to product adoption and usage.

3 Establish a way to communicate metrics regularly. Recurring status meetings are one way to keep everyone on track and on the same page.

4 Establish clear ownership of key metrics to foster a sense of responsibility and provide accountability. Although metrics can be spread across departments, functions, or teams, they should also be tied to high-level, company-wide metrics and goals. For example, the product management team could own the metrics associated with product adoption, usage, and customer feedback. The metrics for renewal rates should be owned by the sales and marketing teams, rather than a back-office department. Regardless of how you assign metrics, the people who own them must have the power to make decisions that affect them.

XVI. CONCLUSION

In this work we have conducted a critical analysis of the services provided by the software architecture. The Enterprises those are working in this area are expected to do well to take up any issue of flexibility and risk-management implications of providing the additions to the SaaS to their portfolios of IT services. Now in recent times there is a great change in this technology. The service providers for SaaS technology are creating the reuse of modules developed by them. The reengineering concepts are now a days applicable to this area also. There are additional components of software reverse engineering. There is a difference between reverse engineering and reengineering. In the reengineering we use the normal components and repeatedly use it for the development of new system and in reverse engineering we start with the goal and design the system towards the starting. Integration and composition are critical components in your architecture strategies to incorporate Service software successfully as a fully participating member of your service-centric IT infrastructure. As a future enhancement the system can be clubbed with the knowledge management platforms to create a system design that exhibit the characteristics of an expert system. A substantial amount of work remains to be accomplished, as software development hustle to keep up with the ever expanding Web. New media, such as images and video, pose new challenges for search and storage. We have offered an introduction into current search engine technologies, and have pointed to several upcoming new directions. Finally, we believe that the future of enterprise computing is not going to be purely on-premise or in-the-cloud. Instead, like the yin and yang, they will exist in symbiotic harmony. As the acceptance of the Service software model grows among customers, opportunities abound for SaaS vendors, offering great potential at a low risk.

REFERENCES

[1] Do not develop your own online billing solution, Mike McDerment, FreshBooks Blog, August 21, 2006.
[2] Service software vs. Open Source for SMB's? A No Brainer, Zoli's Blog, March 6, 2006.
[3] Seeking the true meaning of Service software, Phil Wainewright, ZDNet Blogs, December 4, 2006.
[4] Software as a Service, Wikipedia, the Free Encyclopedia.
[5] Software as a Service: How far can the pay-as-you-go approach take you? Norm Alster, CFO.com, June 22, 2005.
[6] Software-as-a-Service Myths, Jeffrey Kaplan, Business Week Online, April 17, 2006.
[7] Why You Need A SaaS Strategy. Michael Biddick. Jan. 16, 2010. Retrieved 2010-03-10.
[8] Chong, Fred (October 2006). "Multi-tenancy and Virtualization". http://blogs.msdn.com/fred_chong/archive/2006/10/23/multi-tenancy-and-virtualization.aspx. Retrieved 2008-05-24.
[9] Schuller, Sinclair (March 2007). "Repealing the SaaS Tax". http://itmanagement.earthweb.com/article.php/3663266. Retrieved 2008-05-24.
[10] A B Gartner Survey Shows Many Users are Underwhelmed by Their Experiences of SaaS, Gartner.com, 2009-07-08. Retrieved 2009-09-16
[11] Institute of Directors - Retrieved 8 April 2010

[12] Who does that server really serve? Retrieved 2010-04-03.
[13] R. Song, Z. Luo, J.-R. Wen, Y. Yu, and H.-W. Hon. Identifying ambiguous queries in web search. In WWW '07: Proceedings of the 16th international conference on World Wide Web, pages 1169–1170, New York, NY, USA, 2007. ACM
[14] C. Zhai, W.W. Cohen, and J. Lafferty. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In Proceedings of ACM SIGIR 2003, pages 10–17, 2003