Software & Information Industry Association
the principal trade association for the software and digital content industry

Software-as-a-Service; A Comprehensive Look at the Total Cost of Ownership of Software Applications

A White Paper

Prepared by the Software-as-a-Service Executive Council
September 2006

Table of Content

List of Figures

1 The SaaS Executive Council

1.1 Mission

The SaaS Executive Council is an initiative from the Software & Information Industry Association (SIIA). The Council is a multi-vendor coalition designed for the formulation of best practices, education and communication to help SIIA members and the industry at large better understand the realities and opportunities presented by the "Software-as-a-Service" model. As SaaS continues to gain traction over traditional premise-based software licensing models, the Council will focus its efforts on delivering primary market research and analysis to help software vendors, channel partners and SaaS consumers get the highest value from SaaS solutions.

The SaaS Executive Council has four working group, knows as Committees. These Committees each focus on a specific area of interest to SaaS customers, vendors and partners. This paper was produced by the Marketing Communications Committee.

More information can be found on the website of the SIIA at www.siia.net.

1.2 Contributors

This paper was written by Jan Sysmans of WebEx Communications, with the help of the members of the SaaS Marketing Communications Committee:

- Alf Abuhajleh, Intacct Corporation
- Casey Potenzone, Uniloc
- Diana Waterson, IBM
- Frances Moscoe, Arias Systems
- Fred Hoch, Illinois Information Technology Association (ITA)
- Ileana Rowe, Learning.com
- Jan Sysmans, WebEx, Chairman Marketing Communications Committee
- Michael Rosenblatt, Lionbridge
- Nick Blozan, Opsource, Chairman SaaS Executive Council
- Rob Bernshteyn, SuccessFactors
- Ron Snyder, Breakthrough, Inc
- Scott Blis, Scalable Software
- Tim Clark, The FactPoint Group
- Veronique Buenos, SIIA

2 Executive Summary

According to Gartner, a global IT research firm, the annual cost to own and manage software applications can be up to four times the cost of the initial purchase. As a result, companies end up spending more than 75% of their total IT budget just on maintaining and running existing systems and software infrastructure.[1] With the introduction of computers, companies have accepted this as a cost of doing business. The number of software applications that a company may need are infinite. The resources to operate these applications however are finite.

The Software-as-a-Service (SaaS) revolution allows companies to subscribe to software applications and outsource operating the back-end infrastructure to the SaaS vendor. In most cases, the SaaS vendor can do this much more cost effective; providing overall cost savings for the company. As a result, companies can spread their IT budget across many more applications to support and grow their business operations which will in turn contribute to the bottom line.

This document educates end-users and decision makers on Software-as-a-Service (SaaS), where it differs from traditional software, and what the key benefits are when deploying SaaS applications. In addition, this document also provides the reader with a comprehensive look at the Total Cost of Ownership (TCO) analysis any decision maker should complete before making a choice between a SaaS or a traditional software deployment.

The first few chapters are educational in nature. Readers with a good understanding of the SaaS delivery model can skip these chapters and go directly to chapters 7, 8 and 9 for the comprehensive TCO discussion.

The key cost drivers for any software implementation are the cost of the software application, the hardware required to run the application and the people services required to design, deploy, manage, maintain and support the application.

- Traditional software pricing is limited to the cost of the software application, in most cases an upfront fee in exchange for a perpetual user license. It is up to the customer to determine the cost of the hardware and the people services.
- SaaS applications are charged on a subscription basis. The subscription fee includes the cost of the software application, the hardware and the people services.

This difference in pricing models can make an apples-to-apples TCO comparison "tricky". Software and hardware costs are well understood but the people resources associated with traditional software applications are often underestimated or omitted in a TCO analysis. As a result, the usage driven subscription cost of SaaS applications can seem to be the more expensive solution over a multi-year period. However, when these

[1] Timothy Chou, The End of Software, SAMS Publishing, 2005, page 6

people resources are correctly associated, deploying a SaaS application becomes – in many cases – the more cost effective option.

This white paper helps in better understanding all the different cost factors and includes a simplified calculator that will help influencers and decision makers to better estimate the true TCO of a SaaS versus a traditional software deployment. The ultimate goal of this paper is to educate the reader that in some cases traditional software applications remain the right choice, but in other cases deploying SaaS applications provide a better business case.

3 Definitions

3.1 Traditional Software Model

Traditional software applications are based on a model with large upfront licensing costs and annual evergreen support costs. Important functions of the package are often optioned in a front-loaded licensing arrangement with annual renewal for upgrades and support. Increasing the number of users may raise the base cost of the package due to the need for additional hardware server deployments and I.T. support. Licensing costs are often based on metrics that are not directly aligned with usage (e.g. server type, number of CPUs, etc.).

A typical enterprise software package requires hardware deployment, servers, backup and network provisioning in order to accommodate the number of users on and off-campus. Security architecture is also taxed in an effort to protect this valuable resource from unauthorized access. Traditional software applications tend to be highly customizable. This customization comes at a cost both in dollar and people resources terms.

All on-going maintenance and management of the application is provided by the customer installing the application on its network. Similarly, the customer is also responsible for providing the logical and physical security to the applications as well as offering end-user training and support. Architecturally, the traditional software model is single-tenant.[2]

3.2 On-Demand, Software-as-a-Service (SaaS) Model

On-demand, Software-as-a-Service (SaaS) applications are based on a recurring subscription fee and typically are a pay as you go model. The cost may increase as the usage of the application increases. In the SaaS model, costs are directly aligned with usage (e.g. # named users, # transactions, etc.).

A typical SaaS deployment does not require any hardware and can run over the existing Internet access infrastructure. Sometimes, changes to firewall rules and settings may be required to allow the SaaS application to run smoothly. A SaaS application can be configured using APIs but multi-tenant SaaS applications cannot be completely customized.

The SaaS vendor assumes all the support, training, infrastructure and security risks in exchange for the recurring subscription fees. The SaaS service model is designed to deliver business applications anywhere, anytime which in turn requires the SaaS vendor to employ dedicated support teams and staff that make themselves available to customers on short notice. Along with the personnel comes reserve capacity to handle any spikes in usage, outages or network mishaps and to do this continuously, globally and securely. Architecturally, the preferred SaaS model is multi-tenant.[3]

---

[2] For more information on single-tenant vs. multi-tenant architecture models, please refer to section 3.3

[3] For more information on single-tenant vs. multi-tenant architecture models, please refer to section 3.3

In summary the key characteristics of a SaaS application are:

1. No difference between the license fee and the hosting fee.
2. The application is delivered over a web browser or other thin client.
3. The application is configurable, but not customizable.

3.3 Single-Tenant vs. Multi-Tenant Architecture

The single most important architecture difference between the traditional software model and the SaaS model is the number of tenants supported by the application.

3.3.1 Single-Tenant

The traditional software model is an isolated single-tenant model. This means that a customer buys a software application and installs it on a server. This server only runs that specific application and only for the end-user group of the single customer. Most software applications today are sold this way.

3.3.2 Multi-Tenant

The SaaS model is a multi-tenant architecture model. This means that the physical back-end hardware infrastructure is shared among many different customers but logically is unique for each customer. A good description of the multi-tenant architecture design is: "When a user at one company accesses customer information by using a SaaS CRM application, the application instance that the user connects to may be accommodating users from dozens, or even hundreds, of other companies -- all completely unbeknownst to any of the users. This requires an architecture that maximizes the sharing of resources across tenants, but that is still able to securely differentiate data belonging to different customers."[4]

3.4 SaaS vs. ASP

SaaS solutions are very different from ASP (Application Service Provider) solutions. There are three main reasons for this.[5]

1. ASP applications are traditional single-tenant applications but hosted by a third party. They are client-server applications with HTML front ends added to allow remote access to the application.
2. The applications are hosted by third-parties who ordinarily do not have specific application expertise.
3. The applications are not written as net-native applications. As a result, the performance may be poor and application updates are no better than self-managed premise-base applications.

By comparison, SaaS applications are multi-tenant applications, hosted by a vendor that has all the application expertise and they have been designed as net-native applications that get updated on an on-going basis.

---

[4] Frederick Chong and Gianpaolo Carraro, Architecture Strategies for Catching the Long Tail, MSDN Library, Microsoft Corporation, April 2006

[5] ASP versus SaaS, Wikipedia.org

4 What Are the Requirements that Drive the Acquisition of a New Software Application?

4.1 Introduction

Before looking at the factors that contribute to the Total Cost of Ownership (TCO) of software applications, it makes sense to look at the requirements companies have when buying these applications.

4.2 End-User Requirements

End-users are most concerned with the ease of use of software applications. Ease of use requirements may include: how intuitive are the applications, how quickly can end-users master the application and how does the application simplify the day-to-day tasks of the end-user.

The end-user is not concerned with how the application is delivered (traditional software or SaaS) as long as it is user friendly and is available whenever and wherever the end-user needs access to it.

4.3 Business Requirements

Business requirements are all about solving business problems. Does the application solve the needs for the business unit? Does it fit into the critical business process? How quickly can it be deployed, how are team members trained and supported and does the cost fit into the budget?

Just like end-users, business units are unconcerned about the method by which the application is delivered as long as it solves the business problems, becomes a dependable part in the business process and fits within the budget.

4.4 Company/Corporate Requirements

Corporate requirements are either driven by a need to increase revenue or to reduce or contain costs. These requirements trump both the end-user and the business unit requirements. If the sponsor for an application cannot build a business case that shows the impact to revenue and cost, most likely there will be no approval at the corporate level.

Just like end-users and business units, the company is often unconcerned about how the application is delivered, as long as the application contributes to the top and bottom line.

4.5 Operational and IT Requirements

When companies buy software applications, they traditionally look to the IT department to support and maintain the application. For this reason, IT departments believe they need to control as much as possible the end-to-end delivery of the applications.

As a result, many IT departments tend to be biased towards traditional software applications because of the belief that they relinquish control over the application by buying an application leveraging the SaaS delivery model. They may only want to share control with other departments (both internal as well as external) as long as these groups can prove that they can do as good or better job running the application.[6] Even then, some IT departments may still want control due to factors unrelated to TCO or end-user satisfaction.

[6] Most SaaS vendors meet this very high standard, since they are the application experts and have the scalable processes and infrastructure in place to support and train many end-users.

5 Core vs. Context Software Applications

In his book, *Living on the Fault Line, Revised Edition*, Geoffrey Moore makes the case that companies should only focus on core activities and outsource all other activities. "For core activities, the goal is to differentiate as much as possible on any variable that impacts customers' purchase decisions and to assign one's best resources to that challenge. By contrast, every other activity in the corporation is not core, it is context. And the winning approach to context tasks is not to differentiate but rather to execute them effectively and efficiently in as standardized a manner as possible."[7]

In addition Moore says that "there is no context task that cannot become someone else's core task" [8] To the question why other people can do a better job at a company's context tasks, Moore's answer is simple; "this is where they are putting their A team. It's context to you but is core to them." [9]

This has tremendous impact to understanding the dynamic between traditional software and SaaS applications. Most software applications are context to most companies. They do not provide any differentiation to the mission of the company. However these applications are core to the mission of the SaaS vendors. The underlying message in Moore's definition for software applications is that companies can become much more efficient and effective if they focus their internal software deployment on core tasks only and outsource their context software tasks to vendors who specialize in that area. Once IT organizations accept this paradigm shift, they can become productivity heroes to the agile companies of the 21st century.

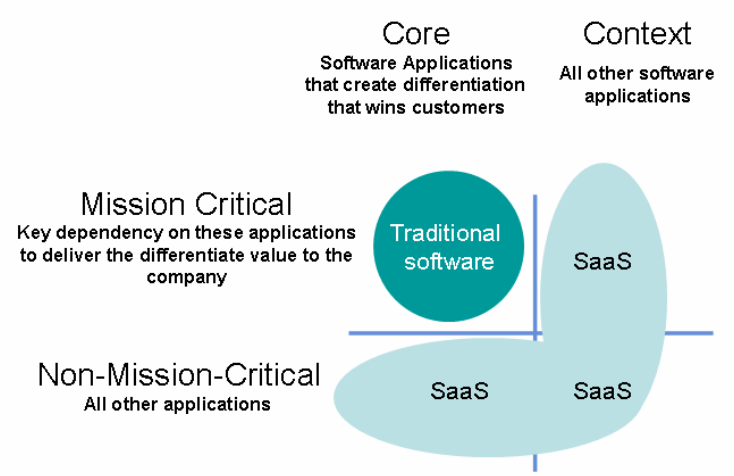

Figure 1: Geoffrey Moore's core vs. context grid adopted for the software industry

[7] Geoffrey A. Moore, Living on the Fault Line, Revised Edition, HarperCollins, 2002, page 26
[8] Geoffrey A. Moore, Living on the Fault Line, Revised Edition, HarperCollins, 2002, page 31
[9] Geoffrey A. Moore, Living on the Fault Line, Revised Edition, HarperCollins, 2002, page 31

6 Key Arguments in Favor of SaaS Applications

There are many arguments that can be made in favor of SaaS applications but there are four key arguments that make a surprising but resounding case for deploying SaaS applications. These four arguments are:

1. Making the IT budget go further
2. No underestimation of people services
3. SaaS allows better growth management
4. Accountability of the SaaS vendor

6.1 Making the IT Budget Go Further[10]

Note: this section was taken from the Architecture Strategies for Catching the Long Tail paper published by Microsoft in April 2006

The first argument in favor of SaaS applications is that SaaS provides a direct and quantifiable economic benefit over the traditional software model.

6.1.1 Cost allocations

In a typical organization, the IT budget is spent in three broad areas:

- **Software.** The actual programs and data that the organization uses for computing and information processing.
- **Hardware.** The desktop computers, servers, networking components and mobile devices that provide users with access to the software.
- **People Services.** The people and institutions that ensure the continued operation and availability of the system, including internal support staff, professional services consultants and vendor representatives.

Of these three, it is the software that is most directly involved in information management, which is the ultimate goal of any IT organization. Hardware and people services, though vital and important components of the IT environment, are properly considered means to an end, in that they make it possible for the software to produce the desired end result of effective information management. (To put it another way, any organization would gladly add software functionality without extra hardware if it could do so effectively, but no organization would simply add hardware without an anticipated need to add software as well.)

In an IT environment based around premise-based software, the majority of the budget is typically spent on hardware and people services, leaving a minority of the budget available for software.

[10] Frederick Chong and Gianpaolo Carraro, Architecture Strategies for Catching the Long Tail, MSDN Library, Microsoft Corporation, April 2006

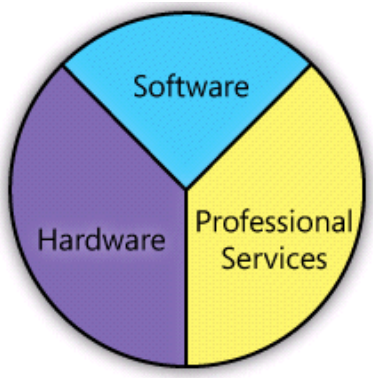

Figure 2: Typical budget for a premise-based software environment

In this model, the software budget is spent primarily on licensed copies of "shrink-wrapped" business software and customized line-of-business software. The hardware budget goes toward desktop and mobile computers for end-users, servers to host data and applications, and components to network them together. The professional services budget pays for a support staff to deploy and support software and hardware, as well as consultants and development resources to help design and build custom systems.

Note: The proportions shown in these diagrams are for illustrative purposes only; they are not intended to advocate any specific allocation of resources, and your allocation may differ significantly.

In an organization relying chiefly on SaaS, the IT budget allocation looks much different.

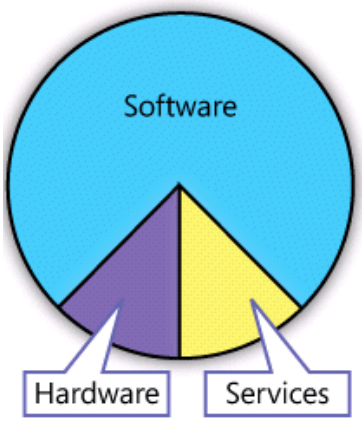

Figure 3: Typical budget for a SaaS environment

In this model, the SaaS vendor hosts critical applications and associated data on central servers at the vendor's location, and it supports the hardware and software with a dedicated support staff. This relieves the customer organization from the responsibility for supporting the hosted software, and for purchasing and maintaining server hardware for it. Moreover, applications delivered over the Web or through smart clients place significantly less demand on a desktop computer than traditional locally-installed

applications, which enables the customer to extend the desktop technology lifecycle significantly. The end result is that a much larger percentage of the IT budget is available to spend on software, typically in the form of subscription fees to SaaS providers.

6.1.2 Leveraging SaaS Economies of Scale

But isn't this result just an illusion? After all, a percentage of the subscription fees paid to SaaS vendors for "software" have to pay for hardware and professional services for the vendor. The answer lies in the multi-tenant architecture and the economies of scale this architecture model offers.

A SaaS vendor with *x* number of customers subscribing to a single, centrally-hosted software service enables the vendor to serve all of its customers in a consolidated environment. For example, a line-of-business SaaS application installed in a load-balanced farm of five servers may be able to support 50 medium-sized customers. This means that each customer would only be responsible for a tenth of the cost of a server. A similar application installed locally might require each customer to dedicate an entire server to the application—perhaps more than one, if load balancing and high availability are concerns.

This represents a substantial potential savings over the traditional model. For SaaS applications that are built to scale well, the operating cost for each customer will continue to drop as more customers are added. As this is happening, the provider will develop multi-tenancy as a core competency, leading to higher-quality offerings at a lower cost. Therefore, even accounting for the hardware and professional services costs incurred by SaaS vendors, customers can still obtain significantly greater pure software functionality for the same IT budget.

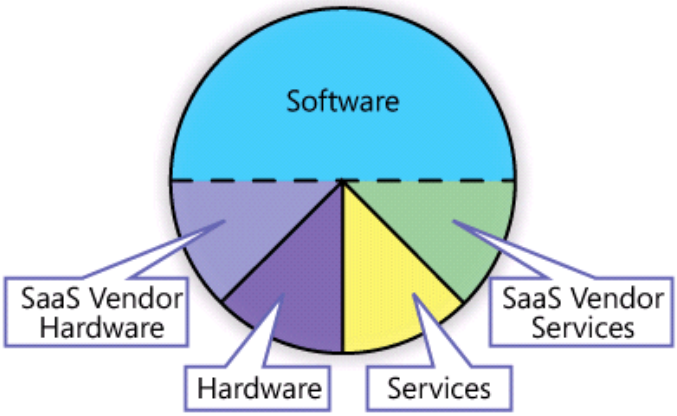

Figure 4: Typical budget for a SaaS environment (accounting for hardware and people services costs)

6.2 No Underestimation of People Services

The second argument in favor of deploying SaaS applications are the underestimated people costs associated with running premise-based traditional software applications. We briefly touched on this is the previous section, but it is worth looking at some industry and analyst numbers in more detail.

Calculating the real cost of people services is not easy and as a result is sometimes omitted from the TCO analysis. This can lead to an apples to oranges comparison. To quote a prospect from one SaaS vendor: "We do not know how to estimate our people services, so we choose not to consider them when comparing the cost of WebEx vs. the premise-based software alternative".

The META Group, a consulting company, determined that "in today's economic environment, even minimal cost savings per seat are tantamount to freeing up discretionary IT investment dollars that might be applied in the enterprise technology portfolio or elsewhere in the organization. Companies no longer have the luxury of looking solely at hardware and software procurement costs and run rates of their technology investments but must examine the purchase decisions across their life cycle as well as how their people are spending their time servicing the application. While companies understand and scrutinize the cost of software and hardware very well, personnel costs are usually not examined as closely as they should be. Examining all these cost factors as a whole and how they impact the total cost of ownership (TCO) is paramount in running an efficient organization."[11]

6.2.1 Personnel Costs Can Be as High as 75%

Gartner Inc, a global analyst firm tracking the high tech market estimates "that more than 75% of the IT budget is spent just maintaining and running existing systems and software infrastructure"[12]. In addition, Gartner believes that "customers can spend up to four times the cost of their software license per year to own and manage their applications"[13].

Another data point comes from Microsoft, which in 2002 told the Wall Street Journal: "that the initial purchase is usually only 5% of the total cost of owning and maintaining a program."[14]

IDC, another global analyst firm, came to a similar conclusion when it did an analysis of the web conferencing industry. It determined that "hidden personnel costs can be as high as 70% of the total cost to run premise-based conferencing software".[15]

---

[11] Messaging Total Cost of Ownership: Microsoft Exchange 2003 and Lotus Domino in Small and Medium Organizations, META Group, July 2004
[12] Timothy Chou, The End of Software, SAMS Publishing, 2005, page 6
[13] Timothy Chou, The End of Software, SAMS Publishing, 2005, page 7
[14] Microsoft Wages Campaign Against Using Free Software, The Wall Street Journal, December 9, 2002
[15] Robert Mahowald, Do Service Providers Deliver Value and Reduce Enterprise Costs?, IDC, 2003

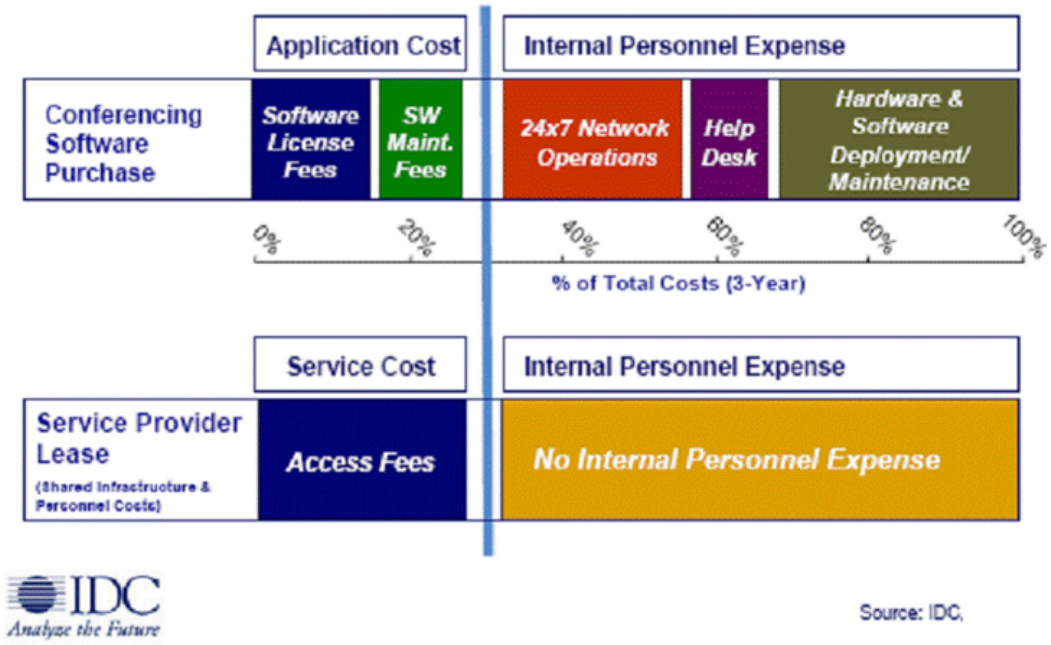

Figure 5: Personnel costs associated with running premise-based conferencing software

Companies seem to instinctively understand this as most have outsourced their audio conferencing applications to audio conferencing vendors. Since running audio bridges is not a core business for most companies, they don't think twice about outsourcing these applications to audio conferencing vendors like AT&T, Verizon, Intercall and BT.

6.2.2 E-mail and Groupware Applications Costs Support This Analysis

By virtue of the extensive experience that enterprises have with e-mail and groupware, the economics of these services are well understood. Analyst reports show that "the personnel costs for these traditional software applications are at a minimum 2.5 times, and can be as much as 7.5 times, the software costs (including maintenance), with a typical range of personnel expenses being 5 to 7 times the software and maintenance costs over a 3-year period".[16]

[16] MultiMedia Communications, A Detailed Analysis of "On-Premise vs. Service Provider" Costs and Risks, WebEx Communications, 2005

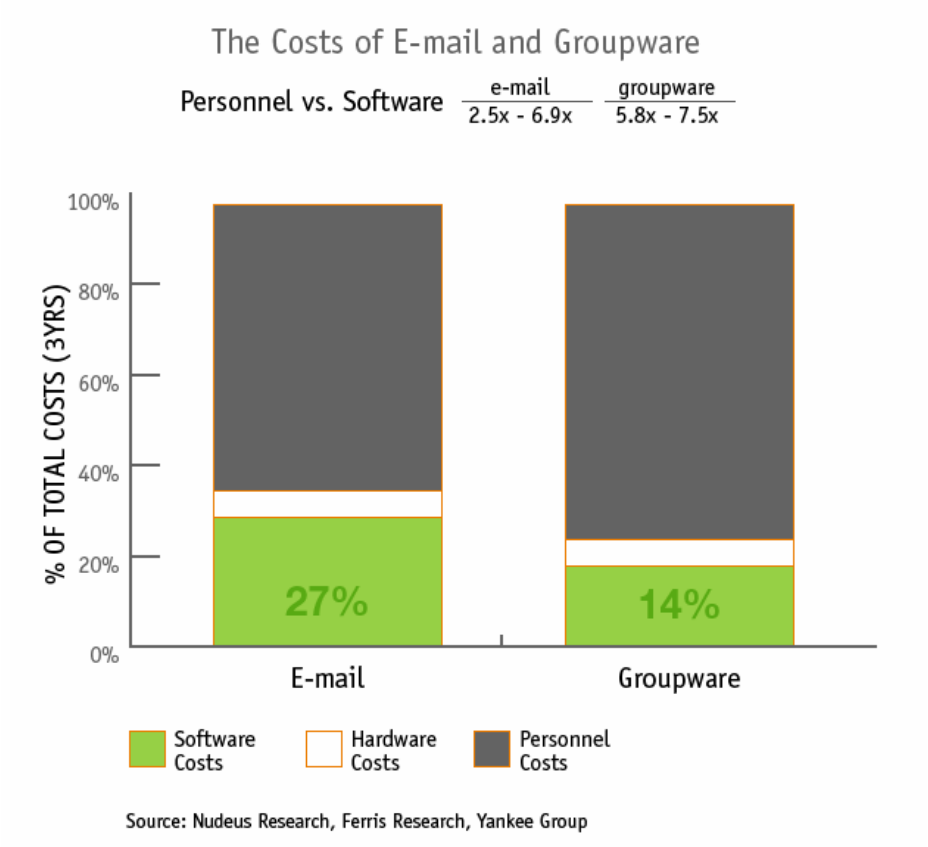

Figure 6: Personnel costs are the biggest TCO part for email and groupware

6.3 SaaS Allows Better Growth Management

The third argument is that SaaS applications grow with you as your business grows. Companies do not have to make decisions on the type of application they need restricted by their own size, but instead they can make decisions based solely on their business needs.

Because of the economies of scale offered by the multi-tenant architecture, SaaS vendors can provide enterprise grade applications in any number of user levels. For arguments sake, let's use the "named user" licensing model to explain this. A named user license model means that a specific end-user has the right to use the application. For example; if a company has 100 employees that need access to a specific application; they would buy 100 named user licenses.

The company does not have a need to roll out the application to all 100 employees at the same time. With a traditional software model, the company would need to deploy hardware infrastructure to support the application and train its IT staff to install, maintain and troubleshoot the application. In most cases it does not make sense to do this to support only 5 or 10 employees. As a result, the company may buy all 100 licenses up front.

In the case of the SaaS application, there is no hardware infrastructure to acquire or IT staff to train. This means that the company can start by purchasing just 5 or 10 licenses, and buy additional licenses as the need grows. This is especially important when budgets and resources are tight. As a result, the SaaS application is much friendlier to the company's growth model when compared to the traditional software application.

There is another important use-case for this flexibility. Companies that are going through mergers and acquisitions many times have a very hard time aligning their back-end IT infrastructure. It can take many months for hardware to be deployed and networks to be built before users in a new location can have access to all the services used in all other locations. This problem is much smaller for SaaS applications, since all that is necessary is an Internet connection, browser and a username and password to start using the application.

6.4 Accountability of the SaaS Vendor

The fourth argument is that SaaS vendors have greater accountability because of the subscription based pricing model.

SaaS customers can actually exert more control over their vendors than traditional software customers. SaaS customers pay a recurring subscription fee for the duration of the contract term. SaaS vendors are typically held to monthly service level agreements (SLA) and are financially motivated to maintain adequate support and operational requirements on a recurring basis. Traditional software vendors are paid a big upfront license fee in exchange for a perpetual license. They have few obligations after the software has been deployed.

As a result, there is much more accountability from a SaaS vendor than from a traditional software vendor. If the SaaS service does not function properly, customers, by the simple act of withholding their payment and enforcing their SLAs, can exert much greater pressure on the SaaS vendor to provide a fix for the application than on a traditional software vendor.

7 Cost Drivers in the TCO Analysis

What are the cost drivers companies need to look at when completing a TCO analysis?

7.1 Capital Expenses

7.1.1 Traditional Software

Software and hardware, network infrastructure enhancements, monitoring and testing tools, security products, supplies, facilities and other required infrastructure are part of the typical capital acquisition expenditure. In many cases, upgrades to other infrastructure may be required which adds additional capital expense. This capital expense is an up-front cash outlay.

7.1.2 Software-as-a-Service

With SaaS, there are no perpetual software licenses to buy. The nature of SaaS is that you pay for what you use. Most SaaS models have a recurring cost structure. You pay a monthly or annual service fee for as long as you use the service. This service fee typically includes maintenance, support, training and upgrades and is inclusive of all hardware, networking, storage, database, administration and other costs associated with SaaS delivery.

7.2 Design and Deployment Costs

7.2.1 Traditional Software

Staff and contract labor needed to research, design, integrate, test, tune and launch is a significant cost associated with deploying an in-house solution. Server and network capabilities must be reassessed and augmented. End-user computer hardware, operating systems and applications have to be evaluated for compatibility with the selected server product and upgraded if necessary. System testing and tuning are necessary to make sure performance is acceptable for launch. Training for end-users and IT staff will be required. Launch activities, awareness and pilots all require IT resources.

7.2.2 Software-as-a-Service

Most SaaS applications can be deployed and put into production much faster and for a fraction of the cost compared to a traditional software solution. This is very important when the opportunity costs of getting the application out are high. On the flip side, because a SaaS application is a multi-tenant application, there are less ways to customize the application to fit the business process.

7.3 Ongoing Infrastructure Costs

7.3.1 Traditional Software

For ongoing operation, network monitoring and management tools are often required to enable real-time problem diagnosis and responsiveness. Additional networking equipment and bandwidth may be needed to accommodate incremental traffic that cannot

be efficiently managed on the internal network. Yearly software maintenance and support contracts and system upgrades make a large contribution to the total cost of ownership. Capacity increases, multiple redundant systems, and add-on feature sets further increase cost. Hardware repair and replacement and recurring environmental costs, such as specialized high-availability facilities and power consumption, add further to the ongoing cost. While these expenditures are spread out over the lifetime of the service, they must be considered in a full TCO analysis.

7.3.2 Software-as-a-Service

Other than additional Internet bandwidth needs, there are almost no incremental infrastructure costs to handle the growth of a SaaS application. Depending on the SaaS application, the IT organization may also have to deploy a desktop application to allow the end-user to communicate with the application. Finally some API (Application Program Interface) development may be required to configure the application to better integrate with existing enterprise applications.

Scaling the infrastructure and the costs associated with growth are fully the responsibility of the SaaS provider.

7.4 Ongoing Operations, Training and Support Costs

7.4.1 Traditional Software

IT organizations will have to allocate resources for monitoring, supporting and maintaining the application. If the application is new to the company, the IT organization will have to train and certify existing personnel and/or recruit new personnel with or without pre-existing application certification. The IT organization will also be responsible for monitoring and maintaining the application and trouble shooting the application in case of downtime. In addition, every time a patch or upgrade needs to be deployed, additional IT resources will be required. This is typically the biggest hidden cost that needs to be considered when making the buying decision for a new application. If this cost is incorrectly estimated, any affect on revenue or cost reduction can greatly change.

Initial and ongoing training is another success factor for the broad adoption of a new application. A vendor may offer initial training or a train the trainer session as part of the upfront cost, but with most traditional software applications it is an internal department that is tasked with the initial and ongoing end-user training. Again, incorrect estimates can greatly affect expected revenue or cost reduction.

Support is the final and most critical success factor to the successful adoption and ongoing use of a new application: whenever end-users have problems with the application, this can lead to a loss in productivity or in the worst case a refusal to use the application all together. This is also known as software turning into shelfware. User issues typically grow with usage and as such the support load in the IT organization grows as well. If you add external use of the application to this equation, support issues

and the costs associated with this typically grow exponentially. Again, if this cost is wrongly estimated, any affect on revenue or cost reduction can greatly change.

7.4.2 Software-as-a-Service

SaaS vendors are responsible for the end-to-end delivery of the application. The only responsibility of the IT organization is to make sure that the necessary ports on the firewall are open and that there is enough Internet access capacity available to allow the end-user base to communicate with the application.

SaaS is a recurring service, so for a SaaS vendor the sale does not end when the initial contract is signed. If a customer does not use the application, they can simply choose not to renew the contract at the end of the contract term. This is called churn. A traditional software vendor does not have to worry about churn, since customers buy upfront perpetual user licenses. As a result, SaaS vendors have a vested interest in seeing customers widely adopt and use the application. It is for this reason that almost all SaaS vendors focus on making their products easy to use and offer initial and ongoing end-user training, and this training is in most cases included in the service fees.

Finally, SaaS vendors also offer 1st, 2nd and 3rd line of support to their customers. This is for the same reasons they offer ongoing training services; if customers churn because of training or support issues, it will have an immediate impact to the SaaS vendor's bottom line.

7.5 Intangible Costs

While the intangible costs are harder to measure and therefore are more difficult to include in a TCO analysis, they are no less real. Some of the intangible cost factors that affect TCO include:

- Reliability and Availability: Failed interactions mean lost employee time and lost opportunities, and may require repeat efforts to persuade users to retry the technology with increasing resistance. What service level agreements (SLA) does the SaaS vendor offer and how do they compare to the internal SLA the IT organization offers?
- Interoperability: How easy is it to integrate with other applications?
- Extensibility: How easy is it to customize the application to fit the needs of the organization?
- Security: The costs of a security breach can be catastrophic if confidential business information is stolen or made available to competitors. What are the security policies that are in place at the SaaS vendor and how do they compare to the internal policies? [17]
- Scalability: As user needs grow, the original system may not keep up. "Busy signals" or functional limitations consume employee time and mean lost

[17] Many companies have very strict policies with regards to the distribution and storage of company sensitive and confidential information. As a result, they are wary of "handing over" the security and control of their data to SaaS vendors. For this reason, SaaS vendors have vigorous security and change control policies in place. For more information regarding security policies of SaaS vendors, please refer to the Security Best Practices White Paper published by the SaaS Executive Council.

opportunities. How well can the SaaS vendor accommodate growth and what are the costs associated with growing the internal application?

- Capacity: Usage and adoption within the enterprise is difficult to predict, making managing capacity difficult. The tradeoffs are poor performance on the one hand or underutilized infrastructure on the other. With SaaS this is more easily managed when compared to an internal application.
- Opportunity costs: The human resource and capital expenditures required by an in-house implementation come at the expense of other projects or could possibly delay the roll-out of new products and services, both of which have a direct impact on the company's bottom line.

7.6 Summary

7.6.1 Traditional Software

The biggest TCO factor of premise-based traditional software applications is the cost of the ongoing people resources that are needed to monitor, maintain and upgrade the application and to provide training and support to the end-user base. These costs are not quoted as part of the cost of deploying the traditional software application and depending on the application, can be between 50 and 85% of the total cost of ownership for the application.[18] Underestimating these costs can have a great impact on the overall TCO predication.

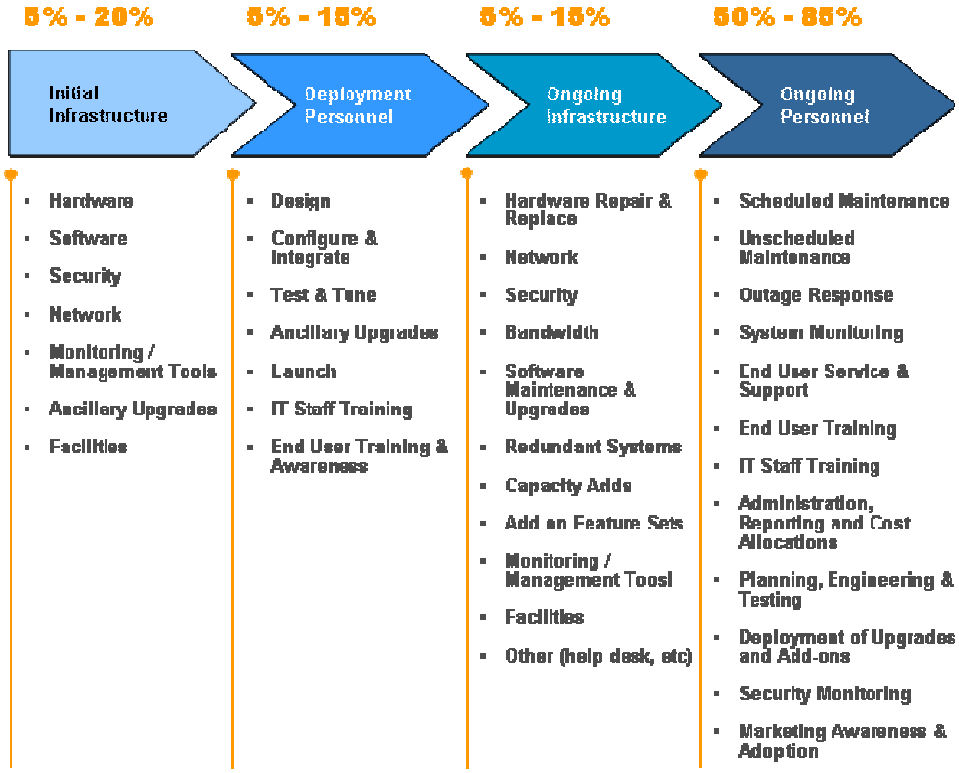

Figure 7: Summary of the cost allocations of a traditional software deployment

[18] For more information and sources for these numbers, please refer to section 6.2 of this paper

7.6.2 SaaS

The biggest TCO factor of SaaS applications are the subscription fees charged by the SaaS vendor. These fees are all inclusive all include the monitoring, maintenance and upgrades to the application as well as the training and support of the end-user base. Compared with the ongoing personnel costs of the traditional software application these costs are quoted as part of the cost of deploying the SaaS application.

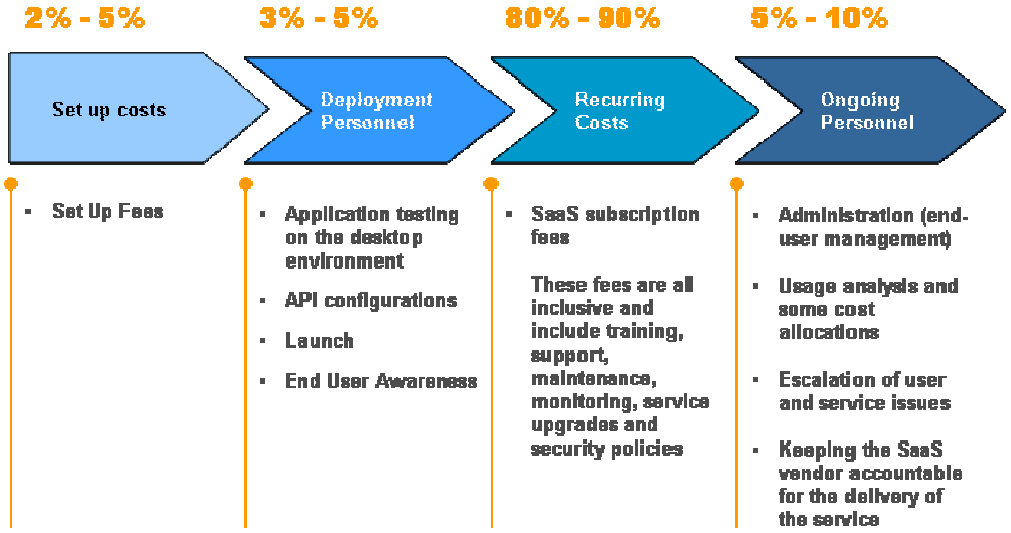

Figure 8: Summary of the cost allocations of a SaaS deployment

8 The TCO Calculation

8.1 Moving Away from "Owning" Software

With traditional software, customers buy a perpetual user license. This gives them the impression that they "own" the software and can use it at will and in perpetuity. With SaaS, instead of "owning" software, customers pay for a subscription to software running on the infrastructure owned by the SaaS provider. The customer's right to use the software goes away once they stop paying for the subscription. They do not loose the rights and ownership of data stored on the infrastructure of the SaaS vendor.

The dynamic between subscription and ownership is used as a TCO argument in favor of buying traditional software. Why rent when you can buy, especially when the plan is to use the application for a long time. On the contrary, SaaS applications offer many advantages over traditional software including avoiding the huge hidden personnel cost in deploying, running and maintaining traditional software.

8.2 The Elusive Break-Even Point

Another argument for "owning" software has been that even with the higher upfront costs, there is a break-even point where traditional software becomes much cheaper than the SaaS subscription model.

The analyst firm IDC reviewed several SaaS versus traditional software deployments and found that when people resources and cost of upgrades are correctly taken in consideration, this break-even point may never be realized.[19]

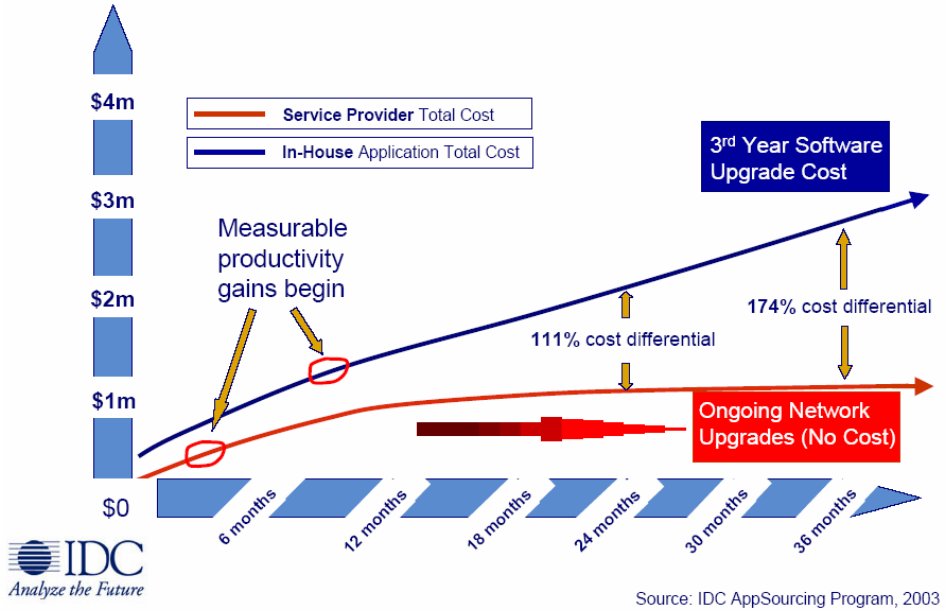

Figure 9: IDC's TCO comparison between traditional software and SaaS

[19] Robert Mahowald, Do Service Providers Deliver Value and Reduce Enterprise Costs?, IDC, 2003

8.3 TCO Calculator

The TCO calculator provided below incorporates all the cost variables discussed in this white paper. Using this calculator should enable companies to make better informed decisions when considering SaaS or traditional software deployments. This worksheet is of course not the answer for all questions but should give companies consulting this whitepaper a good initial feeling to what deployment solution works best for them. An electronic copy of this calculator is available for download on the SIIA website (www.siia.net)

TCO Comparison Matrix	Initial Cost		Ongoing Costs							
	Setup & Deployment		Year 1		Year 2		Year 3		Year n	
	Software	SaaS	Software	SaaS	Software	SaaS	Software	SaaS	Software	SaaS
Comporable Cost Analyis										
Capital Expenses										
Hardware										
Software or License costs										
Support and Maintenance										
Upgrades										
Facilities/Datacenter Expenses										
Management and Operations										
Design and Engineering										
Integration/Implementation										
IT and Helpdesk Staffing										
End User Training										
Scheduled Maintenance										
Unscheduled Maintenance and Outage Recovery										
Monitoring and Security										
Legal/Purchasing and General Administration										
Intangeable Costs										
Reliability and Availability										
Interoperability										
Extensibility										
Security										
Scalability										
Performance										
Capacity										
Opportunity Costs										
Initial and Annual Costs										

Total Cost after Year n		

Figure 10: Sample TCO Calculator

9 Bibliography

- Frederick Chong, Gianpaolo Carraro and Roger Wolter, Multi-Tenant Data Architecture, MSDN Library, Microsoft Corporation, June 2006,
- Frederick Chong and Gianpaolo Carraro, Architecture Strategies for Catching the Long Tail, MSDN Library, Microsoft Corporation, April 2006,
- Robert Mahowald, Do Service Providers Deliver Value and Reduce Enterprise Costs?, IDC, 2003
- Robert Mahowald, Conferencing Through Service Providers for Low Cost and Reliability, IDC Executive Brief, November 2003
- Geoffrey A. Moore, Living on the Fault Line, Revised Edition, HarperCollins, 2002
- Timothy Chou, The End of Software: Finding Security, Flexibility and Profit in the On Demand Future, SAMS Publishing, 2005
- Microsoft Wages Campaign Against Using Free Software, The Wall Street Journal, December 9, 2002
- MultiMedia Communications, A Detailed Analysis of "On-Premise vs. Service Provider" Costs and Risks, WebEx Communications, 2005
- Messaging Total Cost of Ownership: Microsoft Exchange 2003 and Lotus Domino in Small and Medium Organizations, META Group, July 2004
- Software as a Service, Wikipedia.org, July 2006