Software Services and Packaged Software Solutions: a Customer Perspective

05.09.2006

Helsinki University of Technology
Systems Analysis Laboratory

Tomi Myllylä
tomi.myllyla@hut.fi
55289W
# Table of Contents

1. INTRODUCTION .......................................................................................................................... 1

2. SOFTWARE SERVICES AND PACKAGED SOFTWARE SOLUTIONS – TWO DIFFERENT ENTERPRISE SOFTWARE MODELS .................................................................................. 4
   2.1 OVERVIEW .............................................................................................................................. 4
   2.2 SOFTWARE SERVICES – ADVANTAGES AND DISADVANTAGES ........................................... 6
      2.2.1 Advantages for customers ................................................................................................ 6
      2.2.2 Disadvantages for customers .......................................................................................... 8

3. CUSTOMER COMPANY POINT OF VIEW ..................................................................................... 10
   3.1 HOW COMPANIES MEASURE THE VALUE OF THEIR IT INVESTMENTS? ......................... 10
      3.1.1 Total cost of ownership (TCO) ..................................................................................... 10
      3.1.2 Return on investment (ROI) .......................................................................................... 11
      3.1.3 Net present value (NPV) .............................................................................................. 11
      3.1.4 Real options analysis (ROA) ......................................................................................... 12

4. SOFTWARE LICENSING MODELS – REAL OPTIONS POINT OF VIEW ...................................... 15

5. CONCLUSION ............................................................................................................................. 17

6. REFERENCES .............................................................................................................................. 18
   6.1 LITERATURE ......................................................................................................................... 18
   6.2 OTHER SOURCES ............................................................................................................... 19
1. INTRODUCTION

Unpredictability of the costs and schedules of IT projects hampers the IT investments in many companies today. It is already a common fact that many of the software projects are delivered behind schedule and over budget. This also hinders the use of enterprise software solutions, such as enterprise resource planning (ERP) and customer relationship management (CRM) systems, in many companies. Gartner studies\(^1\) have implicated that between 50 and 80 percent of IT budgets are not spent on acquisition of software and hardware, but rather on system implementation and maintenance. Therefore, software licences usually only include fraction of the total costs of enterprise solutions. In addition, maintaining and running enterprise applications usually requires considerable amount of IT resources from underlying firms. All these facts contribute to the unpredictability of the total costs of enterprise solutions for the acquiring companies, and are naturally the characteristics of traditional models of packaged software. Service-oriented software models, such as Application Service Provisioning (ASP) and Software-as-a-Service (SaaS) models, provide another alternative for using software applications compared to acquiring packaged software solutions. Service-oriented software models transfer the IT burden from application customers to software vendors in the sense that software is totally maintained by software vendors and only accessed and used by customers. Thus, customers of enterprise software applications are relieved from the burden of maintaining extensive applications themselves. In addition, service-oriented software models often enable the use of pay-per-use based licensing models for applications, which should reduce the unpredictability of costs of IT customers. Strategic benefits of service-oriented software models are easy to identify: Customer companies can divert their resources more specifically for their core businesses, and software vendors can develop an infrastructure to operate software applications through scope and scale advantages. However, the benefits of service-oriented software models have not turned

\(^1\) Neela AM, Mein J (2003), How to Cut Your IT Maintenance and Support Costs, Gartner, 23 December, p. 1.
out to be that clear as many pioneering software service companies have failed in the market. Various explanations have been identified for this. These include the reasoning that software initially designed for customer-site deployment wasn’t suited to be offered as service, the transition from products to services was too quick, and that the mainstream customers were not willing to adopt the new solutions so quickly. Regardless, service-oriented software models are now raising their head again in the software industry.

Service-oriented software models, such as ASP and Software-as-a-service, have received attention in the research community: various studies have examined the potential and adoption of service-oriented software models in the software industry. General findings have been that small and mid-size companies could be the most potential adopters of enterprise software services. This stems from the fact that smaller companies often don’t have the needed IT resources to run traditional packaged enterprise applications that usually require large up-front licensing costs, thus many enterprise applications are tailored for the needs of large organizations. Therefore, smaller companies could be a likely segment to adopt more light-weight solutions that don’t bind considerable amount of resources and provide more flexibility and advantages already in smaller-scale use. Second, existing studies also have pointed out that companies highly value the cost-effectiveness and lower IT staff support requirements that service-oriented software solutions enable.

This brief study summarizes the research findings related to service-oriented software models and traditional models of packaged software solutions from the customer viewpoint. First, the unique characteristics and differences between these two enterprise software models are analyzed. Second, this study presents various methods that the customer companies of enterprise


applications may utilize when estimating the value of their IT investments, and for comparing alternative investments like ASP-based software services and packaged software solutions. Finally, a brief example shows how to use real options to analyze service-oriented software models and traditional packaged software solutions, and compare the value of these investments.
2. SOFTWARE SERVICES AND PACKAGED SOFTWARE SOLUTIONS – TWO DIFFERENT ENTERPRISE SOFTWARE MODELS

2.1 Overview

Software as a service (SaaS) can be defined as “a software solution that is hosted and supported by a vendor as a service, which is accessed by users via the Internet, without the need to deploy and maintain an on-premise IT infrastructure”\(^4\). The SaaS model is different compared to traditional models of packaged software in many ways. Whereas in traditional software models customers acquire their enterprise software usually with large up-front license costs and then install and maintain the software themselves, SaaS model shifts most of the software related tasks to software vendors. By utilizing SaaS model, software customers should be able to diminish their IT infrastructure investments, reduce their administrative burden related to hosting and maintaining enterprise applications by themselves, and take advantage of web based access to services that can be deployed quickly and are hosted and operated completely by software vendors. In addition, companies should able to reduce their total costs of ownership related to software since costs would be more predictable and defined through service agreements\(^5\). Service-oriented software models make many new business models possible, since customers of enterprise services would be able to acquire software services in the scale they need. At least in theory, Software as a Service could provide many benefits both to software vendors and customers, especially by shifting IT burden from customers to software vendors allowing both parties to concentrate on strategic aspects that are most important to them. The roles of software application suppliers and end-users in service-oriented software models are illustrated in Figure 1, compared to traditional model of packaged software presented in Figure 2.

---

\(^4\) The Future of IT in Large Corporations, a White Paper of Software as a Service, Thinkstrategies, 2005.

Figure 1. The application service provider provisions and operates an application, offering it for use over a wide-area network.

Figure 2. The enterprise application model leaves application software development and provisioning to specialized firms while retaining most operations in the end-user organization.

---

2.2 Software services – advantages and disadvantages

2.2.1 Advantages for customers

Waters\textsuperscript{8} identifies several benefits that utilizing software applications as services bring to customer companies.

- Reduced total cost of ownership (TCO)

Waters argues that customers of enterprise applications achieve more predictability to their IT costs by utilizing service-oriented software models. He reminds that, in software-as-a-service models, the total costs can be estimated better since the costs can be contractually defined when subscribing to application services. This reduces the hidden costs of IT investments from the customer point of view, since this burden is now shifted to software vendors. From the software vendor point of view, vendors are offered possibilities for gaining economies of scope by offering same services to multiple customers. The illustration of predicted and hidden software costs is presented in Figure 3.

\textsuperscript{7} Messerschmitt DG, Szyperski C (2003), Software Ecosystem: Understanding an Indispensable Technology and Industry, Cambridge, MIT Press.

• Reduced speed of deployment

Kaplan\(^9\) reminds that, depending on complexity, enterprise applications can sometimes take months or even years to implement and they may consume a significant portion of IT staff’s time. Maintaining and upgrading processes, security issues and other tasks related to enterprise software management also bind considerable amount of IT resources of underlying firms. Due to delayed implementation, it may sometimes happen that firm’s original business needs have changed during the enterprise software implementation, and installed enterprise software turns out to be ineffective or even obsolete after it has been deployed\(^11\). However, service-oriented applications improve this aspect, since there are no software components that the customer needs to install and maintain, and software vendor can


subscribe new customers to services almost instantly. Therefore, the process of software service deployment is rather straightforward and customers are able to obtain returns for their investments quicker than acquiring traditional packaged software. Waters also reminds that customization and configuration of software can be more easily conducted, since software service vendors may perform these operations in their own datacenters.

- Optimized capacity utilization

Utilization of software services often leads to more effective use of capital by software customers, since they are able to purchase just the needed capacity (data storage, bandwidth, software features) for the services that they need, and they can expand their service capacities almost instantly when needed. Therefore, software services are expected to bring more flexibility for enterprise software customers.

- Risk mitigation

Software service customers can be considered better positioned against risk than the customers of traditional packaged software applications, since the unpredictability related to IT costs is shifted to software vendors.

2.2.2 Disadvantages for customers

- Reduced customization possibilities

Sharing the same application with multiple customers provides benefits for the software vendors, but may cause limitations for application customers by reducing the customization possibilities of service applications\(^\text{12}\). When utilizing in-house packaged software solutions, companies are able to freely customize the applications for their individual needs. However, since software services are one-to-many solutions the customization possibilities

by individual users may become limited. Limited customization possibilities may also hinder the ability to integrate the application services with companies’ existing IT systems that customers might possess.

- Data sensitivity

Companies are usually careful not to expose confidential information to external parties. Therefore, many companies may hesitate in storing sensitive data to application servers hosted by external parties. Packaged software solutions hosted internally, on the other hand, provide stronger means for companies to maintain the control of their sensitive data.
3. CUSTOMER COMPANY POINT OF VIEW

3.1 How companies measure the value of their IT investments?

Firms utilizing or planning to utilize enterprise applications in their businesses need consistent approaches for estimating the cost and value of their investment alternatives. Taking into account the nature of software projects, this often becomes extremely difficult. Some useful tools for companies estimating the costs and value of their IT investments are: (1) total cost of ownership (TCO) model, (2) return on investment (ROI) model, (3) net present value (NPV) model, and (4) real options analysis (ROA) model.

3.1.1 Total cost of ownership (TCO)

Ellram\(^{13}\) states that “total cost of ownership (TCO) represents an innovative philosophy aimed at developing an understanding of the “true cost” of doing business with a particular supplier for a particular good or service”. The purpose of TCO approach is to widen the scope of cost estimation beyond mere price-based estimation, and to take into account all the relevant costs linked to specific product or service\(^ {14}\). Since software licenses usually only represent small portion of the total costs of the enterprise software\(^{15}\) TCO approach is considered a useful tool for analyzing the total costs for acquired and maintained enterprise software. Therefore, many companies are using total cost of ownership models when estimating the costs of their potential IT investments. Related to IT investments, TCO calculations may include supplier search and qualification, license fee, delivery, installation, running, maintaining, upgrading and even estimated downtime and warranty costs assigned for specific IT investment.


\(^{15}\) Neela AM, Mein J (2003), How to Cut Your IT Maintenance and Support Costs, Gartner, 23 December, p. 1.
Ellram\textsuperscript{16} reminds that TCO analysis can be applied not only when pondering the make-or-buy decision, but also “after an organization has determined that it will use a third party (buy) rather than use an internal source (make)”, since transaction costs may vary significantly between different suppliers and this can be an important decision factor.

### 3.1.2 Return on investment (ROI)

Return on investment is a simple tool for determining the validity of investments or to compare different investment alternatives with each other. In general, ROI can be calculated as follows:

\[
ROI = \frac{(GainFromInvestment - CostOfInvestment)}{CostOfInvestment}
\]  

Basically positive ROI indicates a profitable investment, although this tool is often considered too simple and flexible estimation tool, rendering it vulnerable for manipulation\textsuperscript{17}. Regardless of the problems with this tool, return on investment is a common tool that firms are using for determining profitability of their IT investments.

### 3.1.3 Net present value (NPV)

Net present value (NPV) model offers a very simple tool for evaluating the value of investments. NPV of an investment can be calculated as follows\textsuperscript{18}:

\[
NPV = C_0 + PV,
\]

\textsuperscript{17} http://www.investopedia.com
Where $C_0$ is the initial investment and $PV$ is the present value of all future cash flows related to the investment. Present value $PV$ can be calculated as follows:

$$PV = \sum_{n=1}^{N} \frac{C_n}{(1+r)^n},$$

(3)

Where $r$ is the discount factor and $C_n$ is the cash flow in time $n$. Generally, if the NPV is greater than zero, the firm should make an investment decision. Investments with larger NPV ratios should take priority over investments that contain lower NPV ratio. Kambil et al\(^{19}\) argue that some disadvantages of NPV in valuation of IT investments are that it doesn’t consider the value of managerial flexibility or the value of potential follow-up investments arising from the project.

### 3.1.4 Real options analysis (ROA)

Finance theory defines an option as a contract that gives its owner the right but not the obligation to buy or sell a specified amount of financial or real assets at a specified price by or on a specified date\(^{20}\). There are two types of financial options. A *call option* enables the owner to buy a specific amount of financial or real assets at a prespecified exercise price at a specific time. A *put option* enables the owner to sell a specific amount of financial or other assets at a predetermined price at a specific time. Generally, the price of an option is determined by the following six parameters: the price of the underlying asset, the strike price (price to be paid to exercise the right), time to expiration, the volatility of the underlying asset, the cash flow generated by the asset (such as dividend payments), and the prevailing interest rate.

Common tools for option valuation are binomial valuation methods and Black-Scholes method for option valuation. Binomial valuation methods can

---


\(^{20}\) Ibid
be used if the time period of the option is discretized, and Black-Scholes valuation applies for continuous time-models, but is computationally more complex.

Black-Scholes equation is a partial differential equation that must be satisfied by any price function \( f(S, t) \) that is derivative to the underlying security with a price process

\[
dS = \mu S dt + \sigma S dz ,
\]  
(4)

Where \( z \) is a standardized Brownian motion\(^{21}\).

Black-Scholes partial differential equation for option pricing:

\[
\frac{\partial f}{\partial t} + \frac{\partial f}{\partial S} rS + \frac{1}{2} \frac{\partial^2 f}{\partial S^2} \sigma^2 S^2 = rf ,
\]  
(5)

Where \( r \) is the interest rate and \( S \) is the price process from (4).

Options theory can also be utilized for evaluating investment opportunities that are not based on financial instruments. These options are often referred as real options as opposed to financial instruments\(^{22}\). Real options solve some of the problems of discounted cash flow models in investment valuation, and they can be considered more flexible in that sense. Brealey and Myers\(^{23}\) identify several categories of real options: expansion options, abandonment options, timing options, and options providing flexibility in production.

Regarding IT investments, real options can be used for valuation of e.g. following aspects\(^{24}\):

\[^{22}\text{Ibid}\]
• IT investments may have strategic potential to increase or maintain company market share

• IT investments may provide basis for new sources of revenues

• IT investments may provide more flexibility in adapting to new business contingencies
4. SOFTWARE LICENSING MODELS – REAL OPTIONS POINT OF VIEW

This part provides an example how service-based software models (such as ASP and Software-as-a-Service) and packaged in-house software models can be approached from real options perspective.

Service-oriented software models often enable the possibilities for usage-based licensing for software applications instead of fixed license costs. Techopitayakul and Johnson\textsuperscript{25} have compared the ASP and packaged software models using real options analysis. Figure 4 shows the economics of usage-based pricing for IT applications. Regarding the figure, if the usage-level of an application is larger than $u_0$, companies should buy their software with fixed license fee for unlimited use. If the usage is smaller than $u_0$, usage-based licenses become more profitable for companies licensing the software.

![Figure 4. Economics of usage-based pricing\textsuperscript{26}. A = Revenue from fixed-price software, B = extra usage-based revenue from infrequent users (previously not customers), C = extra usage-based revenue from heavy users (previously a surplus to customers), A + B + C = usage-based revenue.](image)

However, customer companies may be uncertain about their usage-levels of enterprise software that they are about to acquire. Real options provide a

\textsuperscript{25}Techopitayakul D, Johnson B (2001), ASP-based Software Delivery: a Real Options Analysis, \url{http://www.realoptions.org/papers2001/Techopitayakul.pdf}
useful tool for hedging against this kind of risk. In this kind of situations, real option may be acquired for subscribing to an ASP-based software delivery with usage-based licensing but retaining an option for switching to a subscription fee. This happens by customer acquiring a call option on their usage costs with a subscription fee as a strike price. This real option is illustrated in Figure 5. By using this option, the user can hedge against the uncertainty risk related to usage-levels of the software.

![Figure 5. Real options analogy of an option to switch from usage-based pricing to a subscription fee.](image)

Real options provide also means to prepare for other situations that service-based software deliveries may cause compared to in-house software solutions. It may happen that as IT customer companies grow the benefits of ASP-based software deliveries decrease. This may happen e.g. because company’s own IT infrastructure grows, and flexibility and scalability become less important factors compared to security and performance issues. Therefore, companies may start preferring to host their own IT solutions instead of service-based software deliveries. Expansion option can be utilized in preparing for this kind of situations, where ASP vendor assumes the investment work associated with future service fees but is compensated by up-front option price. Real options can also be used for offering IT customers the possibilities to end ASP contract prior to its’ expiration.

*Ibid*
5. CONCLUSION

This study has analyzed and compared the differences of software service models and packaged software models, especially from the customer perspective. First, unique characteristics of these different software models were presented, and later on their advantages and disadvantages were analyzed from the application customer perspective. Several valuation methods for IT investments were presented that companies can utilize when measuring and comparing the value of alternative IT investments. These included total cost of ownership (TCO) models, return on investment (ROI) models, net present value (NPV) models, and real options analysis (ROA) models. Finally, example IT investment problem formulations utilizing real options were presented.

Conclusively, software service models provide high potential that is yet to be realized. It is clear that the advantages of software-as-services models appeal for many companies but, on the other hand, traditional models of packaged software are perhaps still more suitable for some companies. Flexible software-as-a-service solutions potentially attract smaller companies with limited IT infrastructure and resources for investments, while traditional packaged software solutions may often turn out to be more attractive for large enterprises commanding more resources for covering large fixed costs and maintaining the applications themselves.
6. REFERENCES

6.1 Literature


6.2 Other sources

http://www.investopedia.com