Evolution of the Software-as-a-Service model

The analysis from a business model perspective

Image source: www.etherfleet.com

Zhanerken Azimbayev

Management of Technology
Faculty of Technology, Policy and Management
Delft University of Technology, The Netherlands

Delft University of Technology

Delft, October 2011

Suggestion for citation:

Azimbayev, Z. (2011). Evolution of the Software-as-a-Service model: The analysis from a business model perspective. Unpublished master's thesis, Delft University of Technology, Delft.

Master Thesis Project

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Management of Technology.

Faculty of Technology, Policy and Management
Section Technology, Strategy and Entrepreneurship
Delft University of Technology

Graduation Committee:

Chairman	: Dr. Cees van Beers	(Delft University of Technology)
First Supervisor	: Dr. Roland Ortt	(Delft University of Technology)
Second Supervisor	: Dr. Mark de Reuver	(Delft University of Technology)

Author:
Zhanerken Azimbayev
Student ID 4056299

Management of Technology
Faculty of Technology, Policy and Management
Delft University of Technology, The Netherlands

This page intentionally left blank

Executive Summary

Software as a Service (SaaS) model allows subscription to a wide variety of application services that are developed specifically for and delivered over the Internet on an as-needed basis without the need to install and manage third-party software in-house. According to Salesforce.com Inc the wide adoption of SaaS model will eventually lead to the end of on-premise software era. Currently the success of SaaS model goes hand-in-hand with popularity of cloud computing. For instance, recently Google in collaboration with Samsung introduced to the mass-market their Chromebook with cloud-based operational system Chrome OS on board, which is also delivered as a service. Nevertheless, the idea of outsourcing the software or hardware is not new. Before SaaS there was Application Service Provider (ASP) model that in the past considered being very promising as well, but failed to meet the requirements of the wide market and serves a niche market today. The interesting fact is that ASP and SaaS models are very similar and some authors even don't make distinction between them. However, we believe that there are differences between them that affected the adoption of the models. Thus, we have set two objectives for this study. First is to conduct comparison analysis of two software delivery models from business model perspective and study factors that possibly affected the adoption. Second is to identify components of the business model that require changes in order to make a shift from one software delivery model to another and barriers that hamper these alterations. Therefore this research raised the following main research question:

"Why is Software-as-a-Service model more successful today than the model applied by Application Service Providers?"

In order to answer the main research question above, we have formed four sub-questions, answers for which were used as a foundation for the final analysis. We have adopted two theoretical frameworks to answer research questions. Firstly, in order to get an overview of the past and current state of the cloud computing industry, the diffusion and development patterns theory was used. Secondly, for the purpose of software delivery models comparison we took the STOF business model framework through which it was possible to highlight differing components of SaaS and ASP. Subsequently, we have formed and validated six propositions through two completely independent methods – case studies and expert interviews.

The results showed that business models of SaaS and ASP differ in four components: Technical Architecture particularly Software Architecture, Pricing, Cost and Market Segment. As predicted the most important component turned out to be the Technical Architecture that practically co-determined differences in other aspects. We confirmed that the Technical Architecture of the SaaS model is better in reaching 'economies of scale' compared to the ASP model, therefore the SaaS model was able to cut costs, drop service prices and serve wider markets which positively contributed to the large-scale diffusion of the model.

Furthermore, a set of recommendations for managers on ways of switching from one software delivery model to another were formed. We also conclude that although the SaaS has more advanced software architecture that makes it more successful on the market, it is still not a perfect solution for all types of companies. Mainly because of security issues that multi-tenant architecture entails, the enterprise application delivered over the ASP model could be a solution for certain market niches dealing with sensitive data, lacking IT expertise and willing to pay extra for the service. Therefore before committing to a certain model, providers have to carefully consider which type of companies they are able and willing to serve. It is also very important for companies that already have large installed base of customers and legacy software with single-tenant architecture. We found that virtualization technologies are rapidly developing and practically enable single-tenant applications to "fake" multi-tenancy and run on comparatively high levels of resource utilization as the SaaS model. Therefore the first recommendation for the traditional enterprise software providers would be to decide whether they are willing to maintain their current market segment or capture a wider market of SMEs and even individual end-consumers. Based on that decision, the most suitable architecture could be chosen.

However, this study has some limitations among which the most serious is the lack of attention to external macro economical factors that could play an important role in diffusion and development of the innovation. Although, to partially cover that topic was done, it still requires more careful and extensive research. Therefore suggestion would be to consider it as an opportunity for the further research in this field.

Keywords: *Software as a Service (SaaS), Application Service provider (ASP), Cloud computing, Diffusion and development pattern of high-tech innovation, Business model*

Preface

This thesis is a final step on the road of completion the Master's study of Management of Technology at Delft University of Technology. I have enjoyed the work on this research and find the field of cloud computing very promising and exciting. The managerial perspective on the issue and invaluable experience I have gained throughout the research perfectly supplements to my knowledge derived from previous studies.

First of all, I would like to express my sincere gratitude to Roland Ortt for being kind and patient, while giving invaluable feedbacks, support and discussions. I also would like to specially acknowledge Mark de Reuver for all insightful comments and clear instructions I have received to improve the research.

Furthermore, I would like to thank four respondents that agreed to cooperate and devoted time to give not just interviews but exciting discussions with valuable insights on the topic.

Also I would like to thank my family: Mother, Grandmother, sisters Gulnaz and Jamilya for their unconditional and infinite support in everything I do. Special thoughts go to my Father who passed away twelve years ago, but I still feel his love and support in all my endeavors.

Last but not least, thanks to my dear friends for adventures and fun we had during our studies. Finally, I am indebted to Bolashak Scholarship for giving me this unique opportunity to study on Master's Program at TU Delft.

Zhanerken Azimbayev
Delft, the Netherlands
October 19, 2011

This page intentionally left blank

Table of Contents

List of Figures

List of Tables

List of Abbreviations

API	Application Programming Interface
ASP	Application Service Provider
CCC	Communication and Collaboration
CRM	Customer Relationship Management
ERP	Enterprise Resource Planning
IaaS	Infrastructure-as-a-Service
ISP	Internet Service Provider
ISV	Independent Service Vendor
SaaS	Software-as-a-Service
SCM	Supply Chain Management
SI	System Integrator
SLA	Service Level Agreement
SME	Small-Medium Enterprise
STOF	Service Technology Organization Finance
PaaS	Platform-as-a-Service
SQL	Structured Query Language
Telco	Telecommunication Company

1 Introduction

Internet is continuously expanding and becoming one of the important channels for businesses in many sectors, making electronic service a separate business paradigm. The concepts of Cloud computing and Software as a Service (SaaS) are in wide use among the Internet community. The SaaS model is relatively young since the first wave (which in literature is also referred as Application Service Provider) of delivering software over the network occurred in 1990s. However, it has failed to meet the standards demanded by the market (Dubey and Wagie 2007). Therefore one of the goals of this research is to study the reasons behind the failure of first generation ASPs. Nevertheless, recently the concept was proved to be viable by success of the companies like SalesForce.com. According to IDC Research the market size of Software as a Service achieved $17.5 billion in 2010 and predicted to reach $40.9 billion by 2014 (Mahowald 2011). Moreover, by 2014 approximately 34% of all business software will be delivered via the SaaS. The huge potential of Cloud computing market triggered large software providers like Oracle, Microsoft, SAP, Google and many others to enter the market. This research aims to identify the reasons behind lower adoption of the ASP model compared to SaaS through the prism of business model analysis. Moreover, based on findings on differences between models we would like to know which components require change to make a shift from one software delivery to another. Therefore in the following section the problem at stake, and then based on that set the scope of the study and form research questions are described.

1.1 Initial definitions of models under study

Software-as-a-Service (SaaS) model – subscription-based software delivery model that implies the use of single instance of application and database serving multiple clients on shared infrastructure.

Application Service Provider (ASP) model – subscription-based software delivery model that implies the use of separate instance of application and database on dedicated physical/virtual machine.

1.2 Problem statement

In 1990s companies such as USinternetworking and Citrix Systems made large investments and committed to the ASP model. However, as has been discussed before the model did not reach the market stabilization phase and was widely replaced by the currently very successful SaaS model. According to the literature the main reasons behind the ASP model failure are the low bandwidth resulted in system availability concerns, the limited range of applications available and security issues (Greschler and Mangan 2002; Ekanayaka, Currie et al. 2003; Vassiliadis, Stefani et al. 2006; Heart, Tsur et al. 2010). Even though the factors provided above are seem to be valid, in our opinion it does not completely explain the situation with the APS model. For instance, cloud computing and particularly the SaaS model still have major security issues (Kandukuri, Paturi et al. 2009), but these did not prevent it from entering the market stabilization phase. Undoubtedly the limited range of apps available contributed to low adoption of the model, but it is hard to believe that such factor played a major role. Consequently, only strong factor left is the lack of infrastructure and particularly low Internet bandwidth. Indeed the limited connectivity drastically reduced

availability and reliability of systems delivered by application service providers. However, the innovation of software delivery model is complex and consists of different technologies. Therefore there are many factors that could affect the adoption process of models on different levels. Table 1 below provides four categories of possible factors (Ortt 2009). First category is “Company factors” which are responsible for the development, production and supply of the new high-tech product/service including company strategy, resources and capabilities etc. Second category is “Market” which describes characteristics of customers and market that use new high-tech product/service. Sometimes market could not be ready for the innovation and cause a delay in adoption. Third category is “Technological system” that mainly describes technological infrastructure requirements for implementation and adoption of high-tech innovation. Fourth category is Environment category addresses macro level factors such as regulation etc.

Table 1 Categories of factors found to affect diffusion of high-tech innovation (Ortt, 2009)

Company	**Market**
Fit mission and other criteria of companies to evaluate the importance of the product for the company Cheap for producer/supplier (overview costs/benefits) Resources main actor (to develop, produce and supply) Expertise (to develop, produce and supply innovation) Market (supply) strategy Number of suppliers for product and technological system; Number and resources of suppliers of alternative products/ technological systems.	Customer need and other customer-related criteria needed to evaluate the product Cheap for customer (overview costs/benefits) Resources customer (ability to adopt and use) Expertise (to use innovation) Adoption strategy Number of potential customers (market potential) Network effects on the customers or suppliers side Cooperation/competition among different actors
Technological system	**Environment**
Relative performance compared to alternative technology Competition other new/old technologies Required and available complementary products/s Reliability, certainty, risk of technology Complexity and network requirements of technology Availability knowledge components (newness) Controlled production is difficult Type of technology (Basic, general purpose and/or competence destroying technology) Visibility of benefits Applications of technology are unknown (newness) Is clear how invention can be turned in innovation? Compatibility with similar systems other regions or with previous systems	Regulatory environment Availability of rules and standards. General public attitude Accidental changes in the macro-environment Accidents during development/exploitation

However, in this research we would like to focus on a company level and study service innovation itself and not the macro-economical factors. Scholars mostly blame contextual factors such as low bandwidth or dot.com’s bubble burst for the failure of the first wave of ASPs in the market. However, we believe that shortcomings of software delivery models are the main reason of the low adoption in the market. Therefore it is necessary to carefully to study both models and compare them.

This research fills this scientific gap by studying the diffusion and development of software hosting service category from a dominant company business model perspective. It is

expected to make a list and highlight the most important aspects of business model to which firms providing software as a service have to pay attention in the first place.

1.3 Research objective

From scientific perspective, the objective of this research is to compare two software delivery models (SaaS and ASP) and understand possible reasons behind the low adoption of the first wave of application service providers.
From managerial perspective, we aim to identify components of business model that require changes in order to switch from one software delivery model to another. Meanwhile, highlighting possible barriers which can hamper that transition. In addition, we would like to demystify and clarify terms and technologies used by the industry, since marketing strategies and inconsiderate use of buzzwords by numerous companies confused the software market and even scholars.

1.4 Scope of the research

This research is focused on development of software hosting service phenomenon. Our findings suggest that there are different implementations of this idea, particularly the ASP and SaaS - software delivery models in which we are interested. Various applications can be delivered over both models including content collaboration and communication (CCC), enterprise resource management (ERP) etc. Therefore in order to assure consistency of the findings between two models we chose customer relationships management (CRM) as a piece of software under study. The arguments in favor of this particular type of software are provided in §2.6. In addition, as has already been discussed in §1.1 we are aware of different groups of factors affecting the adoption of models. However, the main interest of research lies in internal factors such as a business model of a company that includes resources and capabilities.

1.5 Research questions

Main research question

"Why is Software-as-a-Service model more successful today than the model applied by Application Service Providers?"

Sub-questions:

1. *What are the definitions of Software-as-a-Service model and Application Service Provider model?*
2. *What is the pattern of diffusion and development of software hosting service innovation?*
3. *What is the method to compare business models of ASP and SaaS?*
4. *What are the differences in business models of ASP and SaaS?*

1.6 Thesis outline

Research starts with the introduction of the problem. It is followed by Chapter 2 that presents discussion on the domain of cloud computing including two main models ASP and SaaS. Chapter 3 describes core theories and concepts used for the design of theoretical

framework of this research, particularly the STOF business model and the pattern of diffusion and development. In Chapter 4 we use this theoretical framework to identify differences between ASP and SaaS from STOF model perspective, the outcome of which are the propositions that are tested by use of methodologies, which are presented in Chapter 5. Accordingly, Chapter 6 presents results of case studies and conducted expert interviews through which propositions are validated. Finally, Chapter 7 elaborates on conclusions and recommendations formed based on our findings.

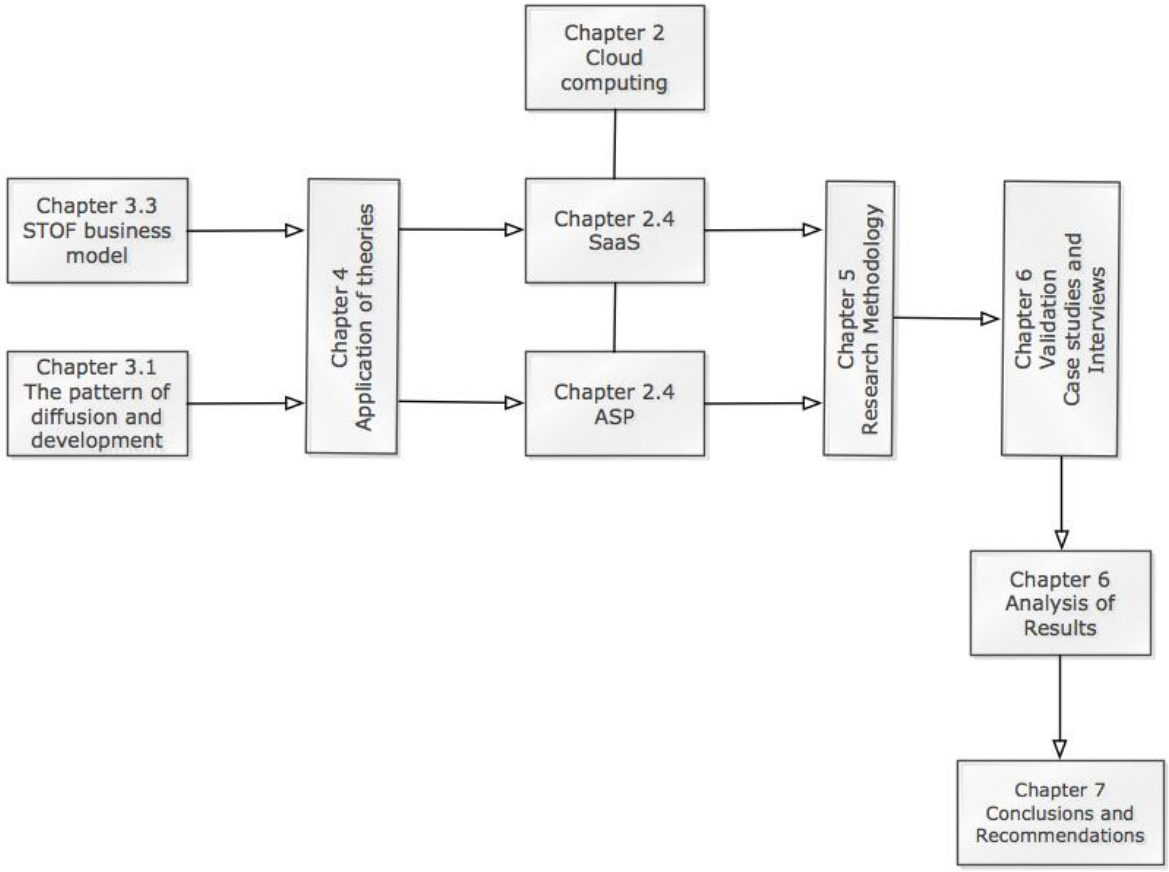


Figure 1 Report structure

2 Domain

This chapter discusses the insights of cloud computing domain. It starts with differentiation of service from products that is followed by more precise definition of the electronic service term. Afterwards it directly goes into details of cloud computing and three core concepts within it. Although the main focus of this research is the SaaS and ASP, for the complete understanding of the situation it is necessary to shortly elaborate on closely related concepts such as Platform-as-a-Service and Infrastructure-as-a-Service. The definition of these terms has been a matter of discussion among scholars. Perhaps the reason behind confusion is careless use of terms by marketers. Therefore the next step will be clarification of differences between them.

2.1 Services and Products

Gronroos (2007) defined a service as 'a process consisting of a series of more or less intangible activities that normally, but not necessarily, take place in interactions between the customer and service employees and/or physical resources or goods and/or systems of the service provider, which are provided as solutions to customer problems'. Service is a result of producer and consumer interaction and it is produced and consumed simultaneously, whereas product is manufactured independently. Usually services require physical products for their usage, for instance cell phones and telecommunication services (Bouwman, De Vos et al. 2008). However, with a product, very little changes can be made without significant time and financial costs, whereas service can be altered relatively easy. For instance, even a small change of a car design may result in costly changes of manufacturing line. As a result, products are more standardized while services are tailor-made, except electronic services that are going to be explained in the next section.
Grongroos (1992) argues that services have four distinctive characteristics: 1) intangibility – services are non-physical 2) heterogeneity – difficult to standardize 3) inseparability – simultaneous production and consumption 4) perishability – impossible to store. Nevertheless, these characteristics are more applicable for traditional services, whereas electronic services differ in several points.

2.2 Electronic service

Compared to traditional services, the delivery process of electronic services does not need human involvement, instead software fulfills this role (Bouwman, De Vos et al. 2008). The preliminary software development for electronic services results in separable consumption of services, therefore it becomes less personal and requires customer self-service. In e-services exceptions are not possible, because of the rules set by software and hardware (Bouwman, De Vos et al. 2008). Moreover separable consumption of electronic service makes any alterations in service design more difficult (due to software and hardware limitations) than in case of traditional services. For instance, even small change of service offering could result in significant change of software architecture accompanied with alteration in hardware requirements.
Second aspect is tangibility. Although e-services are considered to be intangible it still needs specific delivery mechanism (DVD, hand held device, personal computer) and format (web page, email, video, text message, voice menu) also offer an important contribution to tangibility (Hofacker, Goldsmith et al. 2007).

Next aspect is perishability. According to Hofacker, Goldsmith et al. (2007) electronic services are not perishable and can be inventoried. Authors give an example of media downloaded from web that can be copied and given to somebody else and yet still be retained. In contrast, traditional services cannot be inventoried and management is able to prevent consumer from copying, storing or exchanging.
Fourth aspect is heterogeneity. As have been mentioned before electronic service needs preliminary software development, which results in standardized offering. Therefore e-services are homogeneous, whereas traditional services allow high customization and personalization - heterogeneous.
The comparison of traditional services, products and e-services is summarized in table below.

Table 2 Product, service and e-service comparison (Hofacker, Goldsmith et al. 2007)

	Products	**Traditional services**	**E-services**
Tangibility	Tangible	Intangible	Intangible, but needs tangible media
Perishability	Can be inventoried	Cannot be inventoried	Can be inventoried
Consumption	Separable consumption	Inseparable consumption	Separable consumption
Proprietorship	Can be patented	Cannot be patented	Can be copyrighted, patented
Heterogeneity	Homogeneous	Heterogeneous	Homogeneous

Definition: *E-service "is deeds, efforts or performances whose delivery is mediated by information technology (including the Web, information kiosks and mobile devices). Such e-service includes the service element of e-tailing, customer support and service, and service delivery"* (Rowley 2006).

In addition, Hofacker et al. (2007) discussed three types of e-services 1) complements to existing traditional services – for instance, online package tracking in FedEx or DHL 2) substitutes for traditional services – for instance, Amazon as a substitute for traditional bookstores 3) uniquely new core services – service that would not exist as offline service, for instance online gaming such as World of Warcraft.

2.3 Cloud computing

Various definitions of the cloud computing are available in the literature. The Expert Group of European Commission gives broad and sophisticated definition "a cloud is an elastic execution environment of resources involving multiple stakeholders and providing a metered service at multiple granularities for a specified level of quality (of service)" (Jeffery and Neidecker-Lutz 2010). In contrast, Experts from University of California at Berkeley give more technical and very short definition where "cloud computing refers to both the applications delivered over the Internet and the hardware and systems in the datacenter that provide those services" (Armbrust, Fox et al. 2009). However, in this paper we are going to adopt third formal definition that in our opinion depicts all main characteristics of cloud computing: "It is an information technology service model where computing service (both hardware and software) are delivered on-demand to customers over a network in self-service fashion, independent of device and location" (Marston, Li et al. 2010). Although

there are slight differences in formal definition of the term, all authors agree that cloud computing is delivered over three models: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) (Armbrust, Fox et al. 2009; Jeffery and Neidecker-Lutz 2010; Marston, Li et al. 2010). This research is going to focus only on SaaS (which is defined in the next part), however a brief description of PaaS and IaaS is provided for the purpose of better understanding of the whole cloud computing concept.

2.3.1 Infrastructure as a Service

Infrastructure-as-a-Service (IaaS) is the most generic form of cloud computing delivery model (Figure 1). It refers to lower level services such as storage, computing, and database capabilities that are required to build an application environment from a scratch (Kraska 2010). Basically it delivers raw computing power as a service. It usually includes hardware, servers, networking components and other types of equipment. Infrastructure as a Service provides more freedom to developer while at the same time requiring the dealing with lower-level details such as virtual machines, patches et cetera. Example of IaaS:

- Amazon's:
  - EC2 – allows to rent virtual computers to run own applications on it;
  - S3 – simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.

2.3.2 Platform as a Service

Platform as a Service (PaaS) provides computational resources for the development and deployment of applications (Jeffery and Neidecker-Lutz 2010; Marston, Li et al. 2010). PaaS represents a higher-level of development platform, hiding from a user the details like an operating system or balancing load (Kraska 2010). PaaS is usually built on top of IaaS, that gives an opportunity for a developer to focus on business logic of the application, though often limiting to a single programming language or set of libraries (Kraska 2010). Basically, PaaS provides tools and environment for building and running an application. Examples:

- Google App Engine – platform for developing, hosting and scaling applications in Google managed datacenters. Provides two alternative environments for building applications: Java and Python.
- Microsoft's Azure - platform for developing, hosting and scaling applications in Microsoft datacenters. Azure offers to developers more freedom in choosing suitable framework than other platforms. By default system imposes to use .NET Framework programming model. Nevertheless, it is possible to develop on Java, Python, PHP and Ruby as well.
- Force.com – platform for developing, hosting and scaling multitenant applications. Requires use of proprietary languages Apex (for database interaction) and VisualForce (a HTML tag library for data-driven UI layouts).

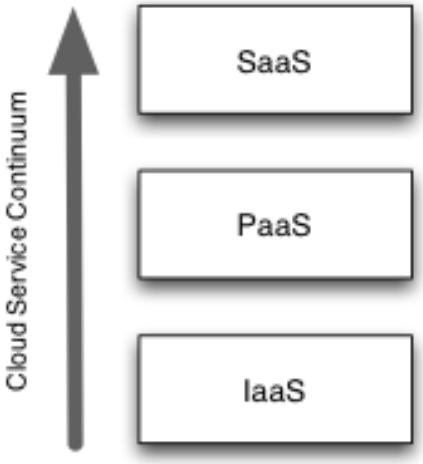


Figure 2 Cloud Service Continuum (Kraska 2010)

2.4 Software as a Service (SaaS) and Application Service Provider (ASP)

As have been mentioned before SaaS is one of the delivery models of cloud computing. Therefore sometimes SaaS is also referred to as Cloud Applications (Salesforce.com 2010). According to the most recent and simple definition, in SaaS the application runs on the cloud without a need to install and run it on the client computer (Marston, Li et al. 2010). SaaS represents the highest form of services delivery in the Cloud Service Continuum (Figure 1). Customer pays not for owning the software but for using, and typically it is accessed over a web-browser (Turner, Budgen et al. 2003). The term was in use since the late 1990s in parallel with Application Service Provider (ASP) term (SIIA 2001). However, the ASP label was largely abandoned after doc-com's downturn of 2001 and was re-designed and re-packaged by vendors under labels Web Service and SaaS in their marketing literature (Currie 2004). The result is a great deal of confusion in the marketplace. The basic idea of delivering software over the network on subscription basis is same in both models. But theoretical differences occur in software architecture and technological aspects. Details on software architecture and technological aspects are described in the following part.

2.4.1 Software as a Service maturity model

According to Chong and Carraro (2006) there are three key attributes that differentiate well-designed SaaS application from poorly designed one: 1) scalability – efficient use of application resources and maximized concurrency; 2) multi-tenant-efficiency – multiple customers utilize single instance of application set; and configurability – instead of writing a custom code for each end-user, metadata controls the configuration of the application behavior and appearance, the challenge of a developer is to simply the configuration procedure for customers.

Based on the attributes listed above, researchers from Microsoft developed the Software-as-a-Service Maturity Model, which helps to see the architectural difference between the SaaS model and ASP model. Model consists of four levels that are distinguished by addition of one attribute on every next level (Figure 2).

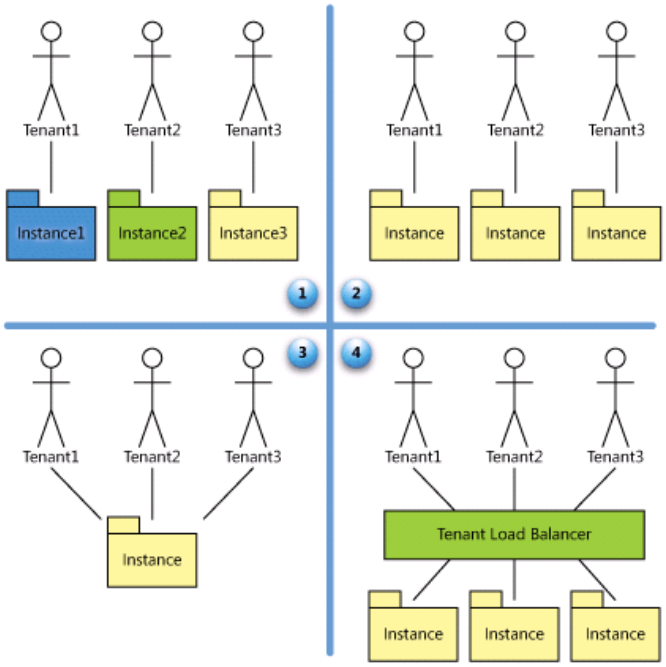


Figure 3 Software-as-a-Service Maturity Model (Chong and Carraro 2006)

The Application Service Provider model belongs to the first level (Level 1: Ad hoc/Custom), where every customer had wholly independent instance of application running on a server. The advantage of the first level is that traditional client/server apps can be moved to the SaaS model with very little effort and re-architecting (Chong and Carraro 2006). Depending on the bandwidth available vendors could choose whether to go with the thick client architecture – where part of the logic is hosted on a local computer and requires high bandwidth, or go with the thin client architecture – where the whole logic is hosted on provider side and allows to work with low bandwidth (Desai and Currie 2003). Basically, ASPs could just rent out packaged software to customer on either transaction or usage basis (Bennet and Timbrell 2000). However, it does not offer the advantages of the fully mature SaaS.

The second level (Level 2: Configurable) is characterized by using the same code implementation for every instance of application, but with provision of detailed configuration option. The second maturity level, as well as the first, usually requires large number of hardware and storage to support application instances running concurrently (Chong and Carraro 2006).

On the third level of maturity (Level 3: Configurable, multi-tenant-efficiency) single instance of application serves customers with additional metadata files that configure the code to behave in a tenant-specific way. Thus, this model and other on higher level no longer require any customer-specific investment by a vendor. In particular, the SaaS vendor is responsible for maintaining the common code base that delivers the standard application services to all customers; while each customer is responsible for maintaining their metadata (Xin and Levina 2008). However, the significant disadvantage of this approach is limited scalability, which could be only overcome only by using partitioning in managing database performance.

The final level of maturity (Level 4: Scalable, configurable, multi-tenant-efficiency) introduces a tenant load balancer that maximizes the utilization of hosting resources; customer's data is

kept separate and unique customization is reached through metadata configuration. The architecture of this model allows scalability to an arbitrarily large number of customers (Chong and Carraro 2006). In order to dynamically support the increasing load, fourth maturity level allows transparent adding of new instances of the application onto the instance pool. It is worth mentioning that even though the fourth level has all the benefits of a fully mature SaaS, it does not make it a ultimate goal for any SaaS application – the choice depends on the context and every vendor has to decide which level to target.

2.5 Delineation

Based on analysis provided in previous part and in order to avoid further confusion, for this research the strict set of defining characteristics of the SaaS are provided below:

1. The application should be developed from the scratch with usage of the Internet standards, e.g. accessible and operational through web-browser without a need to install any additional software or hardware.
2. Multi-tenancy. The application should be at least on the third level of the SaaS maturity model.
3. On-demand. Customer should be able to gain access to an application for free or on "pay-per-use" basis without any up-front investments in the application license, servers, people and other resources.

The strict set of defining characteristics of the ASP model:

1. Should be possible to adapt traditional on-premise software for delivering it as a service through client/server architecture
2. Single-tenancy – running a separate instance of application on dedicated physical/virtual server for every client. The application should be on the first or the second level of the SaaS maturity model
3. On-demand. Customer should be able to gain access to an application for free or on "pay-per-use" basis without any up-front investments in the application license, servers, people and other resources

2.5.1 Enablers of software delivery models

Concerning technologies behind the SaaS, ASP and Cloud computing, based on the architecture analysis we derived four of them: networking, virtualization, multi-tenancy and web services (Waters 2005; Marston, Li et al. 2010).

- ***Networking*** (e.g. the Internet, LAN, WAN et cetera) – the main component without which the delivery of the software from remotely located computer, data center, cloud, grid or cluster would be simply impossible.
- ***Virtualization*** is the technology that hides physical structure of a computing platform by presenting an emulated computing environment – it allows easy configuration on demand, replication and maintenance (Vouk 2008).
- ***Multi-tenancy*** allows "a single instance of an application software to serve multiple clients", which results in better utilization of a system's resources (Vouk 2008). Consequently, the addition of another client does not require installation of a separate hardware and software, which at a certain point results in economies of scale for a provider.

- ***Web Service*** that according to W3C is "a software system designed to support interoperable machine-to-machine interaction over a network" (2011). It helps standardize the interfaces between applications and solves the incompatibility problems of different systems that may be used by a customer.
- ***Standardized communication protocols*** (e.g. HTTP) that provide relative homogeneity and ubiquity of workstations. Regardless whether it is Windows, Mac or Unix the data is delivered over the same communication protocol.

2.6 Cloud Deployment Types

There are numerous cloud deployment types among which the most well-known are private, public and hybrid clouds. So far, there has been a trend in the industry to start with private clouds for internal purposes. Afterwards companies usually consider to sell capabilities publicly – public clouds and only then move to providing hybrid solutions (EuropeanCommission 2010).

- *Private cloud* – typically infrastructure that servers a single enterprise, managed internally or by a third-party and not a commercial offering. This solution has higher level of security, but entails limited scalability and high costs.
- *Public cloud* – infrastructure that designed to serve a market and not a single organization. Allows other enterprises to outsource services and use clouds for their own purposes. In contrast to private cloud, this solution has lower level of security but provides higher scalability and lower costs.
- *Hybrid cloud* – mixed employment of public and private cloud infrastructures what gives certain level of scalability and costs reduction through public clouds while keeping sensitive data under control in private cloud.

2.7 Security and Privacy

Security concerns are one of the main factors hampering the adoption of cloud computing. The massive concentration of data and resources make the cloud very attractive target to attackers. Moreover, high fragmentation of cloud deployment options and types of services within it make almost impossible to create a list of security controls that can cover all circumstances. Therefore Cloud Security Alliance recommends the adoption of risk-based approach that helps in evaluation of initial cloud risks and allows making informed security decisions (CSA 2009). Nevertheless, the European Network and Information Security Agency (ENISA) identified a list of most important classes of cloud-specific risks (ENISA 2009):

1. ***Loss of governance*** – moving data to the cloud implies loss of direct control over it
2. ***Lock-in*** – lack of standard data formats, procedures and interfaces results in difficulty to migrate from one provider to another
3. ***Isolation failure*** – shared resources or multi-tenant architecture always posses a risk of the failure of mechanism separating storage, memory, routing and even reputation between different tenants (e.g. guest-hopping attacks).
4. ***Compliance risks*** – using a public cloud in some cases implies that certain kinds of compliance cannot be achieved (e.g., PCI DSS (4)).
5. ***Insecure or incomplete date deletion*** – it is frequently the case when a request to delete doesn't result in true wiping of the data. In happens usually because of

backups stored on mirrored servers or the disk t be destroyed is used by other clients.

6. ***Malicious insider*** – even though the risk is lower due to high security measure taken by provider, still the damage that may be caused it far greater.
7. ***Data protection*** – in some cases it is difficult to customer to check the data handling practices of the provider. Although some providers have certifications such as SAS70 on their data processing and data security activities and the data controls they have in place.

Usually it is possible to customers to transfer risks to the cloud provider. However, there could be occasions when a risk leads to business failure, reputation loss or legal implications that is impossible for any other party to compensate for this damage (ENISA 2009).

2.8 Types of the Application

There is a wide range of application that could be delivered as a service – ideally any software could be delivered on the SaaS or ASP basis including operating systems, e.g. Chrome OS and Joli OS. However according to findings of IDC (Perry, Hatcher et al. 2009) the segment of business applications or enterprise systems is the most profitable among all IT Services. According to Gartner (2009) Customer Relationship Management is the most popular SaaS application ahead of Enterprise Resource Management (ERP); content communications and collaboration (CCC); and supply chain management (SCM). CRM SaaS applications tend to cover a wider range of functions in common processes, such as sales automation, marketing automation and customer service support whereas applications such as ERP, CCC and SCM usually focus on specific areas of business process support such as expense management, talent management, recruitment, web conferencing and procurement. Therefore we are going to focus on CRM providers that deliver their solution on SaaS or ASP basis.

Originally CRM systems were designed to support call center and e-mail channels, Internet and Mobile channels. CRM often includes local approaches to data management and business rules, for example call center-based CRM, Marketing-based CRM or Sales Force Automation-based CRM. However today companies move towards more unified system with centralized logic that keeps track on customer information and responsible for all customer interactions (Simons, Loon et al. 2009). With use of CRM software sales department can track potential customer and provide assistance, whereas support department can monitor customers on any complaints.

2.9 Conclusion

This chapter gives us an overview of cloud computing domain, particularly focusing on Software-as-a-Service and Application Service Provider models. Moreover, we looked into the difference between more general terms such as service, product and electronic service. Based on the overview, we are now able to answer first three sub-questions of this research.

"What are the definitions of Software-as-a-Service model and Application Service Provider model?"

Sub-chapter 2.3 presents cloud computing domain that consists of three delivery models: Software-as-a-Service, Platform-as-a Service and Infrastructure-as-a-Service. Each model is

defined in separate section and provided with illustrative industry examples. In this research the following formal definition of cloud computing is adopted: “It is an information technology service model where computing service (both hardware and software) are delivered on-demand to customers over a network in self-service fashion, independent of device and location” (Marston, Li et al. 2010). Concerning the SaaS and ASP models, these are both software delivery models that use networks (usually the Internet) to provide application on subscription basis. Although these two terms have conceptually similar definitions, there is an important difference between them on architectural level that are discussed in detail in a separate research sub-question. Basically, ASP model uses client/server architecture with separate instance of application running to serve each tenant. The main advantage of this model is that it’s relatively easy to adapt legacy software to work in accordance with the method, whereas pure SaaS model requires development of entirely new application based on web standards and use of multi-tenant architecture with single instance of application serving all tenants.
We have determined set of defining characteristics for both models.

Characteristics of the SaaS model:

1. The application should be developed from the scratch with usage of the Internet standards, e.g. accessible and operational through web-browser without a need to install any additional software or hardware.
2. Multi-tenancy. The application should be at least on the third level of the SaaS maturity model.
3. On-demand. Customer should be able to gain access to an application for free or on “pay-per-use” basis without any up-front investments in the application license, servers, people and other resources.

Characteristics of the ASP model:

1. Should be possible to adapt traditional on-premise software for delivering it as a service through client/server architecture
2. Single-tenancy – running a separate instance of application on dedicated physical/virtual server for every client. The application should be on the first or the second level of the SaaS maturity model
3. On-demand. Customer should be able to gain access to an application for free or on “pay-per-use” basis without any up-front investments in the application license, servers, people and other resources

This page intentionally left blank

3 Theoretical Background

In Chapter 2 we have discussed domain background. The following Chapter 3 defines theoretical notions used as a research framework. We begin by presenting the diffusion and development pattern of a high-tech innovation that is used for the industry level analysis and helps in identification of versions and key contributors to development of innovation. Afterwards, we switch to the single innovation level and study the business model construct within which we describe the STOF model specifically designed for the service innovation. Finally, conclude with an attempt to integrate both theories and use it in one single framework.

3.1 Diffusion and development pattern of a high-tech innovation

The diffusion research by Rogers (Rogers 2005) is the most well-known in the literature studying technological innovation. He describes diffusion process as S-shaped curve divided in five market segments on it (Figure 4), which afterwards was also re-shaped into the product life cycle (PLC). Indeed, Rogers' representation of diffusion process is found to be valid for wide variety of product/service categories (Ortt 2009). However, Rogers assumes that innovation occurs only in pre-diffusion period, which means that product/service remains invariant over the diffusion process. That could be true for relatively simple innovation (hybrid corn or dynamite), but in case of more complex innovation (for instance high-tech innovation), several designs of the same product/service may occur along time. That was the main point of Utterback et al. (Utterback and Brown 1972; Utterback and Abernathy 1975; Utterback 1994) who distinguished emergence of different designs of innovation in a single industry. Authors discussed the issue of appearance of dominant design during four phases of product life cycle. However, both theories pay less attention to pre-diffusion phases that may take long period before the large-scale diffusion, during which pioneers who actually invented product/service may no longer exist (Ortt 2009).

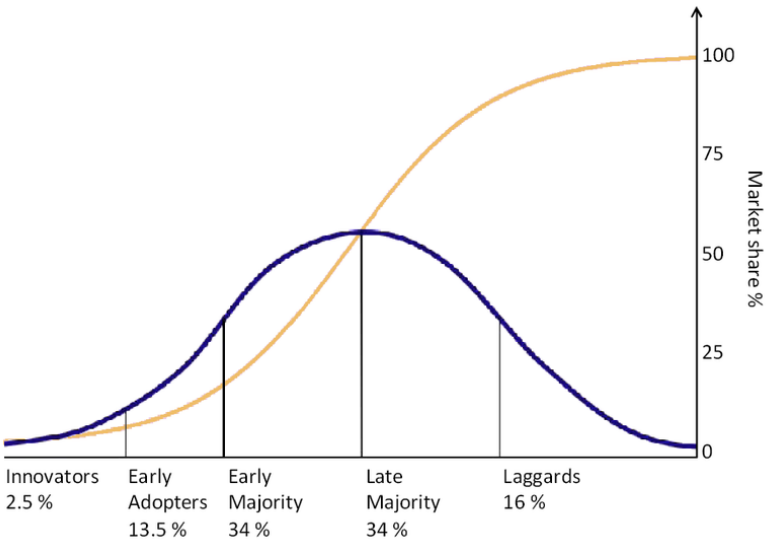


Figure 4 S-Curve

These issues were addressed in the diffusion and development pattern of high-tech innovation by Ortt (2009), where author introduces three milestones 1) the invention 2) the (first) introduction and 3) the start of large-scale production and diffusion. These milestones define two pre-diffusion phases: the innovation phase and the adaptation phase (Figure 4).

Milestones should be selected very carefully since it directly affects the duration of the phases, therefore the formal definitions by Ortt (2009) for each are provided below:

Milestone 1:
The invention of new high-tech product categories is defined to be the first time the technical principle of this category is demonstrated and mastered

Milestone 2:
The introduction date is defined to be the date at which the product is available for sales or can be transferred to users. In some cases products are not sold, for example if government institute develops a new weapon that is used by military forces

Milestone 3:
"The milestone is defined using 3 elements:

- *A standard product is required that can be reproduced multiple times (or standard product modules that can be combined in many different ways but are based on the same standard platform);*
- *A (large-scale) production unit with dedicated production lines (industrial production of standard product);*
- *Diffusion of the product/service"*

Each of the phases requires different resources and capabilities on behalf of the companies that are developing or commercializing high-tech products. In practice, it is found that different types of companies dominate subsequent phases in the pattern, but is it is unknown exactly why. To explain this shift in dominance an investigation of factors that affect innovation adoption is needed.

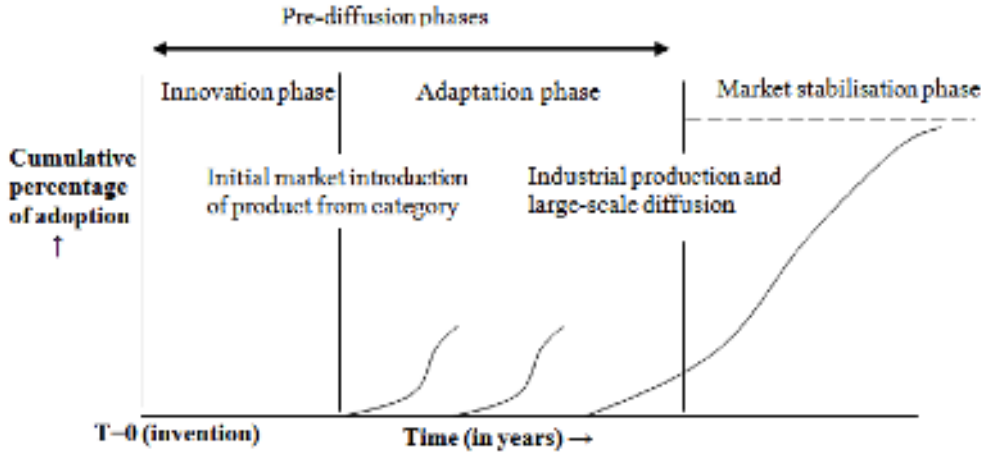


Figure 5 Pattern of diffusion and development of high-tech product categories (Ortt 2009)

Factors that affect the length of the pre-diffusion phases are important part of the model. Ortt (2009) groups them in four categories that describe the relevant environment for a high-tech innovation: 1) the company responsible for the development, production, supply and use of the high-tech innovation (company strategy, expertise); 2) customers – characteristics of customers, customer expertise in high-tech innovation 3) the technological system required for usage of high-tech innovation (required infrastructure, complementary goods, performance issues etc); and 4) the market-environment (actors, regulations etc).

Table 3 Categories of factors found to affect pre-diffusion phase (Ortt 2009)

Company	**Customers**
Fit mission and other criteria of companies to evaluate the importance of the product for the company Cheap for producer/supplier (overview costs/benefits) Resources main actor (to develop, produce and supply) Expertise (to develop, produce and supply innovation) Market (supply) strategy Number of suppliers for product and technological system; Number and resources of suppliers of alternative products/ technological systems.	Customer need and other customer-related criteria needed to evaluate the product Cheap for customer (overview costs/benefits) Resources customer (ability to adopt and use) Expertise (to use innovation) Adoption strategy Number of potential customers (market potential) Network effects on the customers or suppliers side Cooperation/competition among different actors
Technology	**Market-environment**
Relative performance compared to alternative technology Competition other new/old technologies Required and available complementary products/s Reliability, certainty, risk of technology Complexity and network requirements of technology Availability knowledge components (newness) Controlled production is difficult Type of technology (Basic, general purpose and/or competence destroying technology) Visibility of benefits Applications of technology are unknown (newness) Is clear how invention can be turned in innovation? Compatibility with similar systems other regions or with previous systems	Regulatory environment Availability of rules and standards. General public attitude Accidental changes in the macro-environment Accidents during development/exploitation

3.1.1 Milestones in diffusion and development pattern of a service innovation

According to Ortt (2009) the pattern depicted above is also applicable for a service innovation in the high-tech industry. The claim seems to be valid since conclusions of the research are based on case studies of numerous service innovations such as GPS, the Internet, Telegraphy et cetera. However, we found that characteristics of the third milestone are not applicable in cases of ASP and SaaS models. The first element requires a standard product/service that can be reproduced multiple times, but in case of the SaaS model an application has multi-tenant architecture where a single instance of an application software to serve multiple clients. Consequently, for a large-scale diffusion of the application delivered as a service no reproduction is necessary. For the same reasons the second element of the milestone, which requires a dedicated production line, is not applicable for the case of SaaS. This leaves us only with the third characteristic that does not really help in defining the large-scale diffusion milestone. Therefore we need to re-define the third milestone in order to make it valid for the case of SaaS and ASP. In his research along with definitions, Ortt (2009) provides criteria that milestones should fulfill:

1. Generic nature of the milestone: it should exist in (almost) all cases.
2. Data availability of the milestone: data should be available for (almost) all cases.
3. Objective timing of the milestone: milestones can be dated objectively.

Even though we have all criteria, it is still a very difficult milestone to define and assess, especially in case of SaaS since there are only few indicators of the large-scale diffusion on which we could possibly rely. In our opinion, one of the most reliable, generic and easily obtained indicators of the innovation adoption is when several large and influential companies enter the market with their solutions. Another possible indicator is adoption rate of an innovation. Both of the indicators fulfill the milestone criteria. However, obtaining objective data on market adoption rates of ASP and SaaS turned to be impossible for this particular research. Therefore, we have decided to go with an indicator of large companies entering the market (Table 5). Milestones are also verified through literature research, where numerous authors chose similar dates as milestones for ASP and SaaS timeline. Results of literature research are summarized in Appendix B. Table 4 below describes three milestones of diffusion and development pattern of software hosting service category. Tables 5 and 6 provide a list of key players in development and adoption of both models with description of their contribution and inceptions/disband dates.

Table 4 Milestones overview

Hallmark	Date	Description
Invention	1961	Invention of a principle of shared system and first public demonstration. Compatible Time-Sharing System (CTSS) - first generation time – sharing system developed at M.I.T. A normal batched job stream was run as background to keep the computer busy, while several users could enter commands to prepare, execute and terminate their programs. The machine directly responded in real-time.
First market introduction	1962	Built on principles of shared system, computing power (hardware, software) was available as a service for the first time. Emergence of computer or data processing service bureaus. A bureau typically owned mainframe computers and employed a staff of systems professionals. Users were charged rent for the terminal, a charge for hours of connect time, a charge for seconds of CPU time, and a charge for kilobyte-months of disk storage.
Market stabilization phase	2007	SaaS proved to a viable model. Leading vendor expands its business and starts provision of PaaS. Large on-premise software vendors such as Oracle and Microsoft enter the market with their solutions that became successful as well afterwards.

Table 5 Actors entering the market

ASP	Inception	Disband	SaaS	Inception	Disband
USinternetworking	1998	2006	Salesforce.com	1998	N/A
Corio	1998	2005	Microsoft Dynamics CRM 4.0	2007	N/A
Citrix iBusiness Partner Program	1999	2001	Oracle CRM on demand	2007	N/A
ASP Industry Consortium	1999	2001	Google services (Google Docs, Mail etc)	2008	N/A

Table 6 Market actors and factors

Phase	Market actors and factors	Description of their influence	Source
Innovation Phase	Massachusetts Institute of Technology (M.I.T.)	Early research and development in a field of the general-purpose time-sharing and virtualization. Created the system called Compatible time-sharing System (CTSS), which afterwards turned into widely known and successful CP/CMS	(Creasy 1981)
	IBM	In collaboration with M.I.T. developed and introduced the first widely available virtual machine architecture CP-67/CMS on IBM System/360 Model 67	(Creasy 1981)
Adaptation Phase	IBM, Honeywell, ADP, Digicon	Opened computer or data-processing service bureaus which turned to be a pioneers "in developing transaction processing software for specific kinds of businesses, and were early experts at integrating software written by one manufacturer with hardware from another"	(Creasy 1981)
	The Defense Advanced Research Projects Agency of the United States Department of Defense	Financed and promoted ARPANET – the first operational package switching network	(Focacci, Mockler et al. 2005)
	Citrix Systems	Founds the ASP industry Consortium with Traver Gruen-Kennedy on a chairman position. Large ASP brought IBusiness solution to the market	(Focacci, Mockler et al. 2005)
	Corio	One of the earliest ASP. The leading ASP in managing PeopleSoft Applications (27% of the total ASP market share for PeopleSoft applications).	(Bennet and Timbrell 2000; AberdeenGroup 2001)
	USinternetworking	The largest ASP in terms of customers potential business. Constructed its own global network of datacenter in US, Europe and Asia	(Kern, Kreijgerb et al. 2002)
Market Stabilization Phase	Salesforce.com Inc	Popularized the concept of SaaS with their successful Customer Relationship Management application. One of the pioneers in bringing to the masses the PaaS concept with their Force.com platform	(Olson, Claire et al. 2010)
	Amazon	Key player in bringing cloud computing to the masses	(MITTechnologyReview 2009)
	Google	Brought very popular both enterprise and consumer versions of Gmail, Google Calendar, Google Docs and Google Talk. Moreover, developed and introduced PaaS -	(Google 2011)

		AppEngine and highly anticipated cloud-based Operating System - Chrome OS	
	Microsoft	Public multinational corporation. Brought to the market PaaS Microsoft Azure and Microsoft Dynamics CRM that has both on-premise and on-demand versions	(Chappel 2011)
	Oracle	Multinational computer technology corporation that specializes in developing and marketing hardware systems and enterprise software products – particularly database management systems. Brought to the SaaS market currently popular Oracle CRM on-demand	(Gartner 2010)

3.1.2 Diffusion and development pattern of software hosting service category

In this part theories and findings described in §3.1.1 are applied for the case of software hosting service category within which we focus on ASP and SaaS. Appendix B summarizes important hallmarks of the innovation development. Every key event in the table is related either to the basic principle of delivering software as a service or to the technological enablers described in §2.5.1.

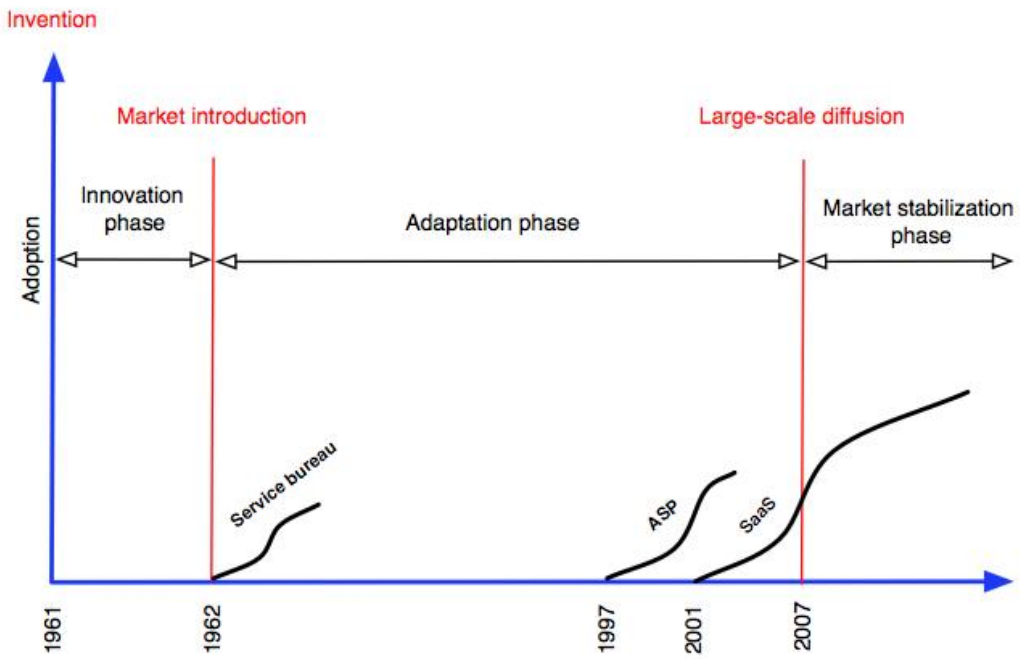


Figure 6 Pattern of diffusion and development of the software hosting service category

3.2 Disruptive innovation

The findings of Keller and Husig (Keller and Hüsig 2009) suggest that the new software delivery models pose disruptive threat to established on-premise software firms. According to authors disruptive innovation has numerous characteristics including:

1. The innovation allows for a product with a new combination of performance attributes (including the price).
2. The resulting product misses main market expectations in one or more established attributes and therefore targets only a niche.
3. Incumbents ignore the niche because of incompatible processes or values.
4. Entrants develop the innovation further and resulting products start to satisfy main market expectations in established performance attributes.
5. Incumbents lack necessary competencies in the innovation. They cannot provide the new performance attributes and fail.

This is in line with our findings on diffusion and development pattern of the service innovation in §3.1.2. The model started to develop with service bureaus, ASP and transformed into SaaS. Early versions of the service showed lower performance missing market expectation in some attributes. Therefore served only a niche market and usually delivering "not mission critical applications" such as Content, Communication and Collaboration. However, in time technology development such as high bandwidth, application architecture allowed to improve the performance of the innovation and reach a performance level that meets demands of the mass market in established attributes. As a result today the SaaS model represents a significant threat to on-premise software vendors.

3.3 Business model

In order to study the service innovation we need a construct that explains value creation process in e-businesses and breakdowns the innovation into components suitable for the analysis. Therefore, we have decided to utilize the business model concept that gained momentum after the dot-com bubble burst. The concept reflects the "architecture of a business" and closely linked to the rise of the Internet (Hawkins 2001).

The lack of common framework in the literature forced us to consider different options of business model interpretation. We started with Amit and Zott (2001) who identified four sources of value creation (novelty, efficiency, lock-in and complementarities) in business model construct. Thus, while authors are mainly concerned with defining purpose, scope and relationships of business modes with strategy, less attention was paid to fundamental components of the construct that could be used as a breakdown structure for design or analysis of a business model. Numerous authors chose more structured approach and addressed that issue by providing business model building blocks. Among them is commonly cited model by Osterwalder and Pigneur (2002), who defined four principal components with nine parameters:

- Product innovation – value proposition;
- Infrastructure management - capabilities, partnership;
- Customer - distribution channels, customer relationships, value configuration;
- Financial aspect - cost structure and revenue model.

Another suggestion was made by Weill and Vitale (2001), who distinguish: sources of revenue, strategic objective and value proposition, critical success factors, core competences, customer segments, channels, and IT infrastructure. Both approaches were recognized as valid, as well as received some critics for the excessive focus on individual

firms' corporate decision (Ballon 2007). Since the unit of analysis of the research is a service category, the excessive focus on companies but not on the services innovation was unacceptable. Therefore we chose STOF model as a framework for the service analysis. Although it shares some similarities with predecessors (Osterwalder and Pigneur 2002; Ballon 2007), it still goes a bit further and describes the construct specifically tailored for a service innovation. The STOF model proposed by Bouwman, De Vos et al. (2008) has several advantages over other models:

- Unit of analysis: service, not a company
- Considers involvement of multiple actors in value creation process
- Considers details as technical architecture
- Describes interdependencies between components of the model

Following parts will define the business model construct and outline the elements of the STOF model.

3.3.1 Definition

Since we use the STOF model in this research, the definition from corresponding literature is adopted: "A business model is a blueprint for a service to be delivered, describing the service definition and the intended value for the target group, the sources of revenue, and providing an architecture for the service delivery, including a description of the resources required, and the organizational and financial arrangements between the involved business actors, including a description of their roles and the division of costs and revenues over the business actors" (Bouwman, Faber et al. 2008).

3.3.2 The STOF model

The STOF is an abbreviation for the four domains of a business model (see Figure 7): Service domain (S), Technology domain (T), Organization domain (O) and Finance domain (F). Domains are interconnected between each other and change in one affects the others according to certain logic that in detail is depicted in Appendix D. The model is focused on a service and not on a company. Therefore the Service domain is a central component of this approach, which mainly refers to the value proposition of a service and the perception of this value in different market segments. Technology domain describes the technical architecture, backbone infrastructure etc "facilitating the process that enables the service development, creation, discovery, delivery, bundling, control and management" (Bouwman, Faber et al. 2008). Organization domain refers to resources and capabilities necessary for a service delivery. However, according to authors a single firm may not possess all the resources and capabilities necessary, therefore it has to collaborate with the network of actors. Finance domain consists of financial resources, financial arrangements (e.g. investment decision and revenue model) and performance indicators of a business.

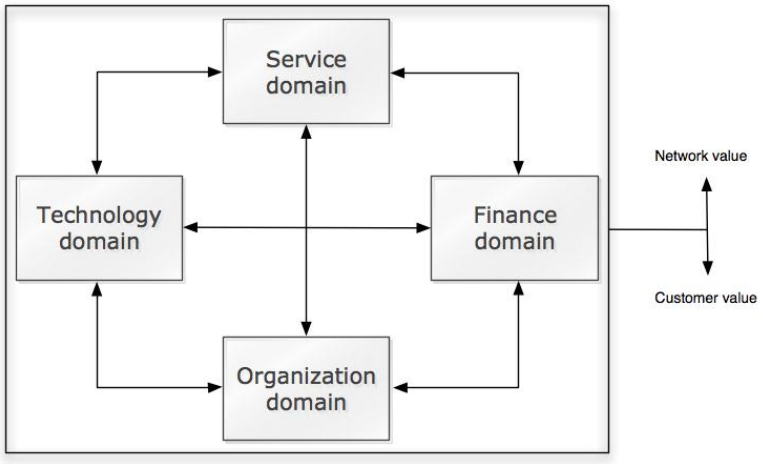


Figure 7 The STOF model (Bouwman and Faber 2008)

Above the general description of the STOF model is provided, but in order to answer the research questions we need apply the model in the context of SaaS and ASP and look into every domain in more detail during the research.

3.3.3 Dynamic STOF model

In previous part we have defined the basic idea of the static STOF model. However, it is likely to believe that business models tend to change over time under the pressure of external drivers such as technology, market and regulation (Bouwman, MacInnes et al. 2008).
Phasing is an essential element that adds dynamics to the framework. It is very similar to the phasing in diffusion and development patterns by Ortt (2009) although applied for a different level of analysis – micro level. It distinguishes three phases in the life cycle of a business model (see Figure 7): 1) Development/R&D; 2) Implementation/Roll-out; 3) Commercialization.

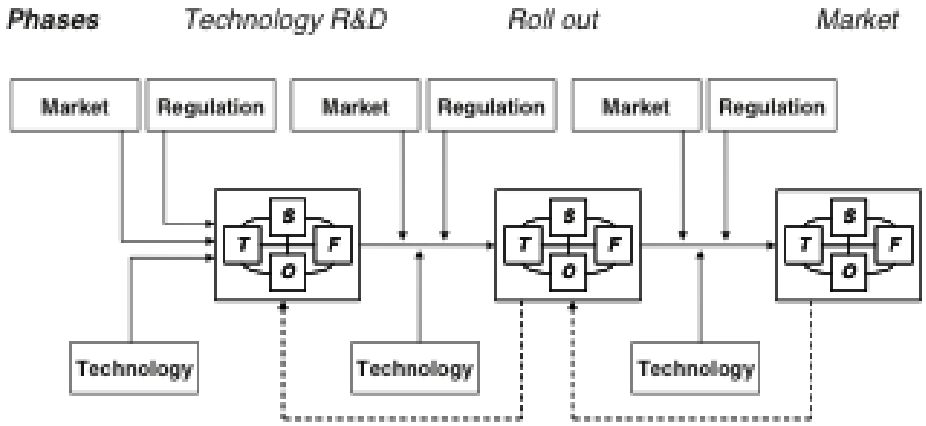


Figure 8 The Dynamic STOF model (De Reuver, Bouwman et al. 2009)

The Development/R&D phase starts with initial conceptualization of a business model. The shift from the first to second phase is characterized by launch of the service on the market, small scale roll-out, field experiment, alpha and beta testing et cetera. After market experiments have proven successful and adoption rate reaches critical mass business model enters the third phase, where focus shift towards commercial exploitation, operations and maintenance. However, business model may not follow the linear path, iterations may occur along the way.

As have been mentioned before changes in the business model are driven by external factors including technology, market and regulation. The findings of De Reuver et al. (2009) suggest that these drivers are more important and varies over time for innovating e-businesses and small start-ups rather than for large established companies. The effect on the business model is depicted above (see Figure 8). It was found that technology and market-related drivers have the most impact on the business model whereas impact of regulatory drivers is quiet weak in all three phases.
Although we aware of dynamics in business models, we would like to compare only the latest phases in the development of a service of particular firms. Since the main interest is comparison of services in their prime and finding strong/weak points of the models and not companies. Therefore we take a snapshot of companies' using ASP and SaaS in their latest business models and compare them with use of static STOF model.

3.4 Theory Integration

In previous sections we have described two models that are used to conduct the analysis on two different levels: the static STOF model – micro or service level; Diffusion and development patterns of a high-tech innovation – macro or industry level. Both theories focus on the innovation itself and not a company.
Research starts from macro level analysis of the whole software hosting service innovation. First of all, the pattern of diffusion and development of this particular innovation was created, where two latest and competing versions of this innovation category were identified – ASP and SaaS. The next step is to understand the difference between models and why one of them is more popular than the other. Therefore we have picked the STOF model that is tailored for the analysis of a service innovation as a framework for decomposition and comparison of models.
There are several advantages of using these two theories in combination:

1. The pattern gives us an overview of the past and current states of the service innovation category. Moreover, it helped to prove that the SaaS model is indeed more successful than competing ASP. While the STOF model allows focusing specific versions such as SaaS and ASP.
2. In both theories an innovation itself is the central point of analysis, not an industry or company. Moreover, both consider the dynamics and similar set of factors that affect the development of innovation.

We didn't encounter any major disadvantages in using the combination of models. However, we are aware of the fact that the pattern of diffusion and development is used for the study of a product innovation. Therefore in sub-section 3.1.1 we had to make few adjustments in definition of milestones for the pattern that allowed us to use it for the case of the service innovation.
Concerning the business model construct, according the authors the STOF model is geared towards mobile service innovation. Nevertheless it's still belongs to the class of electronic services as well as ASP or SaaS. Therefore we didn't experience any difficulties in applying for this particular innovation.

3.5 Conclusion

In this chapter we discuss theoretical concepts used to answer the main research question. The analysis on industry level is done through use of diffusion and development patterns of high-tech innovation. The micro level analysis was done on basis of STOF business model, which is introduced in this chapter and used in for the comparison in Chapter 4 and 5.

Prior answering following sub-question, first we need to recall the differences between services and products mentioned in §2.1. According to our findings products are produced and consumed separately, standardized therefore less personal compared to services. Moreover, it is more expensive to alter the product offering due to significant up-front investment into production line. However, it is easier to reproduce standardized product thus reach economies of scale. In addition, tangibility of products makes it possible to protect it with patents. On the other hand there are electronic services that share some characteristics with products. For instance, e-services allow separable consumption, which results in possibility to standardize the offering and easily reproduce it. Although e-services are intangible they still require products such as computer or smartphone in order to be delivered to consumers. In contrast to traditional service, e-service can be protected with copyrights.

"What is the pattern of diffusion and development of software hosting service innovation?"

The generic pattern of diffusion and development of high-tech innovation with three defined milestones is tailored for product innovation, but not service. Therefore we had to redefine the third milestone in order to make it applicable for the case of electronic services. As an indicator of the large-scale diffusion we picked the adoption of the model by numerous large software vendors, which showed that starting 2007 companies such as Oracle, Microsoft and Google enter the SaaS market with their solution. In contrast to the situation of ASP, where according to our findings only one multinational software corporation Citrix adopted the model and dropped it in 2001. Based on these and other facts that could be found in the Appendix B we identified milestones and created the pattern, where SaaS model managed to reach the large-scale diffusion with ability to cover wider markets and ASP model is still in adaptation phase and serves niche markets.

This page intentionally left blank

4 Business model comparison

In previous chapter we have discussed theoretical background of the research. In the next part, we are going to use the STOF model to break down both software delivery models into components and compare them one by one. Based on the results of the comparison - create proposition on differences between models and validate them in the following chapter. The starting point of the analysis is the Service domain.

4.1 Service domain

In this section we take a look into components of the service domain where possible differences between business models of the SaaS and ASP can be detected (see Appendix D). The important aspect of the service delivery is to create value for a customer, that in STOF model consists of four interrelated concepts: intended and delivered value on the part of the provider, and expected and perceived value on the side of the customer. The *Intended value* is the starting point in the analysis. It is usually equated to *Value proposition* of a service, but in reality there is often a gap between Intended and Perceived value (Bouwman, Faber et al. 2008). In case of the SaaS and ASP the *Intended value* by companies is similar. Both aim to deliver business applications on subscription basis over the network that entails no maintenance cost for the customer, scalability and low system integration costs. However, the interesting part is that in order to create this value provider has to translate the Intended value into functional requirements (Technology design) and into requirements for the value network (Organizational design) (Bouwman, Faber et al. 2008). This is where the differences may occur and will be furtherer discussed in sections 3.5 and 3.6.

The *Expected value* is influenced by the previous experience of customers, which means that they expect it to be better or at least not worse than the previous version of the service. We know that the market segment for both the ASP and the SaaS vendors was small and medium enterprises (SMEs) (Ekanayaka, Currie et al. 2003). In both cases SMEs opted for implementation of state-of-the-art technology without having to acquire in-house software development and maintenance expertise (Heart, Tsur et al. 2010). From financial perspective customers wanted predictable cash flow without a need to deal with some unexpected costs, e.g. upgrade costs, disaster recovery costs etc. Considering previous experience, the SaaS and ASP models were always compared with traditional on-premise software. And one of the important issues is the backward compatibility of the service (Bouwman, Faber et al. 2008). According the description of the ASP in §2.4 provider just hosted already existing packaged business application on pay-as-you-go basis so compatibility was not an issue. In case of the SaaS, applications are built from a scratch with use of the Internet technologies. Consequently, the application has different architecture than its on-premise counterpart, therefore customer migration from on premise to on-demand software could be a challenge and switching costs could vary depending on company size, and level of integration etc. However SaaS vendors and some Independent Software vendors provide different migration tools that drastically reduce switching costs, therefore we assume that the effect it has on Expected value of the SaaS model is relatively low, especially in case of small enterprises.

The next aspect is the *Delivered value* which according the descriptive model is co-determined by value network, technological functionalities and Intended value. We have already analyzed these co-determinants in §4.2 and §4.3 and it did not show any significant differences between two models. Therefore the conclusion is that Delivered value in ASP and SaaS is similar.

Perceived value is the evaluation of the innovation by customer or end-user. It is co-determined by numerous variables depicted in Appendix D. According to authors Perceived value reflects the difference between Delivered value and Expected value. Moreover, it is co-determined by components such as Context, Effort, Pricing and Bundling. In §4.2.1 we have discussed low bandwidth as one of the factors affected level of customer satisfaction, and made a conclusion that theoretically even dial-up connection was enough to deliver the service. Therefore there are no grounds to say that low connection speed significantly changed the experience. The next important factor is the pricing of a service. Presumably as discussed in §4.1.1 ASPs had offerings with higher prices compared to SaaS vendors. However, ASPs delivered slightly re-designed solutions from well-known packaged software vendors with better data isolation due to use of dedicated virtual servers for each client, whereas SaaS provider had to develop completely new application with multi-tenant architecture that even today is perceived as a less secure solution. Therefore even if customers had to pay higher subscription fee, they probably perceived it as an investment into more secure and well-known service. Overall, we didn't encounter any reasons to assume any significant differences in customer satisfaction differences between these two software delivery models.

4.1.1 Pricing

According the SaaS maturity model described in §2.4 the ASP model is on the first level that presumes client-server architecture. Basically, a vendor hosts and rents out traditional on-premise software with separate instance of application for each client, which results in limited scalability of the service. Therefore vendor can't fully utilize and maintain the 'economies of scale'. Consequently it has negative effect on price – makes it higher for a customer. On the other hand, the mature SaaS model which is built with use of the Internet standards and most importantly has characteristics such as scalability and multi-tenant-efficiency that help the vendor to achieve and maintain 'economies of scale'. As a result, provide lower prices. Moreover, in both cases the subscription-based revenue model resulted in large initial investment and delay for break-even.

Proposition 1 *Typically SaaS providers charge lower prices than Application Service Providers*

4.1.2 Market segment

The underlying idea of delivering software as a service on subscription basis geared towards smaller and medium enterprises that normally could not afford expensive enterprise applications. Both models are aimed to decrease total cost of ownership (TCO) and allow companies to focus on their core business rather than waste scarce resource on implementing the IT infrastructure in-house. However, as we discussed in §4.1.1 companies using ASP model are expected to charge higher price for their service due to technological limitation resulting in difficulty to reach economies of scale. Consequently, TCO increases and enterprise applications would be still expensive to implement in SMEs. In contrast, SaaS is able to charge lower prices that make applications affordable for smaller enterprises and even end-consumers. Basically SaaS provider is able to cover significantly wider market due to use of multi-tenant architecture.

Proposition 2 *Potentially SaaS provider is able to cover wider market compared to ASP*

4.1.3 Context

The physical context in which software delivered over the SaaS or ASP models are consumed is very similar and pre-determined. Usually an application is accessed through the Internet browser on a computer at work or home. In case of the ASP sometimes the end-user had to install client application (so called thin client) on a computer to access remotely located software. In case of the SaaS, current vendors also allow access to their services through mobile devices such as tablet computers, smartphones etc. However the context in which enterprise apps delivered via the SaaS or ASP models remains the same – professional life.

4.2 Technology domain

The most important technological difference between the SaaS and ASP is the *Technical Architecture*. As have been mentioned in §2.4 multi-tenancy is the core element in the SaaS model, whereas in the ASP model every client had separate instance of the application running on dedicated physical/virtual server. Application Service Providers heavily relied on virtualization technology, which means that basically they didn't have to provide expensive separate physical server for each customer. Computing resources were shared among tenants of a server and data isolation was achieved through virtualization. Even though it provides a better level of resource utilization, it is still has higher hardware and hardware maintenance costs in comparison with multi-tenant architecture. Consequently, for the ASP model the frequent need in acquisition of additional servers for provision of required computing power had negative effect on scalability of a service and high maintenance costs made reaching the 'economies of scale' was difficult.

Proposition 3 *Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model*

4.2.1 Access networks

Another possible difference lies in the *Access Networks* aspect of the Technology domain. In § 2.9.2 we have mentioned that the context in both cases remains very similar – work or home with the Internet connection and a computer. It is quiet pre-determined situation that allows usage of fixed network connection.
Concerning the Internet connection speed, in times of the ASP model (late 1990s – early 2000s) in developed countries it varied from dial-up (56 Kbps) to T1 (1544 Mbps), whereas today the average global download speed is around 5920 Kbps reaching 35 Mbps (~35000 Kbps) in some developed countries (Cherrytreeco.com 2000; Desai and Currie 2003). Certainly low bandwidth had a negative impact on customer experience, because availability and response time of a service depends on it. However, we believe that it had small contribution to low adoption of the ASP model. As it is described in §2.4 thin client architecture was an option in delivering remotely hosted application and according to findings of Dewire (2003) even dial-up connection was enough for reliable and secure transportation of keystrokes and screen updates. Therefore putting responsibility for the low adoption solely on bandwidth, that actually started to significantly grow in 2000s, would be inappropriate.

4.2.2 Devices

Devices used by end-users in both usually are desktop or laptop computers. However, today applications architecture applied by the SaaS vendors made them accessible from less

powerful devices such as tablet computers and smartphones. But in this case it barely affects the business model, since the context remains the same - professional life.

In the other design aspects we did not find any significant differences. Both models used the same Internet infrastructure with similar Service Platforms (for Billing and Metering, Personalization, Authentication etc), Web servers, Application servers, and Data storage servers. Moreover, the Technical Functionality is did not change as well. Basically it is functionality of enterprise apps such as ERP, CRM that has to be secure, always available with certain level of personalization.

4.3 Organization domain

The Organization domain stresses the importance of actors in a value network that possess resources and capabilities required for the service delivery. The literature study showed that the SaaS and ASP models have very similar value networks (Ekanayaka, Currie et al. 2003; Perry, Hatcher et al. 2009). Below we provide the generic value network suitable for both models.

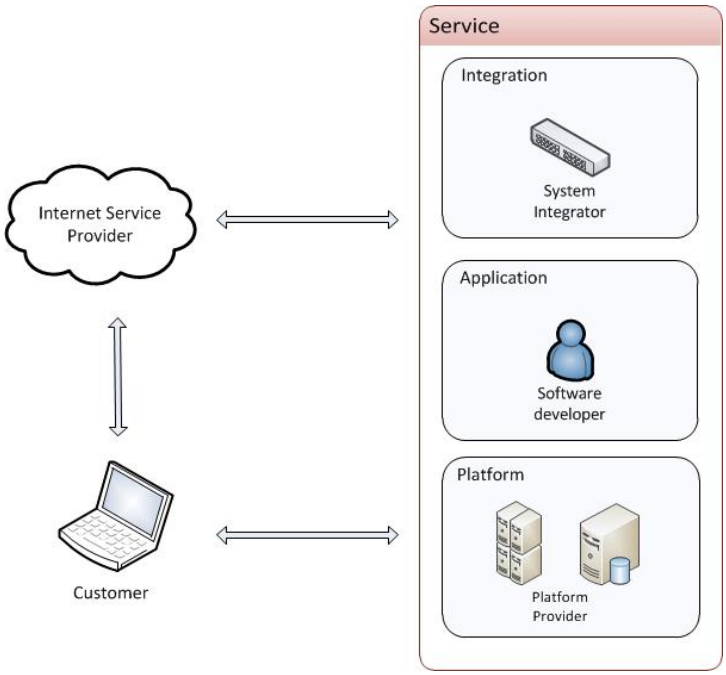


Figure 9 Value network for software delivery models

Basically there are four key roles in this value network that could be fulfilled by different actors depending on resources and capabilities possessed by them:

- Platform provider – typically they own and operate datacenters and offer managed hosting platforms that Software developers could use to deliver their solutions (Ekanayaka, Currie et al. 2003).
- Software developer – focuses on development of an application that delivered as a service.
- System integrator – outsource the rest of the layers, ultimately focusing on the final delivery of the service to the customers (Perry, Hatcher et al. 2009). However, the existence on this role is not compulsory for the value network and usually is fulfilled by software developer, infrastructure provider or ISP.

- Internet Service Provider – delivers the Internet connectivity to vendor and customers.

Depending on resources and capabilities available, one actor can fulfill multiple roles in this value network. In case of the ASP model typically single company plays roles of Infrastructure provider, System integrator and sometimes ISP. That is in line with the description we gave in §2.4 where the ASP vendor just hosts the application developed by a third party. In contrast in the SaaS model, company often fulfills roles of Software developer and System integrator. However, sometimes the SaaS vendors opt for vertical integration by fulfilling all the roles in the Service block, resulting in considerable initial investment and ongoing maintenance and personnel costs (Pang 2009). Advantage of vertical integration is a full control over every layer in service delivery process and better chance in reaching 'economies of scale'. Basically, the main value network difference of two models is the roles companies play: in the ASP model – actor usually fulfils any role but Software developer; in the SaaS model – actor is Software Developer plus any role in value web depending on resources and capabilities possessed.

Proposition 4 *In contrast to the SaaS provider, the ASP usually does not play a role of Software Developer*

Platform layer could be also made open for Independent Software Vendors (ISV), so they could develop solution for a certain platform – it is called Platform as a Service and shortly described in §2.3.2.

4.4 Finance domain

The Finance domain refers to design variables such as Investment sources, Cost sources, Performance indicators, Revenue sources, Risk sources, and Financial arrangements. As depicted on descriptive model in Appendix D, Technical Architecture of the application generates Costs.

4.4.1 Cost

We have already conducted the comparison of the SaaS and ASP architecture in §2.4 and §2.11 and the conclusion was that the ASP fully relies on virtualization which utilizes servers in less efficient way than multi-tenancy applied by SaaS providers. Consequently, ASP has to have more server than the SaaS provider in order to support the same number of clients. That results in higher initial hardware and operational costs for Application Service Provider than for SaaS vendor. Obviously, the large initial costs entail higher risks for investors who provide the *Capital*. Even though, the pay-as-you-go revenue model in both cases provides predictable cash flow for vendors, downside is a delay in breaking even. Moreover, in case of the ASP model high licensing costs could make accessing these applications through them more expensive compared to accessing them through the SaaS model where companies usually own the offering (Ekanayaka, Currie et al. 2003). All in all, it results in significantly higher costs for the companies applying the ASP model in comparison with the SaaS.

Proposition 5 *The ASP model implies significantly higher hardware and operational costs than the SaaS model*

Consequently, as it been mentioned in §2.11 for the ASPs it is way more difficult to gain sufficient customer momentum needed to drive the economies of scale they require to become profitable (Cherrytreeco.com 2000).

4.4.2 Risk

The next factor that threatens profitability is *Risk*. Both of the models face competition in enterprise application market from traditional packaged software vendors. However, two models deal with competition in a different way. In case of the ASP model, companies usually collaborate with packaged software vendor and deliver their solution as a service through infrastructure they have. Basically, the infrastructure is source of competitive advantage for the ASP, not the application. However, the infrastructure (e.g. datacenters) is perfectly imitable resource, therefore according to Barney (1991) it does not lead to sustainable competitive advantage for the company. The competitive analysis shows low barriers to entry as a major weakness of the model (Desai and Currie 2003). In contrast, the SaaS model focuses on the application and considers it as potential source of competitive advantage – it may have unique architecture that provides high scalability, efficiency etc. Software intellectual property could be protected in numerous ways including copyright, trade secret and patents (Schilling 2008). Intellectual property protection rights make the software unobtainable and inimitable resource for rivals. Therefore proprietary SaaS provider can differentiate itself from competition and have sustainable competitive advantage.
To summarize, one of the *Risk Sources* are similar for both models – on-premise software providers, but the resulting risk is different. The threat of large established packaged software vendors entering the market is significantly higher for the ASP model, whereas the unique characteristics of the SaaS model provide an edge over rivals.

Proposition 6 *The threat of large established packaged software vendors entering the market is significantly higher for Application Service Providers than for SaaS providers*

4.4.3 Revenue sources

We did not find any significant differences in revenue sources of models. The main revenue for both comes from subscription and support of the service delivered by companies. Although there are attempts to differentiate revenue streams by authors Churakova and Mikhramova (2009) who discuss some additional revenue streams for the SaaS model such as ancillary revenues – initial system set-up or installation costs; products revenue – selling additional equipment; counseling and support. However, we think that only counseling, training and support services are possible ways to get a small additional revenue stream, whereas the rest of the methods such as initial system set-up or equipment selling contradicts to the very essence of ASP and SaaS models. Therefore, they are not applicable as additional revenue sources.

4.5 Propositions

In the business model analysis above we defined several expected differences between ASP and SaaS. The next step is testing propositions on existence in practice through the case studies and interviews. Below formulated propositions are summarized in Table 7.

Table 7 Overview of propositions

Domain	Component	ASP	SaaS
Service	Pricing	**Proposition 1** *The SaaS providers charge lower prices than Application Service Providers*	
Service	Market segment	**Proposition 2** *Potentially SaaS provider is able to cover wider market compared to ASP*	
Technology	Technical architecture	**Proposition 3** *Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model*	
Organization	Value network	**Proposition 4** *In contrast to the SaaS provider, the ASP usually does not play a role of Software Developer*	
Finance	Cost	**Proposition 5** *The ASP model implies significantly higher initial and ongoing costs than the SaaS model*	
Finance	Risk	**Proposition 6** *The threat of large established packaged software vendors entering the market is significantly higher for Application Service Providers than for SaaS providers*	

4.6 Conclusion

This chapter introduced theoretical concepts used for the analysis of micro level and helps answering the following sub-question:

"What is the method to compare business models of ASP and SaaS?"

Since unit of analysis is the electronic service, specifically designed for the service innovation STOF model was used as a framework for theoretical comparison of SaaS and ASP models. STOF model consists of four domains (Service, Technology, Organization and Finance) that cover wide range of criteria necessary for delivering a service. ASP and SaaS models were compared in each aspect, based on what six propositions on differences were formed. Although we are aware of dynamics in companies' business models, due to time limitations and lack of data the decision was made to compare snapshots of latest available state of business model of a particular company.

Furthermore, prepared propositions are going to be validated through case studies and interviews. Leading and well-known companies using SaaS and ASP models were used as cases for the research. Moreover, four industry experts with different background were interviewed, elaboration on which is presented in following chapter. Through this validation, we expect to draw the most reliable conclusions on differences between models.

5 Methodology

Previous chapters provided domain insights and defined two main theories we use in our research. The goal of this thesis is to investigate possible reasons behind the success of the SaaS model compared to the less successful ASP model. We used diffusion and development pattern of high-tech innovation for the macro level analysis and business model concept for the micro level. With use of the STOF method we study differences between models and formulated propositions that are tested in following parts on real-life examples. In this part research methods, research framework and case protocol application are described.

5.1 Research method

Wide range of research methods are available such as survey, experiment, case study etc. However, there are always some requirements and limitations that push researcher towards one the most suitable method. The limited number of cases available excludes a possibility of conducting quantitative research. Therefore, in order the gain insights of cases, the qualitative approach from Yin (2002) is proposed. The research question provided above fulfills the requirements of descriptive research question. Moreover, it is impossible to isolate the phenomenon from its real-life context. Therefore according to Yin (2002) a case study research strategy is applicable. Case studies usually rely on multiple sources of data and often specifically selected rather than randomly chosen as it is done surveys or experiments. Case selection procedure is described in §5.4. In this research the cases are first examined individually. Further the results of the SaaS case are compared with the ASP model case that will allow confirmation/rejection of propositions made in §4.5. Contrasting results are expected for the cases of SaaS and ASP – theoretical replication. In addition, we conduct expert interviews as a way to validate proposition from a case independent source.

Table 8. Research methodology

Method	Purpose
Interviews	To gather industry experts' opinion on the subject
Desk research	To study the documentation and secondary data of selected firms
Literature study	To develop analytical framework for analysis of gathered data
Case study	Analyze the business model of SaaS and ASP on real-life example

5.1 Unit of analysis

The unit of analysis of this research is the software hosting service category within which we have defined two models ASP and SaaS. Models share the basic idea of delivering software from remote location, but have differences in implementation that potentially lead to the failure of ASPs. Therefore we would like to investigate differences between two models by means of business model analysis of companies applying ASP and SaaS models. In §5.4 we have selected two cases that were studied in Chapter 6 with use of the STOF model.

5.2 Information sources and data gathering

The outcome of the research is highly depended on quality of information and data sources. Therefore Literature study will be used as a mean for forming the foundation of the research. However some data may not be obtainable through the literature study therefore

the next step will be to confirm findings and obtain additional information over interviews. Desk research will be used to gain insights on subject of cloud computing, SaaS and ASP. Theories used and formed propositions will allow conduction of semi-structured interviews that will be held in real-life or over other communication channels (e.g. telephone, Skype). In order to improve the reliability of the results, triangulation method is applied that implies the use of multiple independent sources.

Sources. Data for the Literature analysis is acquired through the desk, scientific and corporate database search. That includes scientific article databases: ScienceDirect, Scopus, IEEE Xplore, Emerald, JSTOR, Google Scholar. Case descriptions were obtained from university repositories and company profiling system, for instance Repository of Penn State University (United States), Repository of Erasmus University (Netherlands), EDGAR System (U.S. Securities and Exchange Commission), LexisNexis Company Dashboard. The following types of sources are used:

- Scientific publications (articles, case studies, reports)
- Case descriptions
- Annual reports
- Press releases
- Interviews done by outsiders

5.3 Case selection criteria

Results of the business model comparison in Chapter 4 require verification through real-life cases. The cases need to satisfy certain selection criteria to ensure the validity of the study. Criteria are formulated based on the analysis of business model and diffusion and development pattern of the innovation.

Service. Solutions provided by companies should be offered as a service on “pay-as-you-go” basis and not as a product with one-time license fee.

Application Service Provider. Solution provided by a company has to belong to the first or second level of the SaaS maturity model. In addition, has to satisfy the requirement listed in § 2.5.

Software as a Service. Solution provided by a company has to belong to the third or fourth level of the SaaS maturity model. In addition, has to satisfy the requirement listed in § 2.5.

Dominance. Cases have to be industry leaders based on market share, company size or revenue. We will choose one case for each type resulting in two cases.

5.3.1 Case selection

The following section will look into the group of companies using the SaaS or ASP model and select the case accordingly. The selection criteria formulated above will be considered as a starting point in choosing the case.

Case 1 – SaaS CRM provider

Even though the SaaS CRM market is quiet competitive, picking a case for the analysis is relatively easy since the industry has an obvious leader Salesforce.com Inc. The company is currently very famous and popular among SMEs and usually referred as the inventor of the SaaS model. It satisfies the requirements listed in previous section. Moreover, Salesforce.com has a long story starting 1999 before the dotcom's bubble burst meaning that it actually has some advantages over companies using the ASP model that allowed surviving the crisis. Another positive side of the case is the single focus of the company on the SaaS CRM solution, which makes the analysis easier unlike corporations such as Microsoft and Oracle having large portfolio of software products. Moreover, according to the SaaS Capital Expert Group (2011) Salesforce.com is by far the biggest company in the market, both in terms of revenue and valuation. In fact, its market capitalization is larger than the rest of SaaS companies combined. Therefore Salesforce.com is chosen as the first case in the research.

Case 2 – ASP CRM provider

USinternetworking (USi) is referred in the literature as the largest ASP in terms of customers, revenue and business potential (Kern, Kreijgerb et al. 2002). Company is one of the first dedicated application service providers and had its own data centers in the US, Europe and Asia. Application service provision was a primary source of revenue for the company comprised 85% of total revenue by the year 2000 (USinternetworking 2000). As well as Salesforce.com, USi has a single focus on on-demand software provision. Therefore we chose USinternetworking as the representative ASP case for the case study.

5.3.2 Case information

In Chapter 4 propositions were formed. Verification of every proposition requires specific information and it is described below.

The SaaS providers charge lower prices than Application Service Providers

To analyze this proposition we need information on pricing and service structure of a company. If the SaaS providers pricing proves to be lower at equal service delivery with the ASP offering, then we need to identify the reason of price difference whether it result of certain costs structure or strategic choice. However, we are also aware that comparison of pricing strategies of cases could be frustrated by contextual factors such as inflation etc. Therefore we also conduct comparison based on additional criteria that strengthen our findings on pricing – Total Cost of Ownership (TCO).

Potentially SaaS provider is able to cover wider market compared to ASP

In order to test this proposition we need information on target markets of both companies. Moreover, it could be the case that companies expand their offering over time, which results in wider market for them. Therefore we need to track the changes they have made over time and try to understand the reasons behind decisions they have made.

Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model

Subscriptions and customer support are the primary sources of revenue of companies picked in §5.5. Therefore reaching 'economies of scale' is vital for both of them. 'Economies of scale' in on-demand software delivery business could be reached through different means including well-designed infrastructure, unique technology or efficient application architecture that allows reaching certain levels of efficiency and deploying less hardware or manpower. Therefore comparison of models requires details on technical architecture necessary for delivery of services.

In contrast to the SaaS provider, the ASP usually does not play a role of Software Developer

Analysis of this proposition requires details on value network required to deliver a service and information on roles division in this value network. If proposition proves to be valid, it is necessary to investigate why the company choose a certain role whether it was strategic choice or some other reasons.

The ASP model implies significantly higher initial and ongoing costs than the SaaS model

To test this proposition we need data on subscription revenue and financial costs that company had in order to deliver the service. Then we calculate and compare how much money each company spent in order to generate a dollar in revenue from the main service offering.

The threat of large established packaged software vendors entering the market is significantly higher for Application Service Providers than for SaaS providers

In order to analyze this proposition we need information on type of competitors that company faces, their service offering and current position on the market. After that indentify to what type of companies the main competitors belong.

5.3.3 Case study reports

Case study reports should have similar structure that will allow comparison of the findings. Below an outline for the case analysis is provided.

Individual case studies

1. General description
2. Business model
    a. Findings on Service domain
    b. Findings on Technology domain
    c. Findings on Finance domain
    d. Findings on Organization Domain
3. Conclusion

Cross case analysis

Analysis compares results of the SaaS case and the ASP case.

1. Comparison of business models
2. Analysis of propositions
3. Conclusion

5.4 Interview methodology

In this section we will discuss the interview methodology used in this research to collect data and validated propositions formed in Chapter 4. First of all, the reasons behind decision of picking interviews as data collection method are explained. After that description of interview process, protocol is provided, which is followed by data analysis approach section.

5.4.1 Interview rationale

This research is exploratory in nature and focuses on complex concepts such as SaaS and ASP, which need in depth understanding. Therefore interviews were chosen as one of the data collection methods. The main purpose of interviews in this research is to validate the findings and propositions formed in previous parts. Nevertheless, the understanding of the reasoning behind answers is very important as well, because cloud computing is relatively young and dynamic field where knowledge become outdated quiet fast.
Semi-structured interview approach was taken in order to allow respondents to elaborate on answers while keeping an option to ask structured questions to validate the propositions.

5.4.2 Interview protocol

As have been mentioned before the interview protocol was designed in a way that allows validation of propositions formed in Chapter 4. We have prepared protocol for the purpose of interviews with industry experts, where we focus on differences between SaaS and ASP and possible reasons of low adoption of ASP. Prior to interview, protocol was review and approved by research supervisor. The full Interview protocol could be found in the appendices, while question summary is provided in the table below. Questions were designed based on STOF business model and formed propositions. Interviews took around 45 - 70 minutes each. In order to introduce the topic and let respondents to prepare for the interview, a short letter with problem description and definitions was sent. Each interview started with definitions phase where we assured the same understanding of the main concepts of the research. In addition, each interview was audio recorded with permission of respondents. After the interviews, we made non-literal transcript of each interview that was sent back to respondent for the feedback on accuracy of the summary.

Table 9 Summary of interview questions

Topic	Question
General introduction regarding the factors affected models adoption	1) Why do you think supplier companies using ASP model didn't succeed? 2) Why do you think supplier companies using pure SaaS model are successful today?
Service	3) Do you think ASP and SaaS served different market segments?
Technology	4) Do you think there are differences between models from technological perspective? 5) Which model is more efficient in hardware utilization? Why? 6) What advantages/disadvantages of ASP and SaaS?
Organization	7) What roles SaaS providers and ASP played in their value networks? 8) What partners do they need to provide the service?
Finance	9) Do you think there are any differences in operational and maintenance costs between ASP and SaaS? 10) Do you think there are differences in entry barriers between ASP and SaaS market?
Closed questions	List of statements with which respondent asked to Agree or Disagree

We start with two general questions that could cover any types of factors that affected the adoption of models. The main interview questions are structured according to STOF business model, from where we could derive differences between ASP and SaaS in four domains: service, technology, organization, and finance. After the section regarding four domains of STOF model, we introduce closed questions that consist of propositions formed prior to interview and ask respondents to agree or disagree with them. Although answers for questions in closed section could be derived from previous questions, we still wanted to assure consistency in answers of respondents.

5.5 Conclusion

This chapter describes research methodology used in this project. Particularly it defines the type of information is necessary to obtain for the conduct of case studies. Case selection procedure was described; two companies fulfilled the requirements and were chosen for the study – Salesforce.com and USinternetworking. Moreover, interview approach and interview protocol with the summary of questions asked were presented in details.

6 Research Results

In Chapter 4 we have formed propositions that are going to be tested through case studies and expert interviews. First we start with the case studies and conduct comparison of our findings. It is followed by the Section 6.2 Interview results where we present summaries of conducted interviews and derive conclusion from them.

6.1 Case study results

This section presents results of case studies conducted for two companies: Salesforce.com and USinternetworking. Case studies are structured in accordance to the STOF model and propositions formed in previous chapters. Each case starts with a short company description after which goes analysis of four domains.

6.1.1 Case 1: Salesforce.com Inc

Mark Benioff found Salesforce.com in February 1999. Company's best-known service is the SaaS CRM introduced in February 2000. Starting 2007 they offer platform as a service Force.com, which provides a feature set and technology environment for building and deploying enterprise applications. As of January 31, 2010, the company had 3,969 employees and 97,700 customers all over the world. Company's principal executive offices are located in San Francisco, California with regional headquarters in Dublin, Tokyo and Singapore. Other major offices are in Toronto, New York and London (Salesforce.com 2010). According to Salesforce.com Inc. the system they provide possesses the key attributes of well-designed SaaS listed in §2.4.1 (Salesforce.com 2011). Moreover, the multi-tenant architecture of Force.com platform fulfills the requirements of the fourth maturity level, which proves that modern SaaS vendors (particularly Salesforce.com Inc) are in the final level of Microsoft's Software-as-a-Service maturity model.

Findings in Service domain

Proposition 1 – Pricing
The SaaS providers charge lower prices than Application Service Providers

Currently Salesforce.com offers five editions of CRM software: Contact manager, Group editions, Professional edition, Enterprise edition, Unlimited edition.

Table 10 Salesforce.com Pricing Strategy

Contact manager	**Group edition**	**Professional edition**	**Enterprise edition**	**Unlimited edition**
Contact management for up to 5 users 5$/user/month	Basic sales and marketing for up to 5 users 25$/user/month	Complete CRM for any size team 65$/user/month	Customize CRM for your entire business 125$/user/month	Premier support tailors CRM for your business 250$/user/month
Features				
Accounts & contacts	Includes all Contact Manager features plus:	Includes all Group Edition features plus:	Includes all Professional Edition features plus:	Includes all Enterprise Edition features plus:

Task & event tracking	Opportunity tracking	Jigsaw data services	Workflow & approval automation	Unlimited customizations
Email integration Outlook, Gmail, Lotus	Customizable sales process	Mass email	Sales teams	Unlimited custom apps
Google Apps	Email templates & tracking	Campaigns	Territory management	Unlimited administration services
Mobile access	Google AdWords	Product tracking	Offline access	Designated support account specialist
Content library	Web-to-lead capture	Real-time quotes	Call scripting	Mobile customization & administration
Customizable reports	Lead scoring, routing & assignment	Contract management	Profiles & page layouts	Increased storage limits
	Dashboards	Customizable forecasts	Custom apps & websites	Multiple developer sandboxes
	Salesforce-to-Salesforce collaboration	Customizable dashboards	Developer sandbox	24x7 Premier Support
		Analytics snapshots	Integration via Web Services API	
		Role permissions		
		Ideas community		

Two well-known research organizations did a research on Total Cost of Ownerships (TCO) of Salesforce.com solutions. Gartner (2004) reported that for large enterprises with around 1000 employees within which there are 100 salespeople and 50 customer service and support users, the use of Salesforce.com CRM in first year will results in up to 50% cost savings compared to on-premise counterparts. That occurs mainly because of very low up-front investment into the system. However, in subsequent years Total Cost of Ownership (TCO) will reach the levels of on-premise software (see Figure 10).

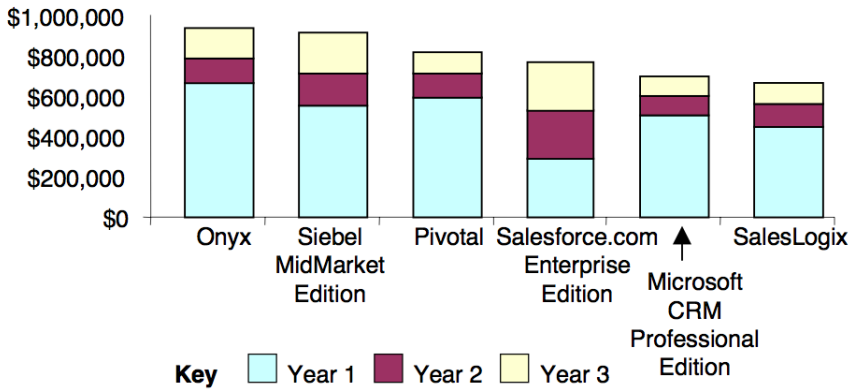


Figure 10 Total Cost of Ownership (Gartner 2004)

Nevertheless, in case of small enterprises with lack of good IT infrastructure in house to deploy and maintain the system, Salesforce.com offers much lower IT hassles and can cost less than on-premise counterparts throughout three years (Gartner 2004).
The Yankee Group studied smaller organizations with from 40 to 300 seats of Salesforce.com (on-demand) and SalesLogix (on-premise), and reported more optimistic results on TCO of Salesforce.com CRM. According to their findings for five year of using the system companies are able to save up to 50% while using Salesforce.com (YankeeGroup 2004) (see Figure 11).

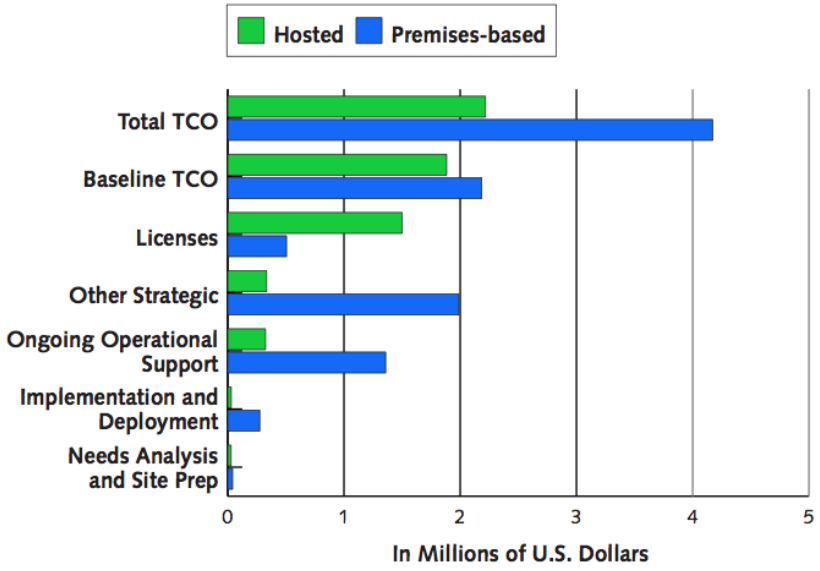


Figure 11 Total Cost of Ownership (YankeeGroup 2004)

Overall, findings from both research organizations suggest that solution from Salesforce.com is the most beneficial for small and medium enterprises.

Proposition 2 – Market segment
Potentially SaaS provider is able to cover wider market compared to ASP

Subscription to Salesforce.com services does not require significant up-front investments in software, hardware, implementation services and IT staff, as customers would have with traditional software solutions. It allows companies to focus on their core business rather than allocating scare resources on handling IT infrastructure. Although the offering appeals more to small medium enterprises since it allows gaining access to technical expertise and "best-of-breed" apps with minimal costs, yet according to our findings Salesforce.com continues to target businesses of any size primarily through direct marketing. For that purpose company has created several editions of the service to address the distinct requirements of businesses of different sizes.

Findings in Technology domain

Proposition 3 – Technical architecture
Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model

In 2009 Salesforce.com revealed that they have 55,000 enterprise customers with 1.5 million individual subscribers and 30 million lines of third-party code running on only 1000 machines, including server mirroring which means that in fact it is only 500 servers (Schonfeld 2009). Consequently, one server per 1500 users with average response time of around 300ms. That level of efficiency is reached through usage of multi-tenant architecture and wide range of other proprietary methods including patented multi-tenant-aware query optimization and shared database etc. Components that facilitate required efficiency for the solution provided by Salesforce.com are depicted on Figure 12. The starting point of Salesforce's technical architecture description will be the shared database.

Software architecture

Most people are used to the idea that data isolation is reached through providing separate databases for every tenant. Although it is the simplest approach, it unfortunately entails very high hardware cost. However, there is another way called Shared database with Shared schema, which is used by Salesforce.com. The basic idea of Shared database method is use of the same database with the same tables to host multiple tenants, where one field for instance 'TenantID' (Figure 13) is usually associated with the appropriate tenant. The advantage of this approach is the lowest hardware costs, although it requires additional development effort in order to ensure data security (Chong, Carraro et al. 2006). The next question is how tenant-specific customization is reached if databases are shared? As have been mentioned in §2.4 the SaaS has metadata-driven architecture. Therefore all the customization (including forms, workflows, reports etc) exists merely as metadata in shared metadata tables with billions of rows and platform's engine uses it to generate the 'virtual' application components at the runtime (Salesforce 2011). However, such massive data structures are very slow and difficult to process, this is where multi-tenant-aware query optimizer takes a lead. Certainly there are more components that enhance the efficiency of the system, but mentioned above are the core ones. All these proprietary methods results for the company in high level performance with minimal hardware and maintenance costs, for instance according to Craig Weissman – Chief Software Architect of Salesforce.com – Salesforce worldwide is running on about 10 databases which are supported by around 50 servers (Schonfeld 2009; Weissman 2009).

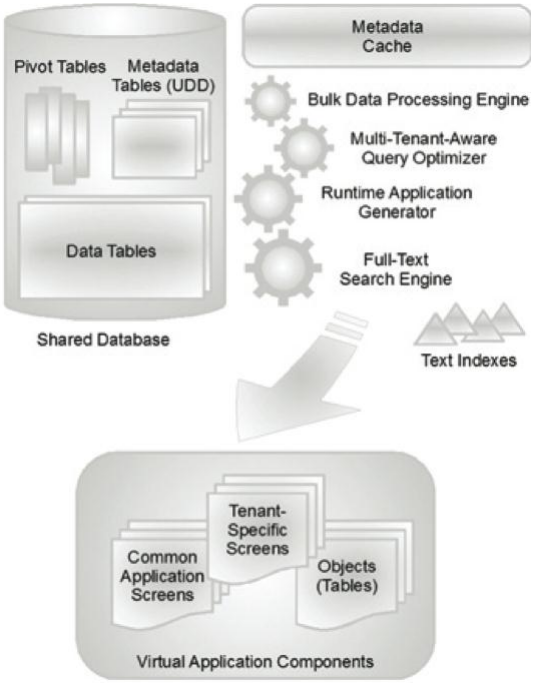


Figure 12 Force.com Architecture (Salesforce 2011)

TenantID	CustName	Address
4	TenantID	ProductID	ProductNam
1	4	TenantID	Shipment	Date
6	1	4711	324965	2006-02-21
4	6	132	115468	2006-04-08
	4	680	654109	2006-03-27
		4711	324956	2006-02-23

Figure 13 Example of Shared Database table (Chong and Carraro 2009)

Findings in Organization domain

Proposition 4 – Value network

In contrast to the SaaS vendor, the ASP usually does not play a role of Software Developer

According to our findings Salesforce.com focuses solely on software development. Therefore in order to deliver the service it is necessary to collaborate with actors who possess critical resources and capabilities. Figure 14 depicts core elements of the value web of Salesforce.com. Since the company is involved only in software development a partnership with Platform provider is required. This role is fulfilled by Equinix Inc. that leases data center hosting facilities in east and west coasts of the United States and Singapore. Equinix facilities are built to the same critical system building codes as hospitals and other vital infrastructures. The facilities are secured by around-the-clock guards, biometric access screening and escort-controlled access, and are supported by on-site backup generators in the event of a power failure. Another important actors involved in the value web are Independent Software Vendors and Third Party software developers. They share the role with Salesforce.com, however the purpose of their involvement is different. The relationships between Salesforce.com and ISVs could be described with the concept of two-sided markets.

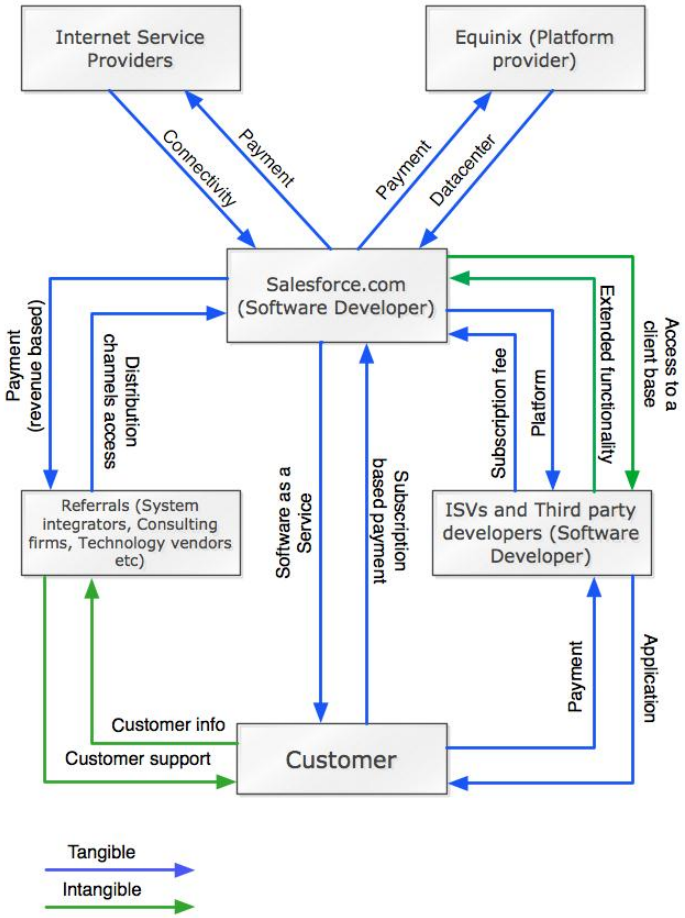


Figure 14 Value network of Salesforce.com

Two-sided markets

Two-sided market is the situation when two different client groups provide each other with network benefits. An example would be the video gaming consoles industry where the success of the gaming system depends on both sides: end-consumers and game developers. Without sufficient number of developers, gamers would not be interested in the system. And other way around without gamers, developers would not be interested in platform. This holds for the software market as well, although the situation with Salesforce.com could be considered as a slightly different since formally two client groups have separate products: SaaS CRM for customers and PaaS Force.com for developers. However, as has been discussed in §2.3 the Software-as-a-Service operates on top of the Platform-as-a-Service. Basically PaaS Force.com allows developers to build applications that extend functionality of the original Salesforce.com CRM and gives access to company's client base. Therefore developers are only interested if there is a sufficient amount of customers using the original system, which to date of Force.com launch was around 41,000 customers.

The next essential element is bandwidth to the Internet that is provided by multiple Independent companies depending on the region. Also Salesforce.com has network of partners that refer customer prospects to them and assist in selling to these prospects in markets where they don't have a large direct sales presence (Salesforce.com 2010). In return, Salesforce.com pay these partners fee based on the first-year subscription revenue generated by the customers they refer (Salesforce.com 2010).

Table 11 Roles in value network of Salesforce.com

Role	Party
Software developer	Salesforce.com Inc.
	Independent Software Vendors (ISVs) and Third party software developers
Internet Service Provider	Multiple independent ISPs
Platform provider	Equinix Inc.
Referrals (including System Integrators, Consulting firms, Technology vendors etc)	Multiple companies including joint venture with SunBridge - Kabushiki Kaisha salesforce.com (known also as Salesforce Japan) based in Tokyo, Japan

Findings in Finance domain

Proposition 5 – Cost

The ASP model implies significantly higher system operating costs than the SaaS model

For this proposition we consider the cost to deliver the service for Salesforce.com. The numbers are taken from the latest 2011 U.S. Security and Exchange Commission (SEC) 10K fillings. According to Income Statement of Salesforce.com (Appendix A), company generates $1,55 billion in subscription revenue by delivering the service to 97,700 customers at cost of only $208 million. Basically, every $1 of revenue costs of about 13 cents for the company. The numbers are very impressive and probably are result of efficient software architecture that decreases operating costs (hardware and maintenance) and allows running the whole system worldwide only on around 1000 servers.

Proposition 6 - Risk

The threat of large established packaged software vendors entering the market of on-demand enterprise software is significantly higher for Application Service Providers than for SaaS providers

The market of enterprise software is highly competitive and fragmented. Salesforce.com defines established packaged software vendors (e.g. Microsoft, Oracle, SAP) as their main competitors. Company is concerned with reluctance of a part of the market to migrate to an enterprise cloud computing application service, due to high financial and personnel investment into traditional enterprise software into their business (Salesforce.com 2010).

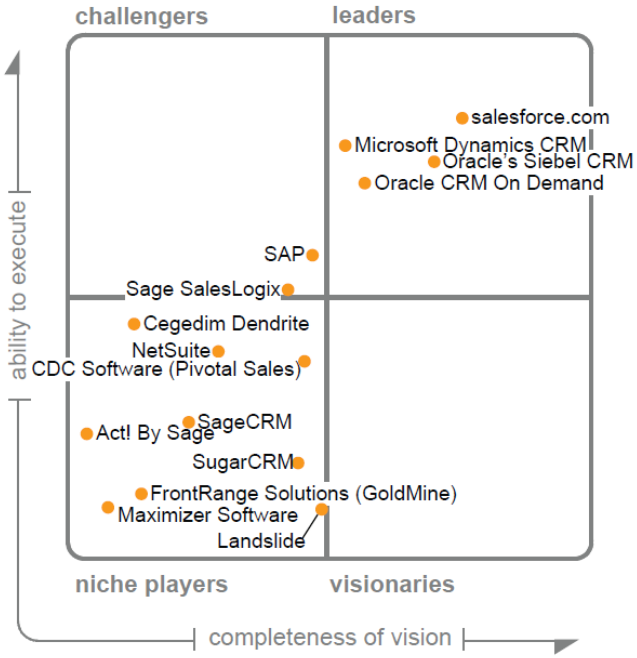


Figure 15 Magic Quadrant for Sales Force Automation (Gartner 2010)

In 2010 Gartner conducted a study of the CRM market where companies were evaluated based on several aspects combined under two generic headers:

- ***Ability to execute*** consists of five criteria: Product/Service, Overall Viability (Business Unit, Finance, Marketing Strategy, Organization), Sales Execution/Pricing, Customer Experience and Operations.
- ***Completeness of vision*** consists of four criteria: Offering (Product) Strategy, Business Model, Innovation, and Geographic strategy.

The result of the research was so called "Magic Quadrant for Sales Force Automation" that grouped companies in four types:

- ***Leaders*** – players that have significant successful customer deployments in North America, EMEA and Asia/Pacific in a wide variety of vertical industries with multiple proof points above 500 users;
- ***Challengers*** – players that have a size to compete worldwide, but lack innovation.
- ***Visionaries*** – players with high innovative potential, but limited in execution.
- ***Niche Player***s – satisfy needs of specific verticals, but demonstrate weaknesses in one or more important areas to support cross-industry requirements, such as complex forecasting or sales effectiveness.

Three vendors with their offerings were identified as CRM market leaders: Salesforce.com, Oracle (Siebel CRM and CRM on demand) and Microsoft (Dynamics CRM). Both competitors of Salesforce.com are large established packaged software vendors with many successful products/services including on-demand versions of their CRM:

- Oracle's CRM on demand offers:
  - On-demand (CRM on demand) and on-premise offerings (Siebel CRM)
  - Single-tenant and multi-tenant versions;
  - Industry specific solutions: medical, pharmacy, high technology, insurance, automotive and wealth management;

  - Full integration with other Oracle apps – e.g. Oracle E-business Suite, Oracle Siebel CRM.
- Microsoft Dynamics CRM:
  - On-demand and on-premise offerings;
  - Integration with other Microsoft products;

Oracle and Microsoft could experience significant growth in on-demand CRM from their already existing large client base, whereas Salesforce.com has to conquer completely new clients. Nevertheless, Olson et al. (2010) suggests that a greater risk to Salesforce's market share comes from a smaller specialized competitors or niche players, since they potentially are able to create software that deals entirely with a specific verticals such as healthcare, education etc. Gartner (2010) also recognized that solutions of Niche player could be a better fit for certain organizations, but may demonstrate weaknesses in more important aspects such as complex forecasting, sales effectiveness or large enterprise support.

6.1.2 Case 2: USinternetworking

USinternetworking (USi) was established in 1998 in the United States. Company offers packaged software solutions that are hosted in datacenters owned by USi and provides access to them over the Internet through proprietary interface called iMAP (Internet Managed Applications Provider). Offering includes a wide range of enterprise applications from leading software vendors such as PeopleSoft, Siebel, Lawson, Microsoft, Oracle, and Sagent. USi takes total responsibility for the secure delivery of the latest enterprise, e-business, and managed web hosting solutions. Company also has stand-alone consulting services. In 2001 USi employed approximately 1,134 full-time employees, of which approximately 78% were technicians and engineers (USinternetworking 2000).

Findings in Service domain

Proposition 1 – Pricing
The SaaS providers charge lower prices than Application Service Providers

USi pricing rates vary widely depending on the type of application, number of users etc. Company considered several standard factors when setting the subscription fee. Those included the up-front consulting requirements, software and software costs, and client training. However, according to Christopher McCleary, CEO of USinternetworking, company usually gets 3-5 years contract with subscription fee varying from $10,000 for e-commerce apps to $200,000 for financial and web-site management software, while on average company charged $30,000 per month (Quinton 1999). That saved the client company 20% over the cost of implementing application in-house. Typically, for USinternetworking required up-front investment $600,000 to $1 million per customer.

Proposition 2 – Market segment
Potentially SaaS provider is able to cover wider market compared to ASP

Originally USinternetworking targeted middle market enterprises and divisions of larger multinational organizations with $50 million to $1 billion in revenue but without the IT depth to support complex advanced networks apps. Rationale behind it was that middle market

enterprises were underserved by large software vendors due to required high up-front costs to obtain the software. Nevertheless, afterwards company faced higher interest from large multibillion-dollar global companies such as Samsung semiconductor group, Excel Capital and US West, whereas smaller companies were still balk at the cost (Quinton 1999). Therefore, in 2000s USi re-focused on larger multinational organizations with 1000+ employees.

Findings in Technology domain

Proposition 3 – Technical architecture
Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model

USi owned four datacenters located in Annapolis, Milpitas, Tokyo and Amsterdam. Two primary datacenters were located in the US where applications were housed, whereas other two served as distribution and mirroring facilities (Roberts-Witt 1999). Servers from Compaq Computer Corp. handled customer running Windows NT, whereas servers from Hewlett Packard Corp. handled customers running Unix (Makris 1999). Each of these facilities is connected to the Internet via Tier 1 Internet Service Providers, including Sprint and UUnet (currently Verizon). Connection speed varied from T-1 (1.5 mbps) to T-3 (45 mbps) (Makris 1999). Typically USi set up virtual private network (VLAN) for a customer through which customers connect to applications. However, option of connecting to a private wide-area network in certain regions for a higher security was available as well. For security reasons, every customer had dedicated port on firewall and Cisco network switch that hooked clients to VLAN or LAN. USinternetworking datacenters provided full isolation of customer data from each other, and only shared medium was after data being backed up (Roberts-Witt 1999).
Figure 16 below depicts the overview of how process of USi renting enterprise applications worked:

1. Customer sends a request via the Internet to use an enterprise app from USi. Interaction of a customer with USi's datacenter occurs through dedicated port in firewall and network switch.
2. The request is received by the nearest datacenter where client authenticated.
3. After client authentication, the request is routed to the appropriate application server.
4. Session of using and sending data between client and application server begins.

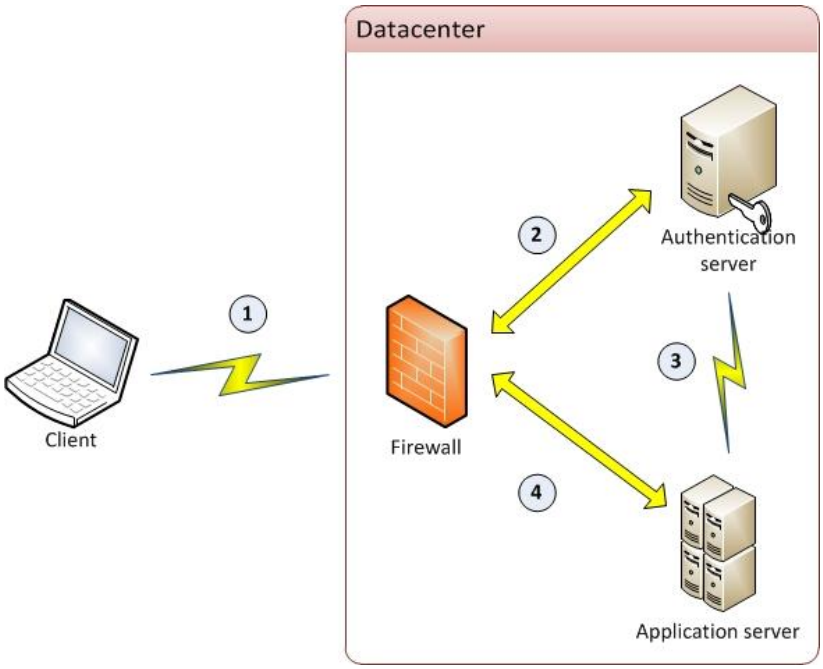


Figure 16 Application Renting Process

Software architecture

As we discussed before USinternetworking focused on delivering packaged software on subscription basis. Typically on-premise software has single tenant architecture that requires running a separate instance of application on dedicated machine/server. However, USi realized that use of virtualization technologies provide better level of datacenter utilization and in 1999 started to use virtualization software from VMware (VMware 2008). Virtualization provides increased utilization rates of the underlying physical hardware from 5-15% to 60-80% by decoupling the physical hardware from the operating system (Cappuccio 2008). In a virtualization environment, a single physical machine runs software that abstracts the physical server resources so that they may be shared between multiple virtual servers (IBM 2007). Figure 17 below depicts differences between regular and virtualized machines.

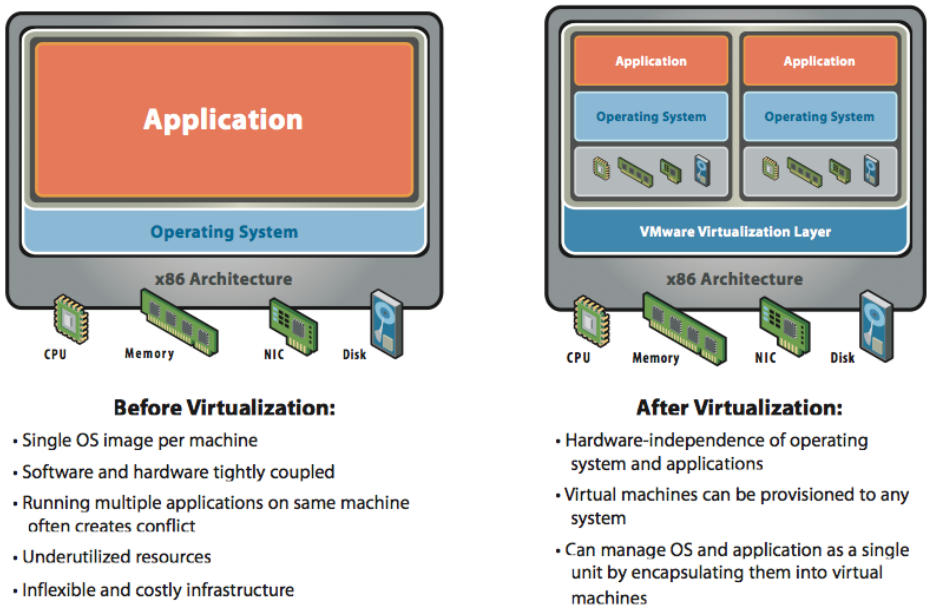


Figure 17 Virtualization (VMware 2006)

Normally there is a single operating system for a machine that owns all hardware resources, on top of which applications run. However, virtualization technologies, particularly VMware specifically suited for x86 architecture allows to abstract hardware from software by creating virtualization layer between them. The deployment of virtual infrastructure is non-disruptive, since the user experiences are largely unchanged (VMware 2006). On top of the VMware layer application service provider (USi) creates isolated virtual containers (sandbox environment) for every client where dedicated instance of application and database run. Above we have considered an example of a single server virtualization, but the same principles are applicable for the entire infrastructure of hardware (Figure 18).

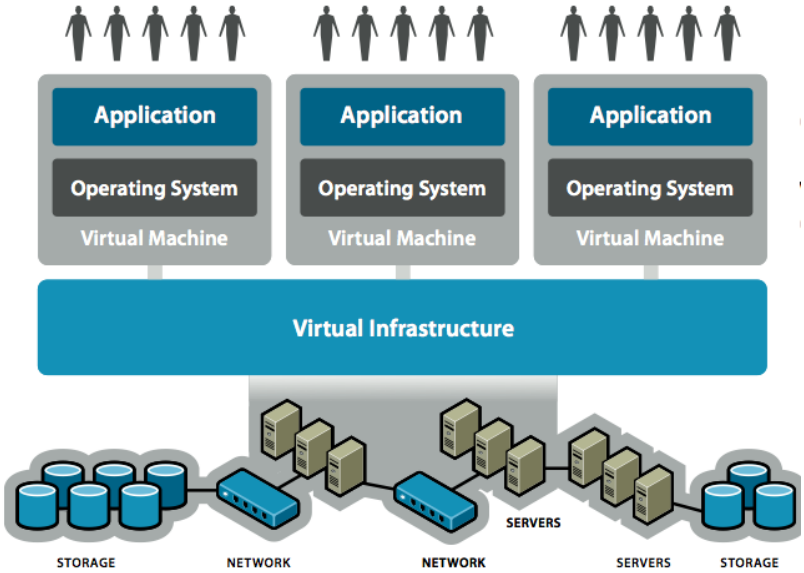


Figure 18 Virtual Infrastructure (VMware 2006)

In this case, layer of abstraction (virtual infrastructure) is created between software and the whole infrastructure represented by storage, servers, and networking equipment (Figure 18). According to Christina Schriver, USi's Director of advanced engineering, USinternetworking virtualized all layers of computing platform: the security, server, storage and network layer, which resulted in true shared resource segregation (VMware 2008).

Besides, virtualization technologies from VMware, USi used software automation tools from Kintana Inc. According to Alex Lobba (Scott 2000), marketing VP of USi, activities such as software installation over the network and configuring the system to specific customer requirements resulted in productivity breakthrough. Solutions from Kintana Inc. were aimed to capitalize on three problems that IT departments faced in 1990s: speeding deployment, minimizing complexity and reducing required manpower.

Both technologies applied, allowed USinternetworking utilize hardware more efficiently with less human intervention in the process. Moreover, USi preserved margins through getting deep discounts on hardware and software (Quinton 1999).

Findings in Organization domain

Proposition 4 – Value network

In contrast to the SaaS vendor, the ASP usually does not play a role of Software Developer

According to our findings USinternetworking fulfills two roles in this web - Platform provider and System integrator. As Platform provider the company owns, manages and operates datacenter required for hosting of solutions from Software developers. Another role of USi is System integrator - obliges the company to hold relationships with customers, assure availability and performance of the service, and adapt software to customer needs. According to the case study done by Kern et al. (2002) USi configures additional functionality of the packaged application by templates and then targets these templates at specific customer requirements.

However company needs partners in delivering service to customers. Figure 19 depicts more detailed view on the value network. First essential partner is Software Developer. USi has formed relationship with well-known software application vendors such as Microsoft, Oracle, PeopleSoft, Siebel and many more. Company had unique agreement with each of them, allowing them to deploy packaged application software as a service to a client without transferring title to the licensed software. Moreover, agreements typically included co-marketing, specialized product training and preferred pricing on the licenses to the software (USinternetworking 2000). USi was certified as a Microsoft Gold Certified Partner for Hosting and Application Services, which granted them early access to product information and inclusion in various business development and marketing activities by Microsoft Corporation.

The next part of the value network is the role of Hardware provider, which was fulfilled by Cisco, Sun, HP, and Compaq. USi used Cisco networking equipment that allowed minimization of hardware incompatibility. Cisco Powered Network relied on redundant network hardware, facilities infrastructure such as power supplies and telecommunications circuits. Hardware necessary for datacenters (e.g. servers) was obtained from Sun, HP and Compaq. Finally, Internet connectivity was provided by multiple ISPs depending on the region, among which was UUNET currently Verizon.

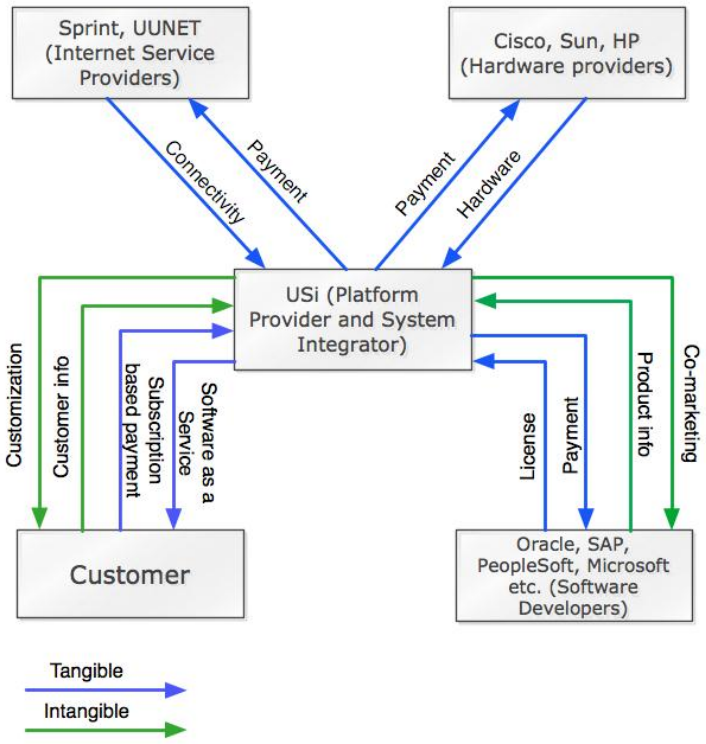


Figure 19 Value network of USinternetworking

Table 12 Roles in value network of USinternetworking

Role	Party
Software developer	Oracle, PeopleSoft, Microsoft, SAP, Siebel, Lawson, Plumtree, Ariba, BroadVision
Internet Service Provider	Multiple independent ISPs
Platform provider	USinternetworking
Hardware provider	Cisco, Sun, HP, Compaq
System integrator	USinternetworking

Findings in Finance domain

Proposition 5 – Cost

The ASP model implies significantly higher system operating costs than the SaaS model

This proposition requires analysis of service delivery costs for USinternetworking. The numbers are taken from the last U.S. Security and Exchange Commission (SEC) 10K fillings dated by 2000 fiscal year. According to Consolidated statement of Operations of USi (Appendix), company generated $109 million in revenue from where subscription revenue comprised 85% - $92 millions. Delivery of their main service offering costs for the company $59 millions. Basically, every $1 of revenue costs of about 64 cents for the company. Even though USi managed to generate revenue from their main service, it had high network, infrastructural, general, administrative and marketing costs in operating loss of $162 millions. Company was making losses for the whole period of independent existence starting from inception date in 1998 and filed for Chapter 11 in 2002. However, the same year re-emerged as private company with significant funding from private equity company, after what in 2006 was acquired by AT&T for $300 millions.

Proposition 6 - Risk

The threat of large established packaged software vendors entering the market of on-demand enterprise software is significantly higher for Application Service Providers than for SaaS providers

Company had very wide range on competitors that included:

Table 13 Competitors in ASP market

Type	Company
Other ASPs	Aristasoft, Breakaway Solutions, Corio, FutureLink, Interliant, Interpath, NaviSite and Telecomputing
On-premise software vendors	Oracle, Siebel and SAP
Local, regional, and national commercial Internet Service Providers and telecommunications companies	AT&T, GTE, Quest, PSINet and Verio
Local, regional and national commercial systems integrators	Andersen Consulting, EDS, IBM and KPMG
Web hosting and managed hosting service providers	Concentric, Digex and Exodus

Among them there were companies with significantly greater market presence, brand recognition, and financial, technical and personnel resources than USi had. Some of

competitor companies such as Accenture, Oracle, AT&T operated in several areas mentioned above.

In the interview CEO of USi McCleary discounted a direct threat from telecommunication companies such as AT&T primarily for financial reasons: saying that investors would not like telcos having co-primary competency of application support (Quinton 1999). However, telcos are still able to obtain these competencies through acquisition or sub-contract of system integrators. Moreover, they have substantially greater network coverage and large existing commercial customer bases. Nevertheless, USi believed that their competence in hosting and managing enterprise software distinguished them from the telecommunication companies (USinternetworking 2000).

The next important group of competitors of USi is system integrators. In contrast to telcos they have extensive experience in providing enterprise software, in addition to wide name recognition. Companies such as PricewaterhouseCoopers, EDS and Andersen Consulting were able to establish strategic relationships with software developers and offer service similar to offerings of USi (USinternetworking 2000).

Competition also comes from traditional software and hardware vendors. In order to enhance their market share companies are able to establish strategic partnerships and deliver competitive service offering. Among them is IBM with their application outsourcing of Lotus Notes, Oracle with their Oracle Business Online – hosted ERP software, and SAP that formed outsourcing organization and developed partnership with leading consulting firms to offer their solution.

6.2 Interview results

In the following part the overview of interviews with industry experts is presented. Detailed transcript of conducted interviews is available upon request.

Expert 1 – Oleg Alexeev
Oleg Alexeev is CTO and co-founder of leading Russian SaaS provider LogneX. Alexeev claims that large enterprises never stopped to use offerings delivered through the ASP model, due to higher customization they require for enterprise software. However, he agrees that approach became much less popular after the dot-com's shakeout and evolved into the SaaS model with multi-tenant architecture. Moreover, the use of multi-tenancy allows easily reach economies of scale and drastically drops the price of the offered services that eventually results in potential to access the wider market that includes large, medium, small enterprises and even separate individuals. According to Alexeev both models use similar core technologies, therefore there is no significant difference in hardware utilization. Efficiency gains of SaaS compared to ASP are mainly result of using single code base that easier and cheaper to maintain. However, multi-tenancy used by SaaS is theoretically less secure than isolation through dedicated virtual machines. Concerning the competition respondent thinks that entry barriers of SaaS are lower compared to ASP.

Expert 2 – Yerlan Ahmetov
Yerlan Ahmetov is Marketing Director at ARTA Solutions. According to Ahmetov ASP model is still in use in niche market and the main reasons why SaaS with multi-tenancy outperformed

ASP are scalability, lower operational and maintenance costs. Respondent perceives these models as two extremes of a continuum of methods to deliver the software. He gave an example of their own solution where single code base serves all the clients, but dedicated virtual machines are still launched for the purpose of data isolation between clients. There are ways to combine these two methods rather going into extremes of one. However he agrees that pure SaaS with multi-tenancy due to lower operational and maintenance costs is better geared towards reaching economies of scale compared to ASP model.

Expert 3 – Martijn Warnier
Martijn Warnier is an Assistant Professor in the Section Systems Engineering, Faculty Technology, Policy and Management, Delft University of Technology. According to Warnier offerings of ASPs are very specialized and very expensive to realize on the large scale. Therefore target market for them is mainly large enterprises, whereas SaaS model is more suitable for small and medium enterprises. In terms of technology, models are quiet close and use similar core technologies such as virtualization that results in higher server utilization rates. However SaaS is more advanced, easier to manage and maintain and rather represents evolved ASP model. Concerning the entry barriers SaaS vendors experience stronger competition due to low entry barriers.

Expert 4 – Sietse Overbeek
Sietse Overbeek is an Assistant Professor at the Faculty of Technology, Policy, and Management, Section of Information & Communication Technology at Delft University of Technology. Overbeek sees the roots of low ASP adoption not in the model itself but rather in external factor – dotcom's bubble burst that resulted in bankruptcy of ASPs clientele and loss of the main revenue source. However, he agrees that the lack of efficiency in use of ASP compared to SaaS with multi-tenancy played an important role in the lower adoption by the mass market. Moreover, pure web-based architecture of the SaaS offering allows utilization rapidly developing web-services and creation of service compositions with wider functionality and improved levels of customization. Concerning the competition Overbeek thinks that SaaS market has lower entry barriers, because companies could only develop an application and run on outsourced infrastructure, whereas ASP implies significant up-front investment into hardware where third party apps had to be hosted resulting in higher entry barriers for ASP market.

6.3 Conclusion

In this section we are going to evaluate propositions with the findings from individual case studies conducted in previous parts. In order to accept proposition as valid it should be supported by both case study and experts' opinion. These results help us to answer following sub-question:

"What are the differences in business models of ASP and SaaS?"

Proposition 1 – Pricing
The SaaS providers charge lower prices than Application Service Providers

The lack of information on pricing structure in case of USinternetworking limits the comparison analysis. However, we have managed to find the average prices for the services

company provided, particularly for high-end applications large size enterprise usually were charged $200,000 per month. According to U.S. International Trade Commission (2010) large size enterprises typically have more than 500 employees. In addition, we adopt an assumption made by Gartner (Gartner 2004) that large enterprise typically needs of around 150 salespeople. It means that USinternetworking charged roughly $1300 per seat/month, which is 5 times more expensive than the Unlimited edition of Salesforce.com CRM.

Price estimates we have made are very rough. Moreover, we are fully aware of macro economical factors such as inflation that also disturb comparison analysis. Therefore it is necessary to provide additional criteria on basis of which we could strengthen our findings. Total Cost of Ownership (TCO) is another way to look on pricing strategies. Companies using service delivered through ASP mode typically saved around 20% over the cost implementing it in-house compared to the first year 50% cost reduction of the SaaS model. Moreover, expert interview results show that four out of four respondents characterized the ASP model as more expensive solution compared to pure SaaS. Therefore based on three independent indicators we accept the proposition.

Proposition 2 – Market segment
Potentially SaaS provider is able to cover wider market compared to ASP

Our findings on case of USinternetworking show that company targeted mainly large enterprises. Although at the very beginning they tried to capture the market of medium size enterprises, the high price of their offering was affordable only for larger organizations. In contrast we have the case of offering from Salesforce.com that targets enterprises of any size and even individual consumers, which was not possible for ASPs to achieve due to technological limitations. The results of case studies support the proposition on market segments. Concerning the interviews three out of four respondents mentioned few times in their answers the difference in target markets between ASP and SaaS, particularly that ASP is geared more towards larger enterprises. Therefore based on results of case studies and interviews we conclude that indeed SaaS providers potentially are able to capture wider market compared to ASPs.

Proposition 3 – Technical architecture
Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model

That is one of the most important propositions, which in fact has direct effect on previous two. Because if a company is able to reach economies of scale fast enough then it would be possible to drop service price and broaden the target market. Our findings show that two models use completely different technical architecture. USinternetworking heavily relied on virtualization and launched separate instance of application and database for each client, which certainly has advantages such as higher security level compared to multi-tenancy, but at cost of lower efficiency for the entire system. All four experts in their answers agreed that both models use similar core technologies such as virtualization, but in slightly different ways. According to respondents the main reason why multi-tenancy is more suitable for reaching economies of scale is the fact that companies have to deal only with one single instance of application and few database that are easier to manage compared to multiple

customized application instances and databases of USinternetworking that had to be managed and maintained each separately. Even though, USi used process automation systems such as Kintana – still maintenance of dedicated virtual machines came at high cost. All industry experts suggested that the SaaS model with multi-tenant architecture is rather the next step after the ASP model. Therefore there is no much of a technological difference between them, but mainly software architectural that results in higher efficiency for the SaaS model. Four out four respondents completely agreed with the proposition and provided very similar reasoning in their answers. Thus we accept this proposition.

Proposition 4 – Value network
In contrast to the SaaS vendor, the ASP usually does not play a role of Software Developer

Case studies showed that USinternetworking was not involved in Software development and provided only hardware, platform, hosting, management and maintenance of third-party enterprise applications. Whereas in case of SaaS provider Salesforce.com, software development was responsibility of the company while hosting facilities were rented from third-party. Case study results support the proposition, while interview results are controversial. Respondents partially supported propositions. Three out of four experts characterized SaaS providers mainly as software developers, which is inline with the first part of proposition. However, two out of four think that in some cases ASPs played a role of software developers as well. Therefore we accept the part of proposition concerning the role of SaaS provider in value network, whereas no definite answer could be given on the role of companies using ASP.

Proposition 5 – Cost
The ASP model implies significantly higher system operating costs than the SaaS model

In order to validate this proposition in both cases we looked into the latest available U.S. Security and Exchange Commission (SEC) 10K fillings. According to financial statements of both companies the main source of revenue is service subscriptions. Nevertheless, there is significant difference in costs of subscription revenue. For instance, USinternetworking spent 62 cents to generate $1 in revenue compared to Salesforce.com with their 13 cents in costs and $1 in revenue. Basically, USi spent almost 5 times more in order to generate the same service subscription revenue. Therefore we can conclude that the ASP model used by USi turned to be much more expensive compared to the SaaS model used by Salesforce.com. Three out of four respondents mentioned in answers that theoretically architectural characteristics of the ASP model imply significantly higher maintenance costs due to necessity to control multiple instance of running application with customization for every client. Although two experts also talked about current developments in virtualization that in practice allow automation of many routine processes related to launch of virtual machines and customization that eventually result in drastic operational cost reduction. Nevertheless, they tend to agree that using only virtualization is still an expensive solution compared to multi-tenancy. Overall, results from both case studies and expert interviews support the proposition.

Proposition 6 - Risk

The threat of large established packaged software vendors entering the market of on-demand enterprise software is significantly higher for Application Service Providers than for SaaS providers

Our findings show that Salesforce.com is experiencing fierce competition from traditional software vendors such as Oracle with their CRM on demand and Microsoft Dynamics CRM. Basically, these companies already entered the SaaS market and managed to take leading positions alongside with Salesforce.com, whereas smaller pure SaaS vendors such as Sugar CRM operate in the niche market. In case of USi, company faces additional competitive pressure from telcos. However, their main competitors are still on-premise software vendors such as Oracle and SAP. Basically, case studies show that companies using both models are under same competitive pressure from traditional software vendors and even from the same companies, particularly Oracle.

The opinion of experts is inline with case study results. Three out of four think that in terms of competition from large on-premise software vendors there is no significant difference between ASP and SaaS. However, all respondents mentioned that entry barriers of SaaS market are lower compared to ASP, due to multi-tenant architecture that allows delivering the service with minimal up-front investments. Therefore because of low entry barriers overall SaaS provider would experience stronger competitive pressure. Nevertheless, results of case studies and interview rejected this proposition.

Six propositions were evaluated, four propositions out of which were supported, one partially supported and one rejected. Table 11 below summarizes results of proposition evaluation.

Table 14 Proposition evaluation results

Propositions	Case studies	Experts	Conclusion
Proposition 1	Supported	Supported	Supported
Proposition 2	Supported	Supported	Supported
Proposition 3	Supported	Supported	Supported
Proposition 4	Supported	Partially supported	Partially supported
Proposition 5	Supported	Supported	Supported
Proposition 6	Rejected	Rejected	Rejected

This page intentionally left blank

7 Conclusions and recommendations

This chapter presents the conclusions we derive from our findings and provide an answer for each research question. Moreover, some scientific and managerial implications are discussed as well as limitations of the research. Finally, we give some recommendations for the future research.

7.1 Main findings

The main objective of this research is to compare two software delivery models and understand possible reasons behind the low adoption of the first wave of application service providers. Our findings serve as a basis to answer the main research and sub-questions outlined below.

The starting point is the first research sub-question:

SQ1: "What are the definitions of Software-as-a-Service model and Application Service Provider model?"

In answering this sub-question, we mainly relied on literature research and interviews that resulted in a set of defining characteristics for ASP and SaaS:

Software-as-a-Service model:

4. The application should be developed from the scratch with usage of the Internet standards, e.g. accessible and operational through web-browser without a need to install any additional software or hardware.
5. Multi-tenancy. The application should be at least on the third level of the SaaS maturity model.
6. On-demand. Customer should be able to gain access to an application for free or on "pay-per-use" basis without any up-front investments in the application license, servers, people and other resources.

Application Service Providers model:

4. Should be possible to adapt traditional on-premise software for delivering as a service through client/server architecture
5. Single-tenancy – running a separate instance of application on dedicated physical/virtual server for every client. The application should be on the first or the second level of the SaaS maturity model
6. On-demand. Customer should be able to gain access to an application for free or on "pay-per-use" basis without any up-front investments in the application license, servers, people and other resources

Although below we present the strict set of characteristics for both, expert interviews showed that these two definitions are rather extremes of a continuum where mix of parts of models is possible. For instance, respondents mentioned certain virtualization solutions from RedHat that allow running single instance of application code while having separate instance of database and virtual server for each client. Basically, virtualization technologies are in

constant development that theoretically could allow companies using ASP model to reach comparable levels of efficiency as SaaS.

In the next sub-question we look into pattern of diffusion and development:

SQ2: "What is the pattern of diffusion and development of software hosting service innovation?"

In answering this question we employed desk and literature research. Three milestones: invention, market introduction and large-scale diffusion of the innovation were identified.

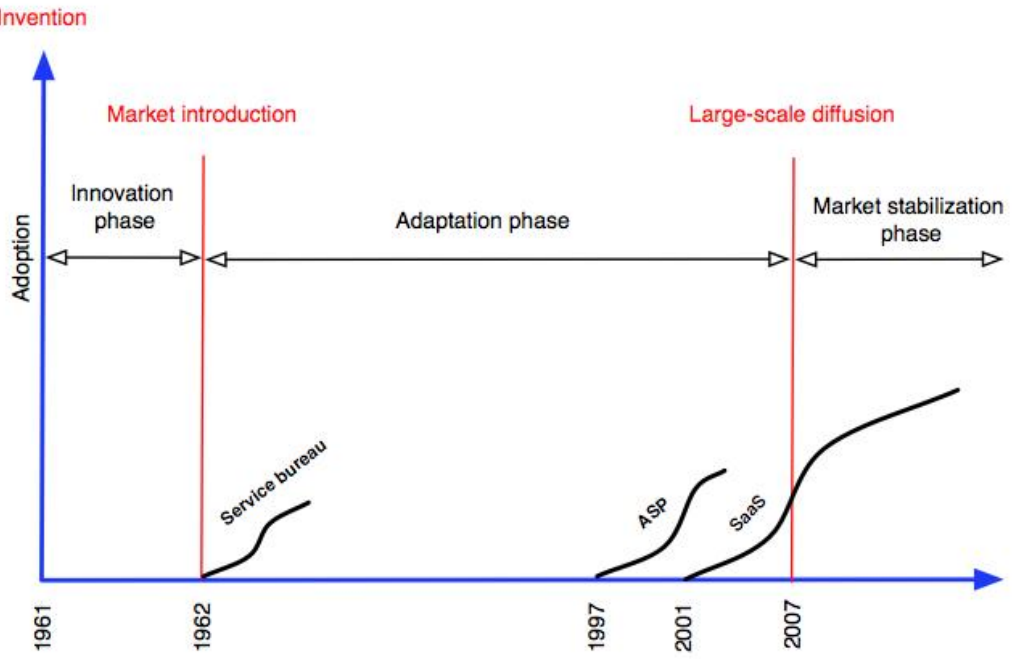


Figure 20 Pattern of diffusion and development of software hosting service category

The invention in 1961 marks the first time when principle of computer time-sharing was demonstrated. Shortly after, in 1962 it was turned into a business in the form of data processing service bureaus. A bureau typically owned mainframe computers and employed a staff of systems professionals. Users were charged rent for the terminal, a charge for hours of connect time, a charge for seconds of CPU time, and a charge for kilobyte-months of disk storage. The second wave of software hosting services emerged in 1997 with Application Service Providers entering the market. Although the model quickly gained popularity in certain market segment, only one large software vendor - Citrix actually adopted the model and started to deliver own solution called iBusiness. In 2000 dot-com's bubble burst resulted in loss of clientele for ASPs and companies had to drop the program, like Citrix did in 2001 or file for bankruptcy like USinternetworking was forced to do just year after (Appendix C).
Meanwhile, in 2001 software delivery emerges in new form of SaaS model. Although, Salesforce.com claims to use models from very launch date in 1999, the official use of a term as a separate concept starts only in 2001, which was taken as an inception date for the SaaS model. By 2007 large software vendors such as Oracle, Microsoft and Google adopted the model and entered the SaaS market with their solutions, what we took as an indicator of the model reaching market stabilization phase.

Next, we are going to cover the next two sub-questions:

SQ3: "What is the method to compare business models of ASP and SaaS?"

SQ4: "What are the differences in business models of ASP and SaaS?"

In answering sub-questions above we employed numerous methods including desk research, literature research, interviews and case studies. First of all, based on theories available we formed theoretical framework for company level analysis. Subsequently, to conduct comprehensive comparison of ASP and SaaS models, the STOF business model specifically designed for a service innovation was adopted. Each software delivery model was broke down into separate components and compared one-by-one. Afterwards propositions on differences in these components were formed and validated.

Table 15 Propositions summary

Domain	Component	Proposition	Conclusion
Service	Pricing	*The SaaS providers charge lower prices than Application Service Providers*	***Supported***
	Market Segment	*Potentially SaaS provider is able to cover wider market compared to ASP*	***Supported***
Technology	Technical architecture	*Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model*	***Supported***
Organization	Value network	*In contrast to the SaaS vendor, the ASP usually does not play a role of Software Developer*	***Partially supported***
Finance	Cost	*The ASP model implies significantly higher system operating costs than the SaaS model*	***Supported***
	Risk	*The threat of large established packaged software vendors entering the market of on-demand enterprise software is significantly higher for Application Service Providers than for SaaS providers*	***Rejected***

Proposition 1 – Pricing

In the research it was confirmed that SaaS providers typically charge lower prices for their services compared to ASPs. Mainly because of multi-tenant architecture SaaS vendors are faster in reaching economies of scale and could drastically drop prices for their solution. Therefore we accept the fact of significant differences in pricing between models.

Proposition 2 – Market segment

The case studies and interviews indicated that companies using ASP model are focusing on larger enterprises whereas SaaS could capture a wider market that includes large, medium, small enterprises and even individual end-consumer. Our findings showed that the main reason of potentially wider market for SaaS providers is ability to drop the prices and make enterprise software more affordable while reaching economies of scale.

Proposition 3 – Technical architecture

Our findings showed that from technological perspective there is not much of a difference in models. Both use virtualization on certain levels, and the only difference is the software architecture they apply. Therefore companies have to decide whether they want to run separate instance of application for each client and achieve data isolation through virtualization that implies high maintenance costs, or use theoretically less secure multi-tenant architecture - run single instance of application that drastically cuts maintenance costs and eventually results in economies of scale.

Proposition 4 – Value network

The case studies and interview showed that indeed typically SaaS providers play a role of Software develop in value network. However, it is still unclear what role companies using the ASP model play in their value web. Although case studies showed that these companies usually are Hardware and Platform providers, industry experts claim that ASPs can also fulfill the role of Software developer. Therefore no conclusion on differences between models could be drawn from this proposition because it is accepted only partially.

Proposition 5 – Cost

In this research it was confirmed that ASPs have significantly higher operational costs compared to SaaS providers. It is mainly because of software architecture they use which requires maintenance of every instance of application running on dedicated virtual machine. In contrast to standardized solution used by SaaS providers that has lower level of customization but allows significantly cut operational costs.

Proposition 6 - Risk

This is the only proposition that was entirely rejected by both sources. It was found that companies using both models are facing comparably equal fierce competition from on-premise software vendors. Although it was also confirmed that due to lower entry barriers of the SaaS market, companies using multi-tenancy experience additional pressure from start-ups.

Differences

This research proved differences between two software delivery models in four business models components (pricing, market segment, technical architecture and cost) that are summarized in the table below.

Table 16 Business model differences summary

	SaaS	**ASP**
Pricing	Lower	Higher
Market segment	Large, medium, small enterprises and individual end-consumers	Large enterprises

Technical architecture	Multi-tenant architecture is quick in reaching "economies of scale"	Single-tenancy and client/server architecture are slow and expensive in reaching "economies of scale"
Cost	Low operational costs	High operational costs

We expected to encounter differences in Risk and Value network of models. However, our study showed that threat from large software vendors entering the market and competitive pressure that companies face is very similar in both cases. Concerning difference in roles companies play in the value network, we couldn't draw any meaningful conclusion because proposition was supported only partially.

Finally, based on answers for sub-questions provided above we are able to answer the main research question:

"Why is Software-as-a-Service model more successful today than the model applied by Application Service Providers?"

Answers for sub-question two showed us that the SaaS is the latest form of software delivery model that managed to reach large-scale diffusion. In contrast to the ASP model that operates in the niche and mainly serves larger enterprises. While answering sub-question four we identified four aspects of business model in which ASP and SaaS differ. We think that these differences are one of the main reasons why the SaaS model is outperforming the ASP model in terms of market adoption. Although four aspects are mentioned we believe that there is a causal relationships between them and in essence it is possible to narrow them down to only few. According to our findings ASP model better fits companies serving larger enterprises due to higher costs that model entails resulting in service prices that SMEs typically cannot afford. In contrast to SaaS providers that are able to reach economies of scale and drop service prices resulting in wider market for the company. Basically, in both cases the key factor is the technology that affects cost and price components and ultimately determines the market segment. The scheme below depicts the logic by which we think the components of both models are interrelated.

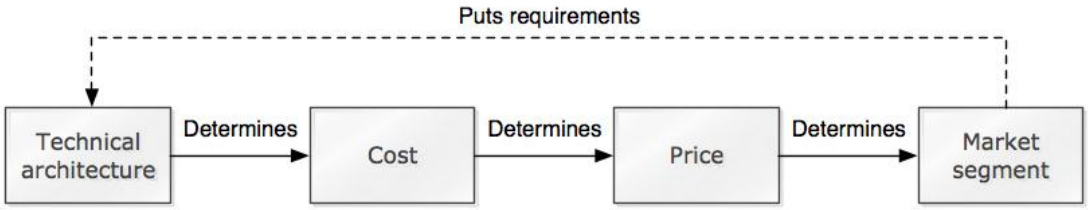


Figure 21 Component Interrelationships

Thus our findings suggest that models differ only in four aspects within which Technical architecture is the core component that determines the rest. Taking that into consideration, we conclude that in terms of business models the main reason of the SaaS model success is Technical architecture and particularly Software architecture that affects the rest of domains and ultimately results in potentially wider target market for the company provider. Consequently making SaaS provider less dependent on a particular customer. However our findings also suggest that the SaaS model is not always a perfect solution for providers or customers. The fact that according to the SaaS model all customers' data has to be stored in single instance of database and run single instance of application raise some security and

reliability issues that could be unacceptable for certain customers dealing with very sensitive data. It could be the case that customers still would like to outsource software services but want to assure higher levels of security and control over data. In that case the ASP offering would be more suitable since the technical and software architecture is comparable with dedicated physical servers with highest level of data isolation. Moreover, recent developments in virtualization technologies allow automation of certain processes and basically able to "fake" multi-tenancy and keeping advantages of dedicated machine by isolating customer data in virtual container. However, the issue of the upkeep of separate customized instances of applications still results in higher costs for providers. Therefore that kind of solution is affordable only by relatively small market with specific needs.
Indeed, the SaaS model is better in reaching "economies of scale" than the ASP model what ultimately results in wider market for companies using it and provides higher rates of adoption and success.

7.2 Methodological Discussion

In this research we mainly focused on micro economical factors that affected the adoption growth of both models. In accordance with the STOF model, the service innovation was decomposed in four domains and important components of business models were compared and highlighted. However, most certainly there are some macro economical factors that played a certain role in the adoption of models. It includes changes in technology infrastructure, economical changes, regulations, social context etc. Therefore we feel obligated to shortly discuss part of them even though they were neglected during the main study.

The first and probably the most important macro economical event that had a huge impact on development of both models is the dot-com's bubble burst in 2000. During which many electronic businesses ran out of capital and were liquidated or acquired. Although our findings suggested that the ASP model is more geared towards serving large enterprises due to costs it implies, according to numerous authors (Ekanayaka, Currie et al. 2003) the main clientele of ASPs were the small medium IT (Information technology) enterprises. Therefore with the dot-com's downturn many of these companies went down and ASP lost huge part of their clientele and revenue sources. Certainly it also damaged the image of electronic services delivered over networks, even though it still was a viable business.

The second external/contextual factor that had an impact on adoption of models is technological infrastructure. We partially covered it in previous parts. Nevertheless, the contextual changes in technologies used around the models should be considered in more detail. Certainly the time variable is very important in comparing ASP and SaaS models. Since the ASP was introduced earlier and faced lack of infrastructure over which services could be delivered. For instance, not many companies had satisfactory level of Internet connectivity. Probably that is also one of the reasons why ASPs mainly served large enterprises – they could afford expensive bandwidth to receive electronic services. Subsequently, connectivity issues caused lower level of reliability and security compared to what the SaaS model had in place at its introduction date.

Moreover, the Internet technologies (protocols, standards and programming languages) were still in very early phase of their development. Whereas at the time of introduction of

the SaaS model, there already were some pure web-based development tools such as Ajax, JavaScript etc that allow efficient use of available bandwidth. However, it is also important to keep in mind that even the development of the Internet technologies could not significantly change the situation with the ASP model, since the architecture applied by it, is based on traditional software that in its nature was not intended to be delivered over the Internet.

Certainly external factors had an impact on cost structures and pricing strategies of companies, since those were applied in different context. That frustrated the comparison analysis we have done, particularly comparison of pricing strategies. Therefore we did the comparison from different angle and took Total Cost of Ownership (TCO) as a mean of comparing two models. Moreover, expert interviews also validated our findings.

Although there is probably many more external factors that interfere results of comparison analysis we have done. We tried to minimize that effect by carefully taking the variables to compare and validated them through additional independent source, as in case of pricing strategy comparison we have described above.

7.3 Scientific Contribution

In this research we tried to make several contributions to the literature. First of all, our study structured knowledge on cloud computing domain. Particularly we looked into Software-as-a-Service and Application Service Provider models. One of the objectives of this research was to demystify the misunderstanding on differences between these two models that is the results of poor use of terms by marketers trying to escape from after waves of dot-com's bubble burst. We have presented the models in simplified way that is suitable for the business model analysis. Although these two software delivery models have significant architectural differences, we have made a conclusion that designing a mixed architecture is possible as well. Moreover, we looked into historical part by indicating the diffusion and development pattern of the service innovation with three critical milestones. The original pattern better (Ortt 2009) suits a product innovation. Therefore we had to adapt it for the case of electronic service innovation by re-defining the third milestone. Further, we used STOF model (Bouwman, De Vos et al. 2008) designed specifically for mobile service innovation and applied it for the case of electronic service in cloud computing domain. Basically, showing that this particular model is applicable for broader types of service innovations.

Secondly, we have made an attempt to integrate patterns of diffusion and development of a service innovation with business model construct by explaining the adoption of the innovation through the prism of business model.

Thirdly, we conducted in depth case studies of two on-demand software providers: Salesforce.com and USinternetworking. Moreover, we have contrasted business models of these companies, discussed advantages/disadvantages and reasons behind occurring differences.

To conclude, our findings contribute to the theory of adoption of electronic service innovation and Software-as-a-Service model in particular.

7.4 Managerial Implications

Results of our study could be useful for managers in several ways. First of all, we would like to look into implications for managers at the customer side. Our findings and conclusions could be used as guidance for companies considering to move from on-premise software and adopt solution from providers using either SaaS or ASP models. We deliberately simplified representation of models and tried to keep technical details on abstract level so managers with any type of background could comprehend it. Moreover, we covered not only the current state of the technology but also look into previous experience with the ASP model and reasons why it practically failed to deliver services to wider markets.

Secondly, there are some managerial implications for provider companies. From the perspective of traditional software vendor, there is always a choice whether do deliver their service over the ASP or SaaS model. Certainly there are barriers when switching from on-premise to on-demand software delivery model. In case of the ASP, it is still possible to adapt legacy software and deliver it on subscription basis by using client/server architecture and isolating instances of application through virtualization. It requires very little effort and re-architecting. Therefore better suits companies with large portfolio of on-premise software and installed base of clients. Although it is an easier way for traditional software vendors, the limited market potential of this model could results in loss of market share in the long-term. The second reason why the ASP model better suits established software providers is the necessity in large initial investments into hardware that would run multiple instances of enterprise applications. Nevertheless, there is always a demand in specific markets for solutions deliver over the ASP model due to current security and reliability issues that the SaaS model entails.
In case of the SaaS model, there is a barrier of software architecture for packaged software vendors. As we have discussed in sub-section 3.2 the SaaS model is considered to be a disruptive innovation. Therefore there are no any other ways for traditional software providers to switch from old to the SaaS model except completely changing the software architecture that will results in chain reaction (as depicted on Figure 21) that ultimately will alter other components of the business model. Another important issue, to be considered by packaged software vendors planning to bring their SaaS solution, is the organizational part and particularly the role in the value network and partners they need to deliver the service. With packaged software, providers were mainly responsible for the development and initial integration of enterprise applications, whereas delivering it on-demand means that provider need to have enough hardware, platform and IT staff to run and support an application. Consequently, partnerships are inevitable since not many companies are able to cover such wide range of roles. As a result complexity of the value network and governance mechanisms increase. Players have to decide on who owns the relationships with customers, organize activities between legally independent organizations et cetera. Table below represents business model components that have to be changed when switching from one model to another.

Table 17 Shifts in STOF business model

	From On-premise to SaaS[1]	
Domain	**From On-premise to ASP**	**From ASP to SaaS**
Service	1. Provider needs to reconsider their *Pricing Strategy* and deliver the service on subscription-basis 2. Value proposition – provider needs to cover wider range of responsibilities that includes deployment and maintenance of enterprise application of each client	1. *Pricing* – provider has to consider drop service pricing to due more efficient software architecture 2. *Market segment* – provider has to consider to cover wider market that includes SMEs and end-consumers
Technology	1. Changes in *Technical architecture* – company needs to obtain hosting facilities to run enterprise software and consider the use of virtualization technologies to enhance server utilization level	1. *Software architecture* – switch to multi-tenant architecture. Legacy single-tenant software is impossible to re-design into multi-tenant. Therefore provider has to build application from the scratch
Organization	1. Complexity of the *Value network* increases 2. More *Actors* involved in the service delivery 3. More *Interactions* between actors	1. Wide target market would require more marketing activities and involvement of additional *Actors* such as Referrals in the value network
Finance	1. Revenue sources – from one-time license fee provider shift towards service on subscription basis that results in more steady cash flow 2. Cost – increase in up-front costs fro each client due to required hosting facilities	3. Cost – drop in costs due to multi-tenant architecture, that would affect the Pricing of the solution

Table above presents the summary of changes in the business model that packaged software providers need to do in order to switch to the ASP or SaaS models. These changes are very complex especially in case of vendors with large installed base of clients and wide range of legacy software. Therefore it is a good opportunity for start-ups to enter the SaaS market with minimal initial costs by developing web-based application from the scratch with use of multi-tenant architecture.

7.5 Future of the Software-as-a-Service

Although in this research the Software-as-a-Service model is presented as the latest form of software hosting service category, we also have noticed that market of the Platform-as-a-Service model is evolving rapidly. In our opinion the main advantage of PaaS over SaaS is the possibility to enjoy network effects with the first one. Moreover, establish the industry standard and lock-in both sides of the market: developers and customers. Industry leaders such as Microsoft, Google and Salesforce.com are already moving in that direction and

[1] Requires changes in components from both columns

starting 2007 introduced their own PaaS solutions. In section 6.3 we have concluded that for the perspective of start-ups the SaaS market has low entry barriers. Consequently, from the perspective of large software providers focusing only on the SaaS market would imply fierce competition from small vendors. Therefore in order to reduce competitive pressure from smaller enterprises probably it would be wiser for large companies to considering entering the PaaS market. In that case it would be possible for these providers to collaborate with independent Software Developers and promote their applications delivered as SaaS, while enhancing the overall value of the PaaS due to network effects.
However, providers still have to decide whether they are ready to move to multi-tenant architecture with all the advantages/disadvantages it entails. Therefore the ASP model is still an option for them to serve certain segment of the market. Although we still think that switching to multi-tenant architecture has more potential in capturing wider markets and ultimately be more profitable.

7.6 Research Limitations

This research has several limitations. First of all the limitations of conducted case studies, despite all the efforts to contact them, company representatives were reluctant to give an interview that could help in enhancing case study results. Therefore we had to heavily rely on desk research that includes case studies, descriptions and interviews prepared and conducted by outsiders. However, we still were able to conduct interviews with industry experts and obtained rich data that directly served as a validation for propositions we have formed.

Secondly, our research is based on limited number of interviews. Although we contacted ten industry experts only four of them were willing to participate in the research. Nevertheless, we have managed to conduct extensive interviews that provided rich qualitative data and increased general understanding of a phenomenon. Yet, we are fully aware of possibility to enhance these findings with more extensive interviews.

Thirdly, there is a risk of circular reasoning with the case of Salesforce.com. The company is considered to be a pioneer and the main promoter of the SaaS model. Therefore many scholars describe the general form of the SaaS based on architecture applied particularly by Salesforce.com. We mitigate that risk by using findings from authors completely unrelated to the cases we study. However, there is still a small chance of these independent authors using Salesforce.com architecture as a foundation for their findings.

7.7 Future Research

In order to address limitations of this study, further research should take several steps. First of all, it would be interesting to prove the differences between models and dependency between components of the business model through quantitative survey study with large sample of industry experts. It is recommended to fairly distribute the sample among academic and industry experts. The first respondents group might include academics doing research in this particular field that are aware of development history and architecture of both models. The second respondents group might include industry representatives from companies using the SaaS or ASP models. We believe that differences in Technology domain cause shifts in other domain, therefore it is strongly recommended to pick respondents who

have technical insights on architecture they use such as Software Engineers, Software Architects etc.
Survey questionnaires could be designed in a way to test the propositions we have formed in this research. Moreover, it would be appropriate to test the importance of particular components of the STOF model and create the set of critical success factors specifically the case of the SaaS model.
Secondly, this particular study mainly focused on a company level and less attention was paid to the context in which innovation developed. Therefore further research could address macro economical factors that played important role in diffusion and development of both models. Moreover, the factors that determine the length of phases in the pattern require investigation as well.

Finally, our research focused only on enterprise software whereas the situation with other types of software could be different. For instance, e-mailing services that are considered to be SaaS as well such as Hotmail and Gmail were introduced to the market earlier and started to diffuse quiet rapidly. Therefore further research could focus on investigating these types of cases that potentially are able to change our vision on the pattern and business models presented in this study.

This page intentionally left blank

References

AberdeenGroup (2001). Corio: Experience and Expertise Alleviate Risk for PeopleSoft ASP Clients, The Aberdeen Group.

Amazon. (2011). "Media Kit: Timeline and History." from http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-corporateTimelines.

Amit, R. and C. Zott (2001). "Value creation in e-business." Strategic management journal **22**: 493-520.

Applegate, L. and R. Montealegre (1991). "Eastman Kodak Company: Managing Information Systems through Strategic Alliances." Harvard Business Review.

Armbrust, M., A. Fox, et al. (2009). Above the Clouds: A Berkeley View of Cloud Computing. Berkeley, UC Berkeley Reliable Adaptive Distribute Systems Laboratory.

Attewell, P. (1992). "Technology Diffusion and Organizational Learning: The Case of Business Computing." Organization Science **31**(1): 1-19.

Ballon, P. (2007). "Business modelling revisited: the configuration of control and value." Info **9**(5): 6-19.

Barney, J. (1991). "Firm resources and Sustained competitive advantage." Journal of Management **17**: 99-120.

Bennet, C. and G. T. Timbrell (2000). "Application Service Providers: Will They Succeed?" Information System Frontiers **2**(2): 195-211.

Bouwman, H., H. De Vos, et al. (2008). Mobile service innovation and business models. Berlin, Springer.

Bouwman, H., E. Faber, et al. (2008). Conceptualizing the STOF Model. Mobile service innovation and business models, Springer.

Bouwman, H., I. MacInnes, et al. (2008). "Dynamic business model framework: A comparative case study analysis."

Cappuccio, D. (2008). Gartner Research. Energy Savings via Virtualization: Green IT on a Budget, Gartner.

Chappel, D. (2011). Introducing the Windows Azure Platform, Microsoft.

Chellappa, R. (1997). Intermediaries in Cloud-Computing: A New Computing Paradigm. Austin, University of Texas.

Cherrytreeco.com (2000). Second Generation ASPs, www.cherrytreeco.com.

Chong, F. and G. Carraro. (2006). "Architecture Strategies for Catching the Long Tail." MSDN Library Retrieved May 11, 2011, from http://msdn.microsoft.com/en-us/library/aa479069.aspx.

Chong, F., G. Carraro, et al. (2006). "Multi-Tenant Data Architecture." MSDN Library, 2011, from http://msdn.microsoft.com/en-us/library/aa479086.aspx.

Citrix. (2011). "Milestones." 2011, from http://www.citrix.com/English/aboutCitrix/milestones.asp?myear=1999.

Commission, U. S. I. T. (2010). Small and Medium- Sized Enterprises: Characteristics and Performance. Washington, U.S. International Trade Commission.

Creasy, R. J. (1981). "The origin of the VM/370 time-sharing system." IBM Journal of Research & Development **25**(5): 483-490.

CSA (2009). Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, Cloud Security Alliance.

Currie, W. L. (2004). "The organizing vision of application service provision: a process-oriented analysis." Information and Organization **14**: 237-267.

De Reuver, M., H. Bouwman, et al. (2009). "Business models dynamics for start-ups and innovating e-businesses." International Journal of Electronic Business **7**(3): 269-285.

Desai, B. and W. Currie (2003). "Application Service Providers: A model in Evolution." ICEC: 174-180.

Dubey, A. and D. Wagie (2007). "Delivering Software as A Service." The McKinsey Quarterly.

Ekanayaka, Y., W. L. Currie, et al. (2003). "Evaluating application service providers." Benchmarking: An International Journal **10**(4): 343-354.

ENISA (2009). Benefits, risks and recommendations for information secu. D. Catteddu and G. Hogben, The European Network and Information Security Agency.

EuropeanCommission (2010). The Future of Cloud Computing. K. Jeffery and B. Neidecker-Lutz, European Comission. Information Society and Media.

Focacci, L., R. J. Mockler, et al. (2005). Application service providers in business, Routledge.

Gartner (2004). The Three-Year Total Cost of Ownership for CRM Software for MSBs, Gartner.

Gartner (2010). Magic Quadrant for Sales Force Automation, Gartner.

Google. (2011). "Chromebook." Retrieved May 18, 2011, from http://www.google.com/chromebook/#features.

Google. (2011). "Revision History." from http://code.google.com/appengine/docs/revision_history.html.

Greschler, D. and T. Mangan (2002). "Networking lessons in delivering 'Software as a Service'—Part I." INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT **12**: 317-321.

Grönroos, C. (1992). Service management and marketing: Managing the moment of truth in service competition. Lexington, Lexington books.

Grönroos, C. (2007). Service management and marketing: Customer management in service competition. Chichester, Wiley.

Group, W. C. W. (2011). "Web Services Glossary." Retrieved May 3, 2011, from http://www.w3.org/TR/ws-gloss/.

Hagendorf-Follett, J. and C. Torode. (2001). "Citrix Drops ASP Program, Pricing." Retrieved May 20, 2011, from http://www.crn.com/news/channel-programs/18818010/citrix-drops-asp-program-pricing.htm.

Hawkins, R. (2001). The Business Model as a Research Problem in Electronic Commerce. Brighton, SPRU – Science and Technology Policy Research. **Issue Report No. 4**.

Heart, T., N. S. Tsur, et al. (2010). "Software-as-s-Service Vendors: Are They Ready to Succesfully Deliver?" LNBIP **55**: 151-184.

Hofacker, C. F., R. E. Goldsmith, et al. (2007). "E-Services: A synthesis and research agenda." Journal of Value Chain Management **1**: 14-44.

IBM (2007). Virtualization in Education. North Carolina, IBM Global Education.

Jeffery, K. and B. Neidecker-Lutz (2010). The Future of Cloud Comptuign, European Comission. Information Society and Media.

Kandukuri, B. R., R. Paturi, et al. (2009). Cloud Security Issues. IEEE International Conference on Services Computing, IEEE Computer Society.

Keller, A. and S. Hüsig (2009). "Ex ante identification of disruptive innovations in the software industry applied to web applications: The case of Microsoft's vs. Google's office applications." Technological Forecasting & Social Change **76**: 1044-1054.

Kern, T., J. Kreijgerb, et al. (2002). "Exploring ASP as sourcing strategy: theoretical perspectives, propositions for practice." Journal of Strategic Information Systems **11**: 153-177.

Kirkham, C. (2006). USinternetworking Agrees to $300 Million Acquisition by AT&T. The Washington Post.

Kraska, T. (2010). Building Database Applications in the Cloud. Zurich, SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH. **Doctor of Sciences:** 227.

Lindquiest, A. B., S. R.R., et al. (1966). "A time-sharing system using an associative memory." Proceedings of the IEEE **54**(12).

Mahowald, R. P. (2011). Market Analysis Perspective: Worldwide SaaS & Cloud Services 2010: New Models for Delivering Software I. Research**:** 38.

Makris, J. (1999). "Software made sensible." Data Communication **28**(1): 28.

Marston, S., Z. Li, et al. (2010). "Cloud Computing - The business perspective." Desicion support systems **51**: 176-189.

Microsoft. (2007). "Microsoft Dynamics CRM 4.0 Launches in International Markets." 2011, from http://www.microsoft.com/presspass/press/2008/jan08/01-30crminternationalpr.mspx.

Microsoft. (2009). "Windows Azure Platform Launch Update." from http://blogs.msdn.com/b/windowsazure/archive/2009/10/29/windows-azure-platform-launch-update.aspx.

MITTechnologyReview. (2009). "Key Players." Retrieved May 20, 2011.

Olson, A., R. Claire, et al. (2010). Strategic report: Salesforc.com Inc., Sector Strategy Group.

Oracle. (2011). "Timeline ", 2011, from http://www.oracle.com/us/corporate/timeline/index.html.

Ortt, R. (2009). "Concept Chapter Pre-diffusion phases."

Osterwalder, A. and Y. Pigneur (2002). An e-Business Model Ontology for Modeling e-Business. 15th Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy. Bled, Slovenia.

Pallis, G. (2010). "Cloud Computing: The New Frontier of Internet Computing." iEEE Internet Computing: 70-73.

Pang, C. (2009). User Survey Analysis: Usage Plans for SaaS Application Software, France, Germany and the U.K., Gartner.

Perry, R., E. Hatcher, et al. (2009). Force.com Cloud Platform Drives Huge Time to Market and Cost Savings. IDC. Framingham, IDC.

Quinton, B. (1999). "Kicking ASP." Telephony **237**(18): 38.

Roberts-Witt, S. L. (1999). "A network's anatomy: USinternetworking." Internet World **5**(31): 70.

Rogers, E. M. (2005). Diffussion of innovation. New York, The Free Press.

Rowley, J. (2006). "An analysis of the e-service literature." Internet research **3**: 339-359.

Ruffer, S. M., D. Yen, et al. (1995). "Client/Server Computing Technology: A Framework for Feasibility Analysis and Implementation." International Journal of Information Managemen **15**(2): 135-150.

SaaSCapital (2011). Leaders and laggards: saas growth and the cost of capital, SaaS Capital.

Salesforce (2011). The Force.com Multitenant Architecture Understanding the Design of Salesforce.com's Internet Application Development Platform, Salesforce.

Salesforce.com (2010). Annual report 2010. Annual Report, SalesForce Inc.

Salesforce.com. (2011). "Company milestones." from http://www.salesforce.com/company/milestones/.
Salesforce.com (2011). The Force.com Multitenant Architecture Understanding the Design of Salesforce.com's Internet Application Development Platform, Salesforce.com Inc.
Schilling, M. A. (2008). Strategic management of technological innovation. New York, McGrawHill.
Schonfeld, E. (2009). "The Efficient Cloud: All Of Salesforce Runs On Only 1,000 Servers." 2011, from http://techcrunch.com/2009/03/23/the-efficient-cloud-all-of-salesforce-runs-on-only-1000-servers/.
Scott, K. (2000). "Innovation: The software factory." InformationWeek **805**: 147.
SIIA (2001). Software as a Serice: Strategic Backgrounder. Washington Software and Information Industry Association.
Simons, L. P. A., J. v. Loon, et al. (2009). "Increasing the Loyalty Effects of eCRM Across the Service Delivery Cycle." Association for Information Systems.
Staten, J. (2008). Is Cloud Computing Ready For The Enterprise?, Forrester.
Sykes, S. (2005). "IBM Completes Acquisition of Corio." 2011, from http://www-03.ibm.com/press/us/en/pressrelease/7567.wss.
Torode, C. (2001). "ASP Consortium To Become Division Of CompTIA." 2011, from http://www.crn.com/news/channel-programs/18818395/asp-consortium-to-become-division-of-comptia.htm.
Turner, M., D. Budgen, et al. (2003). "Turning software into a service." Computer **36**(10): 38-44.
USinternetworking (2000). Form 10-K. Washington, D.C., United States Securities and Exchange Commission.
Utterback, J. M. (1994). "Mastering the Dynamics of innovation." Harvard Business School Press.
Utterback, J. M. and W. J. Abernathy (1975). "A dynamic model of process and product innovtion." Omega **3**(6): 639-656.
Utterback, J. M. and J. W. Brown (1972). "Monitoring for Technological Opportunities." Business Horizons **15**: 5-15.
Vassiliadis, B., A. Stefani, et al. (2006). "From application service provision to service-oriented computing: A study of the IT outsourcing evolution." Telematics and Informatics **23**: 271–293.
VMware (2006). Virtualization overview. White paper. Palo Alto, VMware.
VMware (2008). USinternetworking, Inc., an AT&T company. Customer snapshot. Palo Alto, VMware.
Vouk, M. A. (2008). "Cloud computing - issues, research and implementations." Journal of Computing and Infromation Technology **16**(4): 235-246.
W3C. (1991). "The Original HTTP as defined in 1991 " Retrieved June, 3, 2011, from http://www.w3.org/Protocols/HTTP/AsImplemented.html.
Waters, B. (2005). "Software as a Service: A look at the customer benefits." Journla of Digital Asset Management **1**(1): 32-39.
Weill, P. and M. R. Vitale (2001). Place to Space: Migrating to E-business Models. Boston, MA, Harvard Business School Press.
Weissman, C. (2009). Craig Weissman Salesforce.com's Chief Software Architect
Wheale, P. R. and L. H. Amin (2003). "Bursting the dot.com 'Bubble': A Case Study in Investor Behaviour." Technology Analysis & Strategic Management **15**(1).

Xin, M. and N. Levina (2008). Software as a Service model: Elaborating Client-Side Adoption Factors. New York, New York University.

YankeeGroup (2004). Understanding Total Cost of Ownership of a Hosted vs. Premises-Based CRM Solution, YankeeGroup.

Yin, R. (2002). Case Study Research: Design and Methods, Sage Publications.

This page intentionally left blank

Appendix A - Salesforce.com Consolidated Statements of Operation

	Fiscal Year Ended January 31,		
	2011	2010	2009 (1)
Revenues:			
Subscription and support	$ 1,551,145	$ 1,209,472	$ 984,574
Professional services and other	105,994	96,111	92,195
Total revenues	1,657,139	1,305,583	1,076,769
Cost of revenues:			
Subscription and support	208,243	159,172	127,082
Professional services and other	115,570	98,753	93,389
Total cost of revenues	323,813	257,925	220,471
Gross profit	1,333,326	1,047,658	856,298
Operating expenses:			
Research and development	187,887	131,897	99,530
Marketing and sales	792,029	605,199	534,413
General and administrative	255,913	195,290	158,613
Total operating expenses	1,235,829	932,386	792,556
Income from operations	97,497	115,272	63,742
Investment income	37,735	30,408	22,774
Interest expense	(24,909)	(2,000)	(107)
Other expense	(6,025)	(1,299)	(817)
Income before provision for income taxes and noncontrolling interest	104,298	142,381	85,592
Provision for income taxes	(34,601)	(57,689)	(37,557)
Consolidated net income	69,697	84,692	48,035
Less: Net income attributable to noncontrolling interest	(5,223)	(3,973)	(4,607)
Net income attributable to salesforce.com	$ 64,474	$ 80,719	$ 43,428

Appendix B - USinternetworking Consolidated Statements of Operation

	Period from January 14, 1998 (date of inception) through December 31, 1998	Year ended December 31, 1999	Year ended December 31, 2000
	(in thousands, except per share data)		
Statement of Operations Data:			
Revenue	$ 4,122	$ 35,513	$ 109,544
Costs and expenses:			
Direct cost of services	3,425	23,570	69,609
Network and infrastructure costs	2,186	16,239	26,718
Selling, general and administrative	25,239	63,999	101,944
Non-cash stock compensation expense	231	10,351	20,998
Depreciation and amortization	3,180	22,480	52,687
Total costs and expenses	34,261	136,639	271,956
Operating loss	(30,139)	(101,126)	(162,412)
Other income (expense):			
Interest income	367	4,115	8,813
Interest expense	(2,681)	(6,307)	(21,361)
Net loss	$ (32,453)	$ (103,318)	$ (174,960)

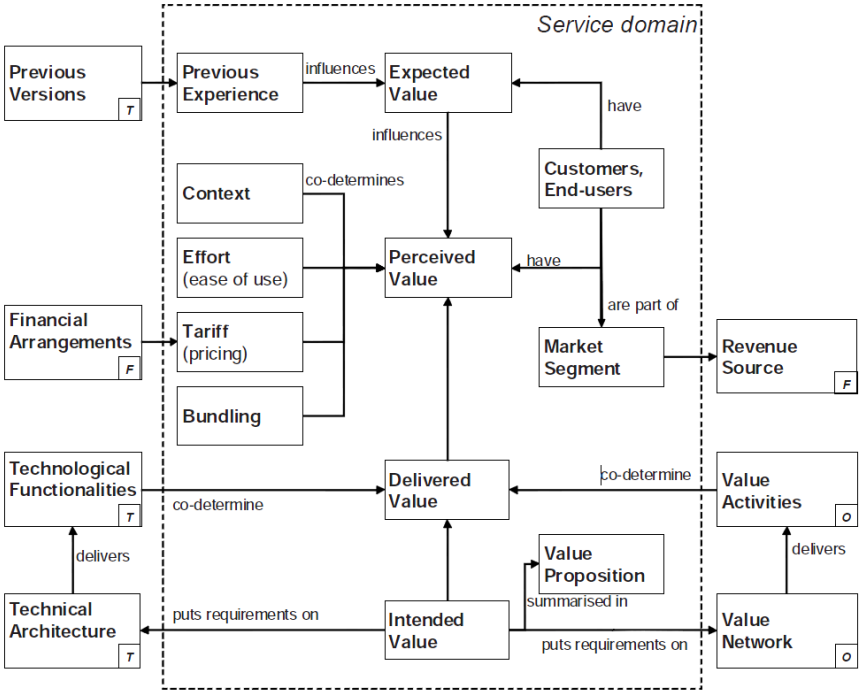
Appendix C - Diffusion and development pattern of software hosting services innovation

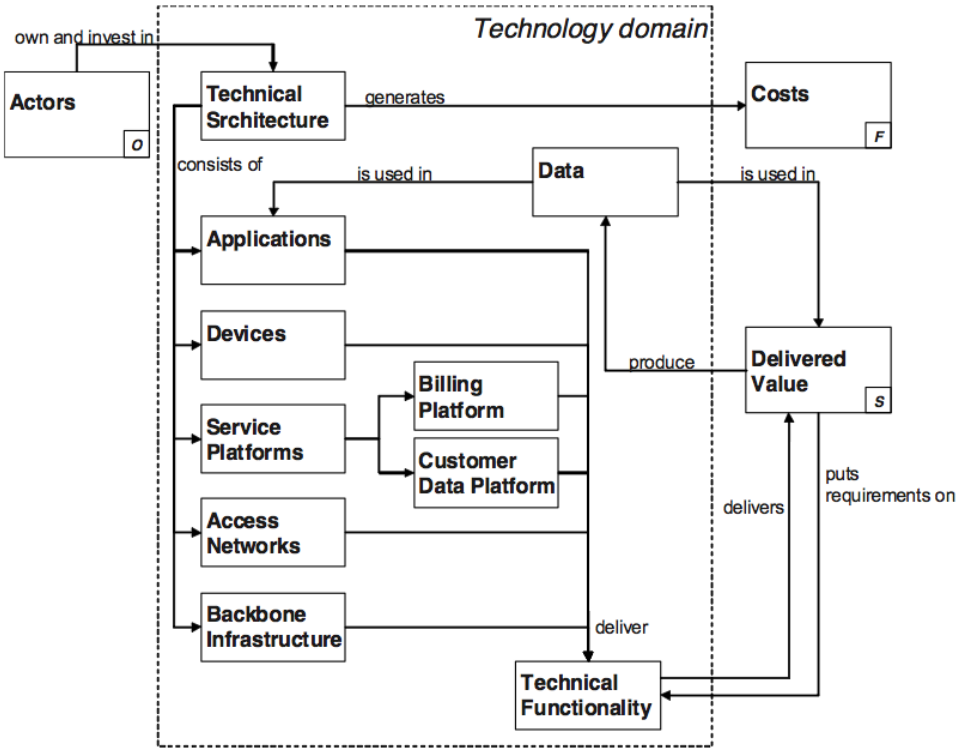
Key events	Year	Remarks	References
	1961	John McCarthy lecture at M.I.T. Envision that «computation» might someday organized as a public utility	(Pallis 2010)
Invention	1961	Compatible Time-Sharing System (CTSS) - first generation time – sharing system developed at M.I.T. A normal batched job stream was run as background to keep the computer busy, while several users could enter commands to prepare, execute and terminate their programs. The machine directly responded in real-time	(Creasy 1981)
Market introduction	1962	Emergence of computer or data processing service bureaus. A bureau typically owned mainframe computers and employed a staff of systems professionals. Users were charged rent for the terminal, a charge for hours of connect time, a charge for seconds of CPU time, and a charge for kilobyte-months of disk storage.	(Attewell 1992)
	1964	The Control Program/The Conversational Monitoring System (CP/CMS) – second-generation time-sharing system. Originally called a pseudo-machine time-sharing system, CP/CMS was named a Virtual Machine system.	(Creasy 1981)
	1966	CP-40/CMS on IBM System/360 Model 40 – first operating system that implemented complete virtualization. CP-40 was run only on unique hardware in Cambridge	(Lindquiest, R.R. et al. 1966)
	1967	CP-67/CMS on IBM System/360 Model 67 – the first widely available virtual machine architecture	(Creasy 1981)
	1969	The U.S. department of defense commissions ARPANET – the first operational package switching network	(Focacci, Mockler et al. 2005)
	The mid 1980s	Inception of Client/Server computing. Corporations start to install LANs (Local Area Network) to connect stand-alone PCs	(Ruffer, Yen et al. 1995)
	1989	«Kodak effect» - Strategic choice of Kodak to outsource Information Technology (IT)	(Applegate and Montealegre 1991)
	1991	The first documented version of the Hypertext Transfer Protocol (HTTP) is known as HTTP 0.9	(W3C 1991)
	1997	Traver Gruen-Kennedy devises the concept of the ASP model and takes it to Citrix Systems	(Focacci, Mockler et al. 2005)
	1997	The first scholarly use of the term «cloud computing»	(Chellappa 1997)
	1998	The largest Application Service Provider - USinternetworking was found	(Kern, Kreijgerb et al. 2002)
	1998	Leading ASP – Corio Inc. was found	(Kern, Kreijgerb et al. 2002)
	1999	Citrix Systems and Traver Gruen-Kennedy found the ASP Industry Consortium that aims to educate	(Focacci, Mockler et al. 2005)

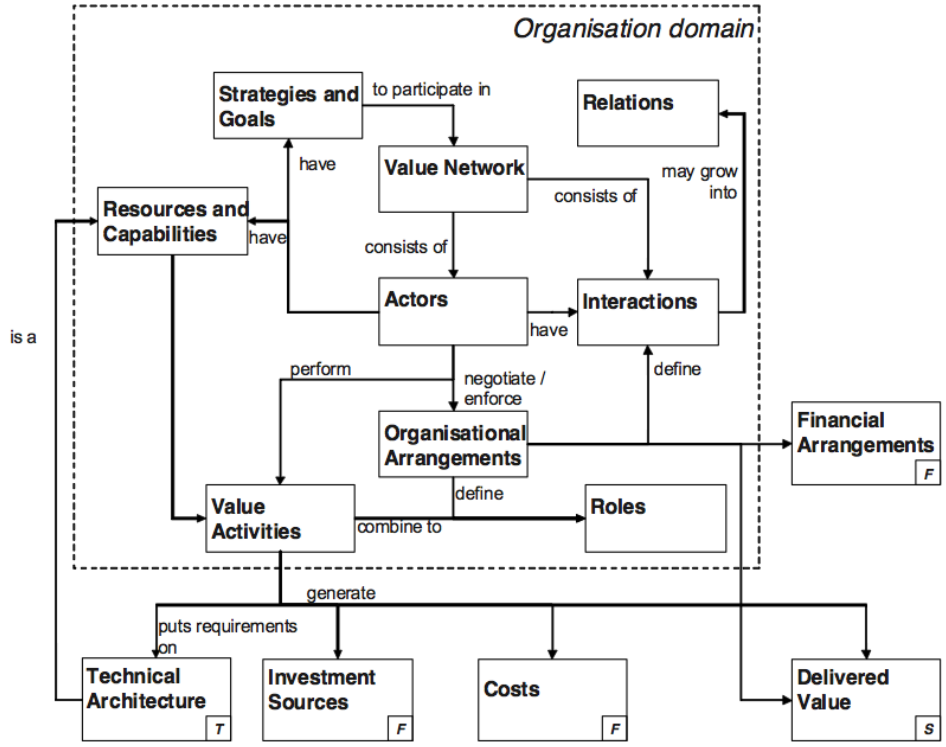
		the marketplace, develop common definitions, sponsor research etc. Other founding member AT&T Corp., Cisco Systems Inc., Compaq Computer Corp., GTE Corp., IBM Corp., Sun Microsystems Inc., and UUNET Technologies.	
	1999	Formation of Internet Business Unit – Citrix iBusiness	(Citrix 2011)
	1999	Launch of SalesForce.com (currently the largest and the most successful SaaS vendor)	(Salesforce.com 2011)
	2000	The dot.com bubble burst. Many dot.coms ran out of capital and were liquidated or acquired	(Wheale and Amin 2003)
	February 2001	The first formal use of the SaaS acronym	(SIIA 2001)
	November 2001	Citrix Consortium drops ASP Program, because the company found it difficult to support hundreds of smaller ASPs from a time and financial perspective	(Hagendorf-Follett and Torode 2001)
	December 2001	ASP Industry Consortium joined CompTIA a non-profit trade association representing interest of the information technology industry	(Torode 2001)
	2002	The largest ASP USinternetworking files for Chapter 11 bankruptcy protection	(Kirkham 2006)
	2002	USinternetworking reemerged as a private company, with significant funding from a private-equity firm	(Kirkham 2006)
	July 2002	Amazon launches new Amazon Web Services (AWS). The most well-known services within AWS are Amazon EC2 and Amazon S3 (one of the first and the most successful utility computing and cloud computing services)	(Staten 2008; Amazon 2011)
	March 2005	IBM completes acquisition of Corio	(Sykes 2005)
	2006	Salesforce.com launches AppExchange and development platform	(Salesforce.com 2011)
	March 2006	Amazon launches Amazon Simple Storage Service (S3) – essential part of AWS; provides storage through web interfaces	(Amazon 2011)
	August 2006	Beta launch of Amazon Elastic Compute Cloud – central part of AWS; allows users to rent virtual computers; allows scalable deployment of apps	(Amazon 2011)
	September 2006	USinternetworking was acquired by AT&T Corp. for $300 million	(Kirkham 2006)
Large-scale diffusion	2007	SalesForce.com Inc. launches Force.com – platform as a service; allows development of multi-tenant apps for SalesForce.com; requires knowledge of proprietary programming language Apex	(Salesforce.com 2011)
	2007	Release of on demand version of popular Oracle CRM	(Oracle 2011)

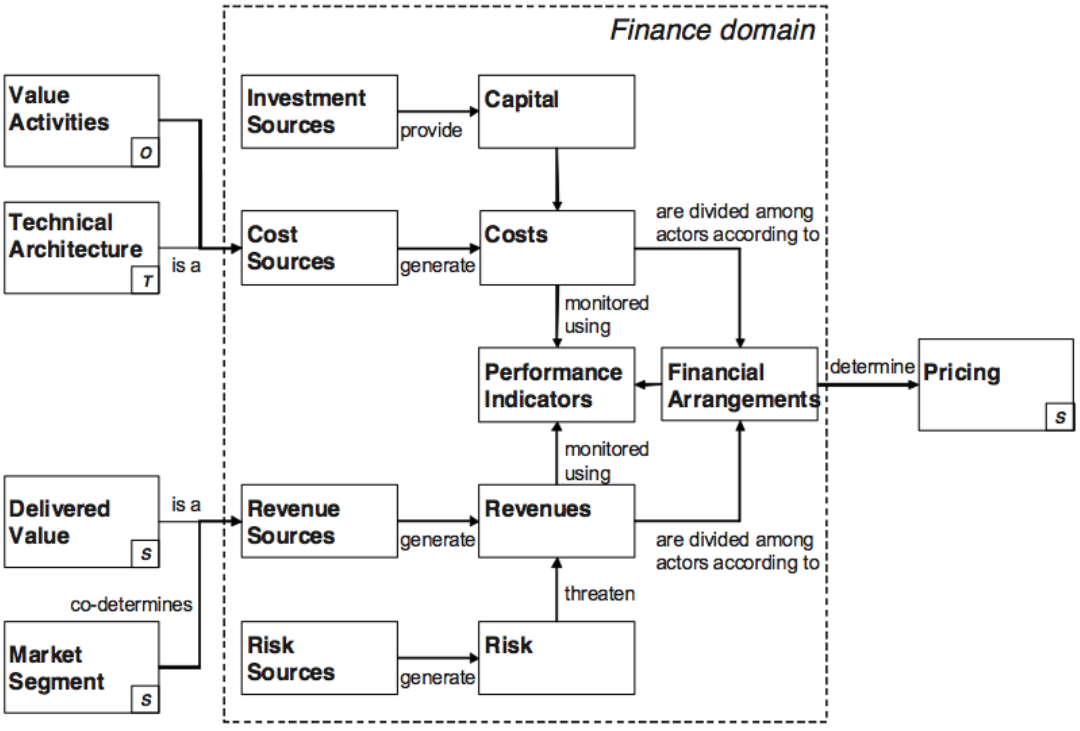
	December 2007	Release of popular Microsoft CRM 4.0 incorporates on-demand version	(Microsoft 2007)
	April 2008	Launch of Google App Engine – platform as a service; allows running web apps on Google's infrastructure; provides two environments for building apps: Java and Python.	(Google 2011)
	July 2009	Both the enterprise and consumer versions of Gmail, Google Calendar, Google Docs and Google Talk are now out of beta	(Google 2011)
	February 2010	Windows Azure Platform commercially available – platform as s service; offers more flexibility then its competitors; allows to reuse most existing Windows code	(Microsoft 2009)
	15 June 2011	Release of Chromebook – a laptop by Google and Samsung with Chrome OS on board (Cloud-based Operational System)	(Google 2011)

Appendix D – Descriptive models of four domains






Organisation domain
Strategies and Goals
to participate in
Relations
have
Value Network
may grow into
consists of
Resources and Capabilities
have
consists of
Actors
Interactions
have
is a
perform
negotiate / enforce
define
Organisational Arrangements
Financial Arrangements
F
define
Value Activities
combine to
Roles
generate
puts requirements on
Technical Architecture
T
Investment Sources
F
Costs
F
Delivered Value
S


Finance domain
Value Activities
O
Investment Sources
provide
Capital
Technical Architecture
T
is a
Cost Sources
generate
Costs
are divided among actors according to
monitored using
Performance Indicators
Financial Arrangements
determine
Pricing
S
monitored using
Delivered Value
S
is a
Revenue Sources
generate
Revenues
are divided among actors according to
co-determines
threaten
Market Segment
S
Risk Sources
generate
Risk

Appendix E - Interview protocol

Topic: Application Service Provider (ASP) and Software-as-a-service (SaaS) business models comparison

Goal: The objective of this research is to compare two models of software delivery (SaaS and ASP) and understand why the first wave of application service providers did not succeed and reach large-scale diffusion. In addition, we aim to develop a list of aspects of a business model to which start-ups have to pay the most attention when developing and launching software-as-a-service business. Moreover, marketing strategies and inconsiderate use of buzzwords by numerous companies confused the software market and even scholars. Therefore it is important to demystify and clarify terms and technologies used by the industry.

Background information
In 1990s companies such as USinternetworking, Citrix Systems, and Corio made significant investments and committed to the ASP model. However, the model did not reach large-scale diffusion and was replaced by the currently very successful SaaS model. Therefore we would like to compare models through the prism of a business model construct.

SaaS – Software as a Service model
Time frame: 1998 - recent
Characteristics:

- Subscription-based
- ***Multi-tenancy*** - data isolation achieved through special application design
  Multi-tenancy allows a single instance of an application software to serve multiple clients, which results in better utilization of a system's resources. Consequently, the addition of another client does not require installation of a separate hardware and software, which at a certain point results in economies of scale for a provider.
- Pure Web-based applications – application is built from scratch on the Internet

ASP – Application service provision
Time frame: 1998 - 2004
Characteristics:

- Subscription-based
- Single-tenancy - data isolation achieved through virtualization
  Virtualization – presents an emulated computing environment where components are abstracted enabling each customer application to appear to run on a separate physical machine
- Host existing traditional on-premise software from vendors such as Oracle, PeopleSoft, SAP etc.
-

Recording confidentiality
I would like to ask your permission to make recording during the interview. Since it can help me to get all the details that are important for the analysis in later stage of the research.

Moreover, it will help to focus more on the conversation without necessity to take frequent notes.
The contents of the interview will be processed anonymously. The outcome of this research will be shared with you.

Data operationalization
The questions below are structured in five groups: Service domain, Technology domain, Organization domain and Finance domain. Grouping is based on STOF business model developed in TU Delft. Prior to the interview, business models of ASP and SaaS were compared and several propositions on model differences were formulated. Consequently, outcome of the interview with industry experts is expected to check the validity of these propositions and highlight the most important aspects of business models.

Open Questions

Generic Intro

1. Why do you think supplier companies using pure ASP model (Usinternetworking, Corio) did not succeed?
2. Why do you think supplier companies using pure SaaS model (Salesforce.com, Sugar CRM) are successful today?

Service domain

3. Do you think that ASP and SaaS models served different market segments?
4. From customer perspective, do you think that the usage of ASP or SaaS models by companies resulted in different customer satisfaction?

Technology domain

5. From technological perspective and particularly technical architecture, do you think ASP and SaaS models have any differences?
   Types of possible differences:
   a. Software architecture
   b. Technical functionality
   c. Backbone infrastructure
6. Which model do you think is more efficient in hardware (datacenters, servers) utilization? Why?
7. What are the advantages/disadvantages of using virtualization as method of data isolation for enterprise apps?
8. What are the advantages/disadvantages of using multi-tenancy as method of data isolation for enterprise apps?
9. Do you think that relatively low Internet connection speed in the late 1990s and early 2000s had a significant impact on customer satisfaction?

Organization domain

10. Companies usually need partners or network of partners in order to deliver such a complex service:
    a. What role application service provider plays in this network (e.g. software developer, hardware provider, Internet service provider etc)?
    b. What types of partners ASP usually needs?
    c. What role SaaS provider plays in this network (e.g. software developer, hardware provider, Internet service provider etc)?
    d. What types of partners SaaS provider usually needs?
    e. What are the differences in roles fulfilled by ASP and SaaS vendors in a value network?

Finance domain

11. From a company perspective, do you think there are any differences in financial costs when using ASP or SaaS models (e.g. operational, maintenance cost)?
12. Competition in ASP and SaaS markets:
    a. Do you think there is any difference in entry barriers between ASP and SaaS markets?
    b. Do you think that the threat of traditional on-premise software vendors is somehow different between ASP and SaaS models? What about the threat from Independent software vendors?

Closed questions

Please indicate whether you *Agree/Disagree* with a statement:

1. The ASP model results is lower customer satisfaction than the SaaS model *Agree/Disagree*
2. Typically SaaS providers charge lower prices than Application Service Providers *Agree/Disagree*
3. Technical architecture of the SaaS model is better in reaching 'economies of scale' than technical architecture of the ASP model *Agree/Disagree*
4. The ASP does not play a role of Software Developer, whereas the SaaS vendor is usually is a Software Developer *Agree/Disagree*
5. The ASP model implies significantly higher hardware and operational costs than the SaaS model *Agree/Disagree*
6. The threat of large established packaged software vendors entering the market is significantly higher for Application Service Providers than for SaaS providers *Agree/Disagree*