8-2008

The Pricing Strategy Analysis for the Software-as-a-Service Business Model

Dan MA
Singapore Management University, madan@smu.edu.sg

Abraham SEIDMANN
University of Rochester

DOI: https://doi.org/10.1007/978-3-540-85485-2_8

Published in GECON 2008, LNCS 5206, pp. 103–112, 2008.
https://doi.org/10.1007/978-3-540-85485-2_8

The Pricing Strategy Analysis for the "Software-as-a-Service" Business Model

Dan Ma[1,*] and Abraham Seidmann[2]

[1] School of Information Systems, Singapore Management University, Singapore
School of Information Systems, Singapore Management University,
80 Stamford Road, Singapore 178902
madan@smu.edu.sg
Tel.: +65 6828 0926
[2] University of Rochester, United States

Abstract. The Software-as-a-Service (SaaS) model is a novel way of delivering software applications. In this paper, we present an analytical model to study the competition between the SaaS and the traditional COTS (Commercial off-the-shelf) software. The main research goal is to analyze the pricing strategy of the SaaS in a competitive setting. The model captures the most salient differences between the SaaS and COTS, including their distinct pricing structures, user initial setup costs, system customization levels, and delivery channels. We find that the two could coexist in a competitive market in the long run, and more importantly, we show how the SaaS could gradually take over the whole market even when its quality is inferior. Surprisingly, our analysis shows that the SaaS should raise (reduce) its prices when its software quality declines (increases) over time (in the relative sense).

Keywords: the SaaS business model; pricing strategy; competition; the COTS software.

1 Introduction

The Internet has enabled a new business model for software providers: the Software-As-A-Service (SaaS) model. The providers could bundle software applications, an IT infrastructure, and all necessary support services and deliver them to users across a network when users have a demand for them. Meanwhile, the providers should store the software system and users' data in a central location and are in charge of daily software maintenance, data backups, software upgrades, and security management. Hence, users obtain and pay for the final computing utility on demand. Such a business arrangement is totally different from the conventional delivery model for software applications. Traditionally, most software has been delivered as commercial off-the-shelf (COTS) products.[1] The provider sells the software application to users

---

* Corresponding author.

[1] A COTS product is a commercial software application that "is designed to be easily installed and to interoperate with existing system components" (see http://whatis.techtarget.com). Almost all software bought by the average computer user fits into the COTS category, such as operating systems, office product suites, word processing, and e-mail programs.

and helps to install it on users' sites. The users possess the full ownership of the software, and must provide IT infrastructure, hardware, and support services in order to enable continuous use of the software.

The SaaS is experiencing fast growth. The AMR Research reports that the on-demand software market is growing more than 20% a year, compared with single-digit growth in traditional software (Lacy 2006). It is expected to reach $10 billion in annual revenue by 2009, up from $1.5 billion in 2006 (Pallatto 2006). To many users, the SaaS constitutes an attractive alternative to the traditional COTS solution. The recent study by *InformationWeek* indicates that 29% of the 250 business technology pros surveyed were using at least one licensed application hosted by a provider and accessed over the Internet, and 35% were planning to buy software that way, or were considering it. More interestingly, interest is found not just among small companies. Instead, 55% of the respondents have annual revenue of more than $100 million, and a third have more than $1 billion in revenue (InformationWeek 2007). Large organizations, such as Amazon.com, Cisco, Sprint, Morgan Stanley, Nokia, and Target, are also attracted by the SaaS and choose to obtain their software on demand, although they can easily set up the internal system without subjecting to any budget constraint. It is clear that the SaaS providers are stealing market share from the traditional providers of COTS software, and putting significant competitive pressure on them (Economist, 2007).

However, the long-term success of the SaaS in such a competitive setting remains uncertain. Data security and reliability as well as application control are always among users' top concerns (Bednarz 2006) which prevent them from opting for this new business model. In addition, the multi-tenancy design by the SaaS, under which providers are hosting a single instance of the software on a single server and maintaining the customer data on a single database (Hickins 2007), brings users the concern of lack of customization. For example, SourceRad, which provides clinical practices with "*integrated office scheduling, web-based viewing, online archiving, disaster recovery, and transcription, all in an affordable, hassle free hosted platform*," (Author visit with SourceRad team, July 2007), operates a "one-to-many" service model, with no customization. As a result, users must exert additional effort to make its standard software application fit smoothly with their existing IT systems.

Although some researchers have already investigated the SaaS, such as Susarla et al. (2003) and Cheng and Koehler (2002), they focus on the monopoly setting and exclude the existence of COTS software providers as well as their market influence. In this study, I look at a marketplace in which the SaaS and COTS software solutions both are available. Our analysis focuses on the competition between the two. The model characterizes three salient differences between the SaaS and COTS. First, they deliver different products: a customized software application (from the COTS) versus a bundle of standard software and services (from the SaaS). Second, they adopt distinct pricing modes: an outright purchase (the COTS) versus a "per transaction" fee structure (the SaaS). Third, they employ different delivery methods: software installed on a user's in-house server (the COTS) versus an interface delivered over the Internet remotely (the SaaS).

We identify several interesting features of such a competition. First of all, we show that pricing its products strategically would allow the SaaS coexist with the COTS in the long run. The market will be segmented in such a way that firms with low transaction volume opt for the SaaS model because of the cheapness and scalability, and firms with high transaction volume prefer the COTS model to enjoy software that fits their specific business needs well. Moreover, we find that if users are concerned about potential changes in their future business environment, the SaaS providers should *increase* their prices. By doing this, they give up the competition with the COTS provider for high-volume users and instead focus on attracting small and medium firms. In contrast, if users expect the unfit costs of using standard software to decrease due to the advance of web technologies, the SaaS providers should *reduce* their prices to compete aggressively with COTS providers for those large corporate users. These counter-intuitive yet important findings help to suggest useful competitive pricing strategies to providers of on-demand software.

The rest of the paper is organized as follows. In Section 2, we describe the model. The analysis of the competition is given by Section 3. Section 4 summarizes our major findings, discusses their practical implications, and concludes the paper.

2 The Model

There are three parties in the market: software users, the COTS provider, and the SaaS provider.

Software users have different IT needs, which are measured by the expected volume of software use. Users who use the software application more frequently (in expectation) are considered with larger IT needs. To capture this heterogeneity, we assume users are uniformly distributed on a unit-length line normalized from 0 to 1. The location of a user on this line represents its expected transaction volume (in terms of the number of transactions). In addition, we assume that each user's actual transaction volume is stochastic. The user only knows the demand distribution, but not the exact number of transactions needed. In light of this, the actual demand for software use for each user i is modeled as a random variable uniformly distributed on $[d_i - \theta, d_i + \theta]$, where the parameter d_i represents his expected number of transactions and θ measures the volatility of the actual transaction volume. Note that d_i itself is a random number distributed from 0 to 1.

The COTS provider sells the packaged software application to users and charges a one-time upfront fee. The source code of the application will be modified to fit the user's specific business needs, and thus the COTS in-house system is well customized. The provider bears an operating cost C to serve one user and receives a one-time payment P from the user. The user needs to install hardware and infrastructures, hire IT staff, and organize an internal IT group to provide software maintenance, data backups, and security and capacity management. Such service costs associated with each use of the software are denoted by c (i.e., the service costs per transaction). Each transaction creates a value of u to the user.

The SaaS provider sells the bundle of software and services on demand. The cost structure faced by the SaaS provider has two components: a setup cost S per user (the fixed part) and a service cost c per transaction (the variable part). Users pay as they go, incurring a payment p_a per transaction to the provider. The software is installed on a central location which is controlled by the provider. All users can access and run it remotely via the Internet. To any individual user, the application is not well-customized, and each transaction gives the user a total value of *u-t.* The parameter *t* measures the user's disutility from not using its ideal product and is called users' unfit costs in this paper. In many cases, it also represents the cost of extra effort to make the outside application work with the user's existing IT components smoothly.

Competition goes as follows. The two software providers are competing on prices. They set their respective prices simultaneously to maximize profit by considering the other's responses. Given the prices, users choose one provider or just stay out of the market by comparing the costs and benefits of using each provider.

3 Competition Analysis

In what follows, we analyze three different competition scenarios and then compare the pricing strategies under each. Section 3.1 first studies the providers' prices in a static competition. It will be used as a benchmark case. Then we discuss the essence of competition in a longer time window with possible dynamic changes in unfit costs, which could increase or decrease for some practical reasons. Section 3.2 and 3.3 study each of the two changes respectively and compare the findings to the benchmark case.

3.1 Pricing Strategy in the Static Competition (The Benchmark Case)

Consider user *i* with expected transaction volume d_i. Denote its actually transaction volume by D_i. If the user opts for COTS which charges a one-time payment P, it needs first to decide and install proper IT service capacity level q_i internally, which is obtained by maximizing the user's expected utility:

$$\underset{q_i}{Max} \quad E[u\min\{D_i, q_i\} - P - cq_i] = uq_i[1 - F(q_i)] + uF(q_i)E[D_i / D_i < q_i] - P - cq_i$$

where $F(.)$ is the cumulative density function of the transaction volume for user *i*.

With probability $1 - F(q_i)$, the actual transaction volume will be larger than the user's pre-installed service capacity. The user loses excess demand. With probability $F(q_i)$, the actual transaction volume will be smaller than the user's service capacity. The user incurs the costs of carrying excess capacity. Solving the optimization problem gives a closed form solution $q_i^* = d_i + \theta\left(1 - \frac{2c}{u}\right)$. Hence, the expected utility for a COTS user is $d_i(u-c) - P - \frac{c(u-c)}{u}\theta$.[2] On the other hand, if the user opts for the SaaS, it gains an expected utility of $(u - p_a - t)d_i$.

[2] To understand this expression: the first term is the expected value from using the software; the second term is the user's one-time payment to the provider; and the last term represents the user's loss due to transaction uncertainty. Detailed derivations are available upon requests.

It is easy to see that in equilibrium the market is segmented in such a way that users with low transaction volume choose the SaaS and users with high transaction volume choose the COTS software, and the indifferent user has the expected transaction volume of $d^* = \frac{P}{p_a - c + t} + \frac{c(u-c)\theta}{u(p_a + t - c)}$. Hence, the COTS provider serves users in $[d^*, 1]$, with a market share of $1 - d^*$, and the SaaS serves users in $[0, d^*]$, with a market share of d^*.

The two providers choose prices P and p_a to maximize respective profit as follows: $\underset{P}{Max} \ (P - C)(1 - d^*)$ and $\underset{p_a}{Max} \ \frac{1}{2}(p_a - c)(d^*)^2 - Sd^*$.

The equilibrium price pair is characterized in Proposition 1.

Proposition 1. *In the static competition, the price equilibrium exists. There is a threshold value t^* for the unfit cost parameter.*

a) When $t^* \leq \frac{u-c}{2}$, the equilibrium prices are given by equations (1) and (2).

$$P = \frac{p_a + t - c}{2} + \frac{C}{2} - \frac{c(u-c)\theta}{2u} \tag{1}$$

$$p_a = \frac{B(t+c) + 2Su(t-c)}{B - 2Su}, \text{ where } B = Pu + c(u-c)\theta. \tag{2}$$

b) When $t \geq t^*$, the equilibrium prices are $(P^*, p_a^*) = \left(\frac{C}{2} + \frac{(u-c)}{2} - \frac{(u-c)c}{2u}\theta, u - t \right)$.

Proposition 1 describes the software providers' pricing strategy in a static competition. It is noticeable that the unfit cost parameter plays an important role. Whether or not it exceeds the given threshold value t^* defines distinct pricing strategy. When such threshold value t^* is not reached yet, both providers' prices are increasing in the unfit cost but the SaaS' price is capped at $u - t$. When the unfit cost exceeds t^*, the SaaS charges $u - t$, the upper limit of the price which could attract users and leaves zero consumer surplus.[3]

In practice, however, unfit costs could change. Unfit costs could grow over time, given software or hardware changes on the users' side, or decrease over time due to technology improvements. For example, if the SaaS uses a browser interface that is dependent on nonstandard aspects of IE7 but business circumstances faced by users drive a demand for the latest IE or Firefox, or if the SaaS's interface involves a module built on top of a program that only works in a pre-Vista MS Windows environment but hardware replacement at the user's site leads to multiple PCs with the Vista OS, unfit costs may increase. On the other hand, if the SaaS provider continuously

[3] At this price upper limit *u-t*, the SaaS provider extracts all consumer surpluses. Any price higher this upper limit will drive users to be out of the market.

invests in improving its system integration features, users' unfit costs may be decreasing over time. For instance, Salesforce.com developed and launched AppExchange in January 2006. AppExchange is an online marketplace for on-demand business software. Currently it includes over 150 applications, and Adobe, Skype, and Factiva are among the various partners. AppExchange allows Salesforce.com and other on demand software providers to integrate their applications and therefore promises software users seamless extension of their existing systems (Cowley 2005; Kuchinskas 2006). In this case, users expect to have reduced unfit costs because a uniform platform eases collaboration across applications. Considering such changing natures of the market, we need a two-stage model to capture the essence of competition in a longer time window with possible dynamic changes in unfit costs. In the first stage, the vendors choose their prices (p_a, P) simultaneously, which are assumed unchangeable in the time line we are studying. The SaaS imposes unfit costs t_1 in the first stage. Users could have certain expectations about a future change in unfit costs: users may expect unfit costs to increase if they anticipate changes in demand or hardware upgrades, or to decrease if they anticipate technological advances that favor the shared software business model. In the second stage, such a change is realized. Users will then consider switching. We make two simplifying assumptions for model tractability. First, users and software providers weight utilities and profits obtained from both stages equally. Second, the initial setup costs of the SaaS to serve a new client are negligible. These two assumptions help to ease the analytical exposition without changing the results qualitatively.

3.2 The Two-Stage Model with Increased Unfit Costs

Consider the scenario that the initial unfit cost is t_1 while users expect such cost to increase to t_H later. Figure 1 depicts this two-stage competition. In the first stage, with an unfit cost t_1, users in $[0, d_1]$ choose the SaaS, and users in $[d_1, 1]$ opt for the COTS software. The user d_1 should be indifferent between the two choices. In the second stage, with an increased unfit cost $t_H > t_1$, SaaS users in $[d_S, d_1]$ switch to the COTS system for a better fit while the rest stay with their initial choices. The "marginal" switcher is given by $d_S = \frac{P}{p_a + t_H - c} + \frac{c(u-c)\theta}{u(p_a + t_H - c)}$.[4] On the other hand, the user d_1, since it is indifferent between the two providers in the first stage, gains the same total utility from both. If it chooses the SaaS and switches to the COTS later, its total utility is $\{(u - p_a - t_1)d_1\} + \left\{(u-c)d_1 - P - \frac{c(u-c)\theta}{u}\right\}$; if it chooses

[4] An existing SaaS user i compares the utility from the SaaS, $(u - p_a - t_H)d_i$, with the utility from the COTS, $(u-c)d_i - P - \frac{c(u-c)\theta}{u}$, to decide whether to switch. Therefore, the marginal switcher is the one who gets same utility from both providers.

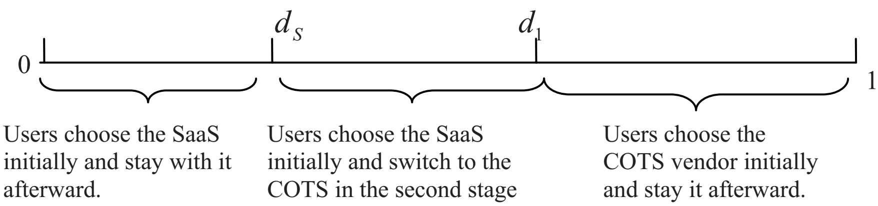

Fig. 1. Competition with Increased Unfit Costs

the COTS initially, it obtains $\left\{(u-c)d_1 - P - \frac{c(u-c)\theta}{u}\right\} + \left\{(u-c)d_1 - \frac{c(u-c)\theta}{u}\right\}$. By equating these two utilities, we get:

$$d_1 = \frac{c(u-c)\theta}{u(p_a + t_1 - c)}. \qquad (3)$$

The COTS provider gets $[d_1, 1]$ users in the first stage and $[d_S, d_1]$ users in the second stage. Its profit comes from users' one-time payment. The SaaS provider serves $[0, d_1]$ users in the first stage and $[0, d_S]$ users in the second stage. It gains profit from users' every use of the software. Their prices are determined as follows.

$$\underset{P}{Max}\ (P-C)[1-d_S]. \qquad (4)$$

$$\underset{p_a}{Max}\ (p_a - c)\left[\int_0^{d_1} x dx + \int_0^{d_S} x dx\right] = \frac{1}{2}(p_a - c)(d_1^2 + d_S^2). \qquad (5)$$

Let $p_a\big|_{t_1,t_H}$ and $P\big|_{t_1,t_H}$ be the equilibrium prices in such a two-stage competition and let $d_1\big|_{t_1,t_H}$ be the indifferent user defined by equation (3). Let $p_a\big|_{t_1}$ and $P\big|_{t_1}$ be the equilibrium prices in the static competition with unfit costs t_1, i.e., the prices in the benchmark case, and $d_{t_1}^*$ be the indifferent user in that case.

Proposition 2. *When users anticipate a future increase in unfit costs, $t_1 \rightarrow t_H$, both vendors will increase their prices; i.e., $p_a\big|_{t_1,t_H} > p_a\big|_{t_1}$, and $P\big|_{t_1,t_H} > P\big|_{t_1}$. More users will choose the COTS software initially, i.e., $d_1\big|_{t_1,t_H} < d_{t_1}^*$,, and the SaaS will lose existing clients to the COTS provider once the cost increase occurs.*

Proposition 2 states three important findings. First, although increased unfit costs imply a decrease in the quality of the SaaS product ($u-t$), the on demand software provider should nevertheless increase its price: $p_a\big|_{t_1,t_H} > p_a\big|_{t_1}$. By charging a high price, the provider gives up competing for high-volume users with the COTS provider; it instead concentrates on exploiting low-volume users that are unable to afford the COTS anyway. Second, the COTS provider also raises its price, which is intuitive because its product becomes more attractive. Interestingly, we find that the COTS provider's pricing function ($P = \frac{p_a + t_H - c}{2} + \frac{C}{2} - \frac{c(u-c)\theta}{2u}$, equation (1)) is the same as that in a static competition with $t = t_H$. This means that the COTS provider should adopt a simple pricing strategy. Software is priced as if it were in a one-stage competition with invariant unfit costs. Finally, we conclude that a belief that unfit costs will increase benefits in-house solution providers but hurts the SaaS providers.

3.3 The Two-Stage Model with Decreased Unfit Costs

Figure 2 shows the two-stage competition when *t* decreases. In the first stage, with unfit costs t_1, users in $[0, d_1]$ choose the SaaS, and users in $[d_1, 1]$ buy the COTS. The indifferent user is at d_1. In the second stage, the unfit cost decreases to $t_L < t_1$, which could be the result of web technology improvements, adoption of software standards and protocols, or creation of a uniform software platform. In such cases, existing COTS users compare their utility from switching to the SaaS, $(u - p_a - t_L)d_i$, and staying with the COTS, $(u-c)d_i - \frac{c(u-c)\theta}{u}$. The "marginal" switcher, d_S, is the one who gains the same utility from these two options: $d_S = \frac{c(u-c)\theta}{u(p_a + t_L - c)}$.

The indifferent user d_1 can be found as follows. If this user chooses the COTS and then switches to the SaaS later, its expected total utility is $\left\{(u-c)d_1 - P - \frac{c(u-c)\theta}{u}\right\} + \{(u - p_a - t_L)d_1\}$; if it chooses the SaaS in the first stage, its total utility is $\{(u - p_a - t_1)d_1\} + \{(u - p_a - t_L)d_1\}$. By equating both, we get

$$d_1 = \frac{P}{p_a + t_1 - c} + \frac{c(u-c)\theta}{u(p_a + t_1 - c)}. \tag{6}$$

Note that the number of switchers (from the COTS to SaaS) is *not* affected by the COTS price because it is considered a sunk cost at the second stage.

The COTS and SaaS providers choose profit-maximizing prices respectively, as described by equations (4) and (5).

Let $p_a\big|_{t_1,t_L}$ and $P\big|_{t_1,t_L}$ be the prices of the SaaS and COTS products, and let $d_1\big|_{t_1,t_L}$ be the indifferent user defined by equation (6).

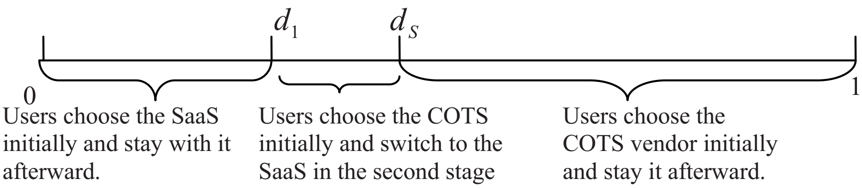

Fig. 2. Competition with decreased unfit costs

Proposition 3. *When users anticipate a decrease in unfit costs,* $t_1 \rightarrow t_L$*, both providers will reduce their prices; i.e.,* $p_a\big|_{t_1,t_L} < p_a\big|_{t_1}$*, and* $P\big|_{t_1,t_L} < P\big|_{t_1}$*. More users will choose the SaaS initially; i.e.,* $d_1\big|_{t_1,t_L} > d^*_{t_1}$*. Existing clients of the COTS provider have little incentive to switch to the on-demand software even if unfit costs decrease, but they may do so if transaction volatility is high.*

Three important findings are stated in Proposition 3. First, the SaaS provider's response to the expected decrease in unfit costs, which represents an increase in the quality of its product, is to reduce its price. The increased quality and reduced price together put the SaaS in a much better position in the competition with the COTS provider to gain high-volume users, who are much more profitable in the eyes of the SaaS.[5] Second, the COTS provider once again will stick to a simple pricing strategy. Its pricing function, $P = \frac{p_a + t_1 - c}{2} + \frac{C}{2} - \frac{c(u-c)\theta}{2u}$ (Equation (1)), is the same as that in a static competition with unfit costs $t = t_1$. It therefore can just ignore the expected future changes and price its software as if in a one-stage competition. Third, existing users of COTS software are unlikely to switch to the on-demand software. These users have two choices: stay with the COTS solution, with a utility of $(u-c)d_i - \frac{c(u-c)\theta}{u}$, or switch, with a utility of $(u - p_a - t_L)d_i$. Since $p_a > c$ always, the user switches only if its transaction volatility (θ) is high. Hence, we conclude that once an in-house system has been installed, users have little incentive to switch to the SaaS unless they need to manage risks caused by volatile demand.

4 Discussion and Conclusion

In this paper we try to shed light on the pricing strategy of the SaaS providers in the competition with the traditional COTS software providers. We examine the equilibrium prices in both static and dynamic market conditions where users face stochastic demand. Our findings show that the SaaS on-demand model is superior when a user faces low transaction volume and/or high transaction volatility. It offers small firms

[5] The SaaS gets paid per transaction. So high-transaction-volume users are more profitable than low-transaction-volume users.

cost-saving access to software and becomes the natural choices of them, and meanwhile it competes with the COTS solution on firms with large transaction volume.

One common belief in designing the SaaS software is that increasing the product quality by reducing its unfit cost should support higher prices. Hence, some results from our two-period analysis may seem counter-intuitive at first glance. We establish that the SaaS provider's optimal response to users' anticipation of decreased unfit costs (which means an increase in product quality) is to *reduce* its price. The SaaS is more competitive in this situation and thus can go after the more profitable high-volume users. Since the SaaS provider is paid on a per transaction basis, users with high transaction volumes are considered more profitable. Although the provider gains smaller revenue per transaction (due to the reduction in its unit price), its market share expands to encompass part of the segment with larger transaction volume. On the other hand, when users anticipate a future increase in unfit costs (which means a decrease in the quality of the SaaS product), the on-demand software provider should *increase* its price. By charging a higher price, it gives up competing against the COTS provider for high-volume users. Instead, it separates the market and concentrates on exploiting low-volume users that are unable to afford the COTS anyway.

Although this work only focuses on analyzing the SaaS providers' pricing strategy in the competitive market, there are many possibilities for further SaaS studies. It would be interesting to examine the role of differential service level agreements (SLAs). When users have demand for different levels of service, the SLA constitutes a way to segment the market and improve the SaaS's profit. Another possible extension encompasses the design and management of a dual channel. Certain vendors (such as Oracle and IBM) have changed their business models to offer both COTS and SaaS products. Typically, they are selling a sophisticated version of their software products as a COTS product and are leasing simplified operating versions as the SaaS. Future research may investigate the proper pricing and functionality bundle per channel.

References

1. Bednarz, A.: Manufacturers Eye on On-Demand Software. Network World, April 24 (2006)
2. Cheng, H.K., Koehler, G.J.: Optimal Pricing Policies of Web-enabled Application Services. working paper (2002)
3. Cowley, S.: Salesforce.com Makes Platform Move with AppExchange. InfoWorld (September 2005)
4. Hickins, M.: Oracle: On-demand is Now On The Grid. Enterprise (April 2007)
5. Kuchinskas, S.: Salesforce Finally Ships AppExchange. Ecommerce, January 17 (2006)
6. Lacy, S.: The On-Demand Software Scrum. Business Week, April 17 (2006)
7. Proponents of Software as a service say it will wipe out traditional software. The Economist, April 21 (2006)
8. Pallatto, J.: IBM Recruiting IS vs. Partners to SaaS. Channel Insider, Feburary 23 (2006)
9. Software as a Service (Research Report). InformationWeek (March 2007)
10. Susarla, A., Barua, A., Whinston, A.: Understanding the Service Component of Application Service Provision: An Empirical Analysis of Satisfaction with ASP Services. MIS Quarterly 27, 123–919 (2003)