A SIMULATION STUDY OF THE INTERDEPENDENCE OF SCALABILITY AND CANNIBALIZATION IN THE SOFTWARE INDUSTRY

Francesco Novelli

SAP Research Darmstadt, Germany

francesco.novelli@sap.com

ABSTRACT

Dominant software vendors, whose applications have been predominantly delivered on-premises so far (i.e., installed, maintained and operated at customers' premises) are challenged by the rising adoption of software as a service (SaaS) solutions, outsourced applications delivered through the web under subscription or usage-based pricing terms. As a response, these incumbent vendors extend their product portfolios with SaaS offerings, but thus risk to engender revenue cannibalization, as a newly introduced SaaS application may attract their own on-premises customers instead of expanding the market or drawing from a competitor's customer base. At the same time they face the novel, severe scalability requirements of the technological and organizational infrastructure underlying a successful SaaS business. Using an agent-based simulation model, we study the interdependence between cannibalization and scalability in monopolistic and duopolistic software markets.

Keywords: software as a service, cannibalization, scalability, agent-based modelling and simulation

1. INTRODUCTION

1.1. Evolution of Software Delivery and Pricing Models

In the last decade the software industry has witnessed the emergence of the so-called software as a service (SaaS) delivery model, whereby vendors provide web-based, outsourced software applications (SIIA 2001), dispensing customers with most installation, operation and maintenance activities otherwise needed at their premises. Moreover, SaaS solutions are usually coupled with subscription or usage-based pricing models (Lehmann and Buxmann 2009), lowering the initial investment in comparison with packaged software.

As a matter of fact, several beholders of the software industry agree that the adoption of SaaS applications has gained momentum: Information Systems researchers (Benlian, Hess, and Buxmann 2009), IT market analysts (Gartner 2010), and investors, who, in the first quarter of 2011, gave SaaS public companies an average market evaluation 6.5 times their annual revenues. This trend, in turn, urges incumbent vendors, which built their dominant market positions on the on-premises delivery model, to launch SaaS counterparts to their established software products, in coexistence or as replacements.

We focus on two key challenges inherent in such a strategic response. On the one hand, it is indispensable for the incumbent to understand the dynamics of revenue cannibalization and its consequences on profitability and on the positioning of the firm vis-a-vis the competition. On the other hand, it is necessary to achieve the degree of technological and organizational scalability required to profitably implement a SaaS strategy.

1.2. Cannibalization

Cannibalization is the switching of sales from an existing product to a new one of the same firm (Newman 1967). It is an issue of paramount importance for an incumbent vendor venturing into SaaS, given the intrinsic degree of substitutability between the already established on-premises products and their SaaS siblings. This may indeed put pre-existent revenue streams and market shares at stake.

As a case in point, let us consider how competition is unfolding in the office automation market. Microsoft is the dominant player with 6 billion dollars revenue from that segment in the second quarter of 2011 (as a term of comparison, 5 billion was the revenue from the Windows OS). However, the entry into the market of free online office applications (such as those by Google) has pushed Microsoft to respond with the development of two SaaS alternatives to its well-known Office suite: a free, ad-supported one with limited functionalities and a subscription-based one with enhanced collaboration features. The delicate challenge is for Microsoft to tame the cannibalization effect this move may engender, i.e., to avert a financially harmful drift of on-premises customers to its own SaaS offerings.

Cannibalization may also represent a deliberate, offensive product strategy, pursued to drive growth (McGrath 2001). As a matter of fact, some on-premises vendors have successfully managed the transition to a hybrid or purely SaaS model. Concur Technologies, for instance, paired its on-premises offerings with the Application Service Provider model (predecessor of SaaS) in the late 90s already, and then transitioned to become a purely SaaS player just as this delivery model emerged (Warfield 2007). Analogously, Ariba started the transition in 2006 and gradually ported all its applications to a SaaS model, now the generator of most of its revenues (Wainewright 2009). Both companies initially went after a SaaS-enabled market expansion aimed towards the mid-segment but eventually – in sharp contrast with the Microsoft scenario – found it profitable to deliberately cannibalize their on-premises customers along the way.

1.3. Scalability

Understanding the financial and competitive consequences of revenue cannibalization and then attempting to avert it or to ride it are not the only concerns facing incumbents. The SaaS delivery model poses a scalability threat as well, both from a technological and from an organizational perspective.

This threat stems from the peculiarities of this newly addressable market. Since SaaS lowers the technological and financial requirements for a software purchase, the market swells in number of potential buyers while the average financial and technological resources available to them decrease. As a matter of fact, since a SaaS offering is hosted and operated by the provider and accessed through the World Wide Web, very simple applications that do not demand supplemental integration and customization virtually appeal to any organization meeting the minimum technical requirement of an available Internet access. From a financial point of view, the SaaS subscription fees dilute over time the investment for the license of a given software functionality. Therefore, small and medium-size companies can, in spite of their usually more limited IT budget and technical personnel, enter application markets once populated by large enterprises only. This is exemplarily shown for the European Union in Table 1. It evidently represents a huge market opportunity to be tapped into by software vendors in terms of number of potential new accounts (enterprises) and users (employees).

Table 1: Key indicators per enterprise size class in the EU-27 area; source: Eurostat (# as of 2010, * as of 2008, † as of 2007)

Size class	Small	Medium	Large
Employees	10-49	50-249	250 or more
Number of enterprises*	1,408,000	228,000	45,000
Persons employed* (million) *	27.9	23.4	45.2
% enterprises that employ IT/ICT specialists†	9%	25%	53%
% enterprises with an ERP system#	17%	41%	64%
% enterprises with a CRM system#	23%	39%	54%

This opportunity for market expansion has its downsides: though larger, the new potential market is more costly to be reached and to be served. Figure 1 compares the trend in total operating expenditures over total revenues for the two leading business application vendors (SAP and Oracle), which have historically kept it in the 60-70% range, and two SaaS competitors, unable, even after having successfully ridden a steep learning curve, to lower it under the 90% mark (Salesforce) or to reach operational profitability at all (NetSuite).

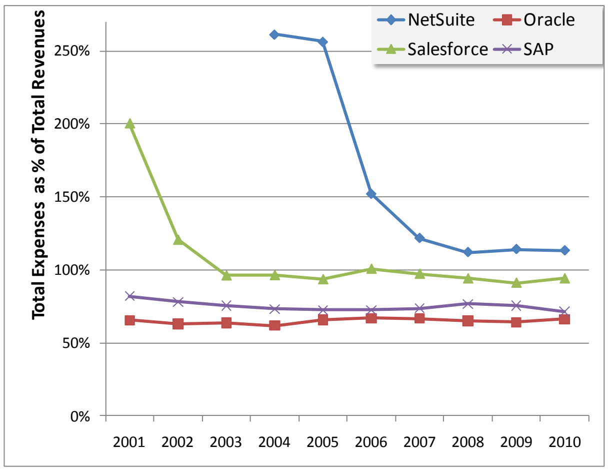

Figure 1: Total Operating Expenses as a Percentage of Total Revenues for selected Software Vendors (source: corporate financial reports)

As a matter of fact, a SaaS software vendor must bear the additional costs of setting up and operating the technological infrastructure needed to deliver the SaaS application. Moreover, beyond those technological repercussions, scalability issues engage the SaaS provider on an organizational level as well, for this new, more fragmented segment of software buyers imposes to think a series of processes anew. For instance marketing and sales, where using dedicated sales team for each account as it is the habit with large enterprises is not possible on a large scale, and other means, such as telesales and innovative internet-based funnels, need to be employed.

1.4. The Interplay of Cannibalization and Scalability

Incumbent software vendors introducing SaaS are confronted with a typical new product introduction problem, where the new product may divert current customers from other offerings of the same firm, instead of attracting new buyers or drawing from a competitor’s customer base (Kerin, Harvey, and Rothe 1978), further complicated by the trade-off between a more saturated but highly profitable current software market of large enterprises and a fast-growing but less profitable potential SaaS market.

Scalability of the SaaS business is certainly a prerequisite to target market expansion. Without it vendors face the risk of not being able to satisfy demand – i.e., failing to build the appropriate level of capacity to ride growth – or doing so inefficiently – i.e., failing to reach the scale economies that make market expansion a profitable endeavour at all.

Changing perspective, limiting scalability can be a radical lever to avert cannibalization, for it puts an upper bound on the volume of intrafirm switching customers. This may expose the incumbent's flank to competition though, whereby customers might switch to a competitor instead.

This work is organized as follows: the reasons that led to the choice of Agent-Based Modelling and Simulation as our research methodology are summarized in the following section (2). In section 3 we describe the model we based our simulation experiments on, which are then detailed and discussed in section 4. Eventually, we conclude in section 5, mentioning the limitations of the present work and possible future developments.

2. RESEARCH METHODOLOGY

2.1. Agent-Based Modelling and Simulation to Study Microeconomics

A market represents an excellent example of Complex Adaptive System (CAS), a collection of adaptive agents (suppliers and customers) concurrently engaged in local interactions (commercial transactions). Local interactions produce higher level conditions (market prices, bandwagon effects, etc.) impacting in turn the way those same interactions will evolve.

Among the paradigms used to investigate such systems we chose Agent-Based Modelling and Simulation (ABMS). In ABMS each interacting component is modelled as an autonomous decision-making agent with attributes and rules defining his behavioural characteristics and how those are to evolve or adapt (North and Macal 2007). This approach lends itself neatly to the exploitation of microeconomic constructs in modelling agents' behaviours and interactions (game theory, for instance, to dictate an agent's strategic response) and is therefore especially suitable to study a CAS populated by economic entities. In fact, the study of economics with ABMS has reached such a respectable status to beget its own specific field of research, called Agent-Based Computational Economics (Tesfatsion 2002).

In this work we use ABMS to investigate at a microeconomic level a stylized business application software market where a multi-product incumbent vendor runs the risk of revenue cannibalization. ABMS suits perfectly the study of this phenomenon since, offering the possibility to observe the behaviours and decisions of individual buyers, it allows a disaggregated analysis of the components of demand. This means identifying exactly which customers switch between software applications of the same vendor (cannibalization), leave for a competitor (competitive draw), or enter the market for the first time (market expansion).

3. MODEL

We model a closed, vertically-differentiated software application market. The market structure is a monopoly with a single vendor selling both an on-premises application and a SaaS one for the first series of experiments, and a duopoly consisting of the same vendor plus a purely SaaS challenger for the second set of experiments. Given space constraints, in this section we describe only the most salient features of the model.

3.1. Software Application

A software application is characterized by the features or benefits it provides (its "quality") to its users, the price to be paid to obtain those benefits (in terms of amount and distribution over time of the fees) and the infrastructure on which it is deployed.

When the application is delivered as SaaS, it will be deployed on an infrastructure operated by the software vendor and priced under subscription terms, with an initial activation charge at the time of purchase and an anticipated, recurrent fee for each period of the simulation in which it is used. When the application is delivered on-premises, the price structure will follow the typical enterprise application pricing model and be once again made up of two components: an initial charge to purchase the licenses and an anticipated, periodical maintenance fee as a percentage of the initial charge.

While the fee structure we employ for the two delivery modes is the same, the proportion between initial and periodical charge differs, with on-premises application weighting more on the front – annual maintenance rates are around 20% of the initial investment for licenses (Buxmann, Diefenbach, and Hess 2011) – and SaaS diluting the expenses more over time.

3.2. Software Application Vendors

The software vendors are price-making suppliers of the same class of business application (e.g., ERP, CRM, etc.) but with different price-quality schedules and delivery models. In the case of a SaaS the vendor must also operate the infrastructure on which the application is deployed. In each simulation period vendors collect the due payments from the customers that adopted one of their applications and bear the costs of the SaaS infrastructure.

3.3. SaaS Infrastructure and Scalability

The infrastructure is made up of a set of technological or organizational resources (e.g., servers, sales representatives) characterized by a certain individual performance. The overall performance is, however, not just the sum of these components, and depends on the level of scalability of the infrastructure. We use Amdahl's law (Amdahl 1967) to account for this issue and formalize the degree of non-scalability of the infrastructure through a so-called contention rate, which exerts a negative impact on the ability to efficiently scale (and, as we will see, to compete) growing exponentially with the scale requirement (Shalom and Perry 2008).

The maximum total capacity K of an infrastructure with N resources of throughput τ each, designed to have a CnR rate of contention is:

$$\kappa = \frac{\tau}{CnR + \frac{1 - CnR}{N}} \tag{1}$$

Equation (1) gives the total capacity a certain infrastructure can attain in a given period. For instance, 20 resources with throughput of 1000 customers per period each, arranged in an architecture designed to have a 20% contention rate, would generate a total capacity of 4167 customers per period. Doubling the resources (i.e., scaling out of 20 additional resources) 4545 customers could be served (an 8% increase). However, the maximum achievable capacity would be bounded to less than 5000 customers per period, no matter how many additional resources are thrown in. Reducing contention would be a much effective lever: decreasing the contention rate of 5% would increase capacity by 25% (to 5195 customers per period).

3.4. Software Application Customers

Customers are current or potential adopters of a software application sold in the market. The decision to adopt an application is made on the basis of the obtained surplus. The surplus for the i-th customer of type θ when adopting an application j is:

$$S_{ij} = \theta_i Q_j - TCO_j(T) + \propto X_j \qquad (2)$$

The first term of equation (2) is the willingness to pay of a customer with marginal valuation of quality θ for an application of quality Q. θ is an input parameter set randomly for each consumer at simulation start (drawn from a uniform distribution with support between 0 and 1). The second term is the present value of the total cost of ownership of the application (detailed below). The third addend is the network externality derived from all consumers that already adopted an application with the same delivery model (i.e., the relevant network X_j is the total number of SaaS customers if j is one of the SaaS applications, or the total number of on-premises customers if j the incumbent's on-premises application).

The total cost of ownership over a horizon of T years is computed for both on-premises and SaaS applications employing the formula for the present value of an annuity:

$$TCO_j(T) = \varphi_{t_0} + \varphi_t + \frac{\varphi_t}{r}\left(1 - (1+r)^{-T+1}\right) \qquad (3)$$

where $\varphi_{t_0}$is initial charge (activation of the SaaS subscription or on-premises license charge), φ_t the anticipated periodical charge (the subscription fee or the maintenance fee respectively), and r the annual interest rate.

When taking a purchase decision, a consumer first calculates (2) for every available application, then adopts the one with highest non-negative surplus among those with available capacity offered in the market. Although we do not explicitly model switching costs, the consumer that has already adopted an application considers the initial charge a sunk cost and drop $\varphi_{t_0}$from equation (3) when comparing the surplus of her current choice with other alternatives in the market.

The offerings in the market have a certain initial market share each in terms of pre-assigned customers (the incumbent's on-premises one being the largest), but the overall addressable market includes potential customers that will take their first buying decision during the simulation.

3.5. Implementation and Runtime Environment

The whole model was implemented in Java using the ABMS open-source toolkit Repast Simphony 2.0 (North *et al.* 2007).

4. EXPERIMENTS

An experiment consisted of 10 replications of 21 periods of length (the equivalent of three 7-years software application life cycles in the temporal scale we chose) for each model configuration, where a model configuration was given by the contention rate of the incumbent's SaaS infrastructure, specified in a 0%-50% interval with 5% steps. Each experiment was conducted in different growth and competitive scenarios as detailed in the two following sub-sections.

4.1. Experiments in a Monopolistic Market

Our first series of experiments dealt with a monopoly in two scenarios: one of high growth of the SaaS segment, where this has a total size (in terms of number of potential customers) 10 times the on-premises segment, and one of low growth, where it merely matches the on-premises segment's size.

In a monopolistic situation the decisions of the incumbent is linked to the trade-off between revenue cannibalization and market expansion. If the potential market tapped into with SaaS is large enough in size to offset the effect of cannibalizing the high-margin on-premises sales, then the vendor should pursue a high-capacity strategy and, therefore, invest in a scalable infrastructure. Otherwise cannibalization could be averted by limiting the capacity offered in the market with a more conservative strategy. Conversely, a company that has not yet reached the needed level of scalability would unprofitably pursue growth in the SaaS segment and should refrain from it.

Examining the results of these first experiments, it can be seen that, in case of high-growth in the SaaS segment, the monopolist may indeed offset (in terms of sales volume) revenue cannibalization with market expansion by pursuing a high-scalability strategy (Figure 2). On the contrary a low-scalability strategy allows minimizing revenue cannibalization in a low-growth scenario, where no significant market expansion would be possible anyway (Figure 3). Please note that throughout the remainder of this section we calculated total cannibalized revenues in terms of the projected on-premises revenues lost when the customer switch to SaaS (i.e., the discounted stream of maintenance fees, as expressed by eq. 3).

The specific contribution margins of the two software products will dictate the overall effect on the monopolist's profit. Given the higher margins enjoyed in delivering on-premises applications (see introduction), mirrored in our model, a multi-product monopolist in a low-growth scenario would be better off slowing the rate of SaaS adoption among its own customers by limiting the offered capacity (Figure 4). On the contrary, being able to scale to expand into the SaaS segment would be, in case of high growth, the more profitable strategy.

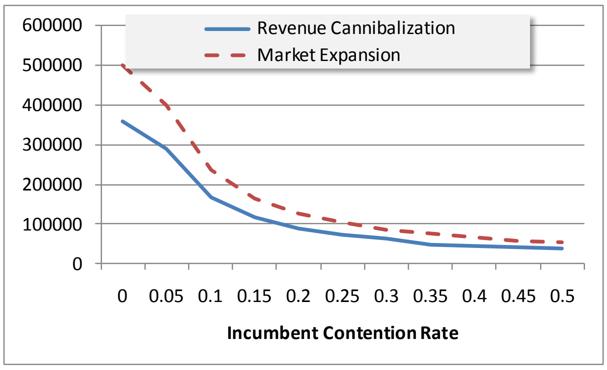

Figure 2: Total Cannibalized On-Premises Revenues and Total SaaS Market Expansion in a Scenario of High Growth (Average for 10 replications)

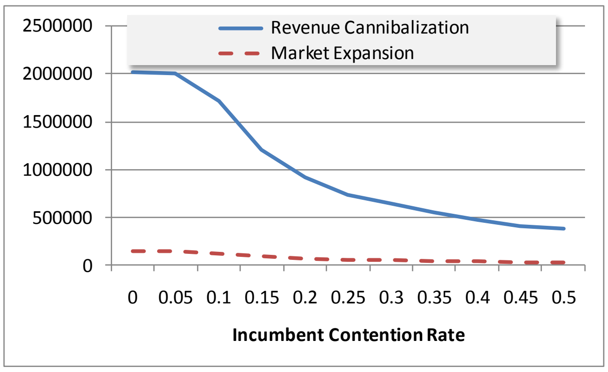

Figure 3: Total Cannibalized On-Premises Revenues and Total SaaS Market Expansion in a Scenario of Low Growth (Average for 10 replications)

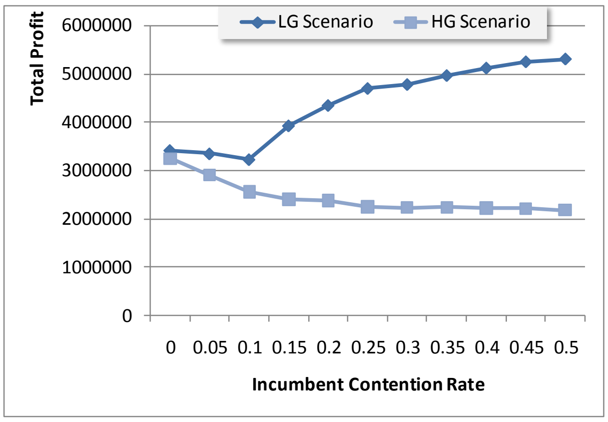

Figure 4: Incumbent's Total Profit in the two Monopolistic Scenarios (HG = High Growth, LG = Low Growth; Average for 10 replications)

4.2. Experiments in a Duopolistic Market

In the presence of a SaaS challenger the incumbent's on-premises customers can switch to either the incumbent's SaaS offering or the competitor's one, adding the risk of competitive draw to the strategic considerations of the incumbent. This risk can be more or less pronounced depending on the scalability of the challenger's SaaS infrastructure. We therefore define four basic scenarios, showed in Table 2.

Table 2: Basic Scenarios for the Duopolistic Market

		Challenger's Scalability	
		Low (CnR = 0.3)	High (CnR = 0.05)
Growth in the SaaS Segment	Low (1X)	Scenario LG1	Scenario LG2
	High (10X)	Scenario HG1	Scenario HG2

In confronting a high-scalable SaaS challenger it always pays for the incumbent to be able to match the competitor's scale, because this allows at least retaining through cannibalization customers that would otherwise be lost (Scenario LG2, Figure 5) if not even offsetting any competitive draw or cannibalization effect by riding growth (Scenario HG2, Figure 6).

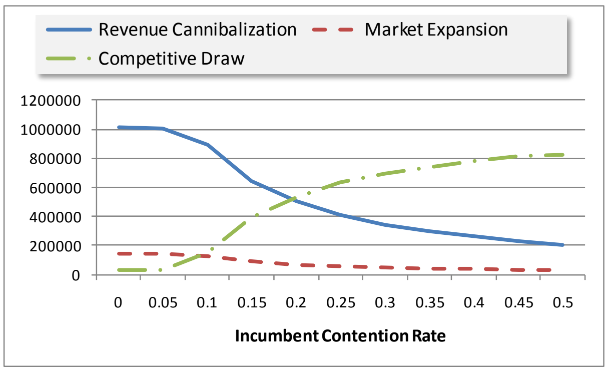

Figure 5: Total Cannibalized On-Premises Revenues, Total SaaS Market Expansion, and Total Competitive Draw of On-Premises Revenues by the SaaS Challenger in Scenario LG2 (Average for 10 replications)

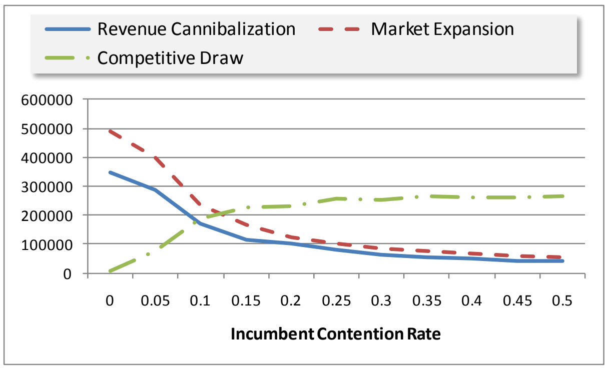

Figure 6: Total Cannibalized On-Premises Revenues, Total SaaS Market Expansion, and Total Competitive Draw of On-Premises Revenues by the SaaS Challenger in Scenario HG2 (Average for 10 replications)

As shown in Figure 7, the incumbent's total profit is generally higher in case of high-growth and negatively correlated with contention, except for the particular case of low growth and presence of a non-scalable challenger (scenario LG1), where the option to limit capacity as a lever to control cannibalization could still be viable. This is due to the lower risk of losing relevant market shares to a poorly-scalable competitor.

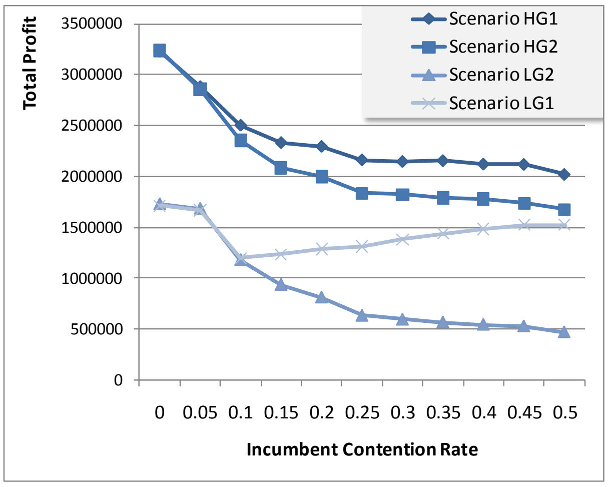

Figure 7: Incumbent's Total Profit in the Identified Market Scenarios (see Table 2; Average for 10 replications)

5. CONCLUSION

This work showed the multi-faceted interdependence of cannibalization and scalability in determining the success of a SaaS strategy pursued by an on-premises vendor, either in a monopolistic position or as an incumbent challenged by a SaaS competitor.

Given the lower margins of a SaaS offering, the monopolist prefers to avoid cannibalization by limiting scale, unless the achievable market expansion proves substantial. In the presence of a SaaS challenger, instead, revenue cannibalization may be for the incumbent a necessary evil whereby customers are retained against the threat of competitive draw. Scalability represents then a key requirement for the incumbent to ride SaaS adoption, cannibalize and possibly expand the market.

These findings were obtained by going after specific strategic interdependences in simplified market scenarios. The modelled market landscape and competitive dynamics should be extended to get a more realistic and comprehensive picture of the trends currently affecting the software industry. Moreover, a thorough validation of the experimental outcomes based on empirical market data ought to be performed.

ACKNOWLEDGMENTS

This work was partially supported by the German Federal Ministry of Education and Research (BMBF) under promotional references 01IA08003A (Project PREMIUM|Services)

REFERENCES

Amdahl, G., M., 1967. Validity of the single processor approach to achieving large scale computing capabilities. *Proceedings of the April 18-20, 1967, spring joint computer conference* (pp. 483-485).

Benlian, A., Hess, T., and Buxmann, P., 2009. Drivers of SaaS-adoption–an empirical study of different application types. *Business & Information Systems Engineering*, *1*(5), 357–369.

Buxmann, P., Diefenbach, H., and Hess, T., 2011. *Die Softwareindustrie: Ökonomische Prinzipien, Strategien, Perspektiven*. Springer, Berlin.

Gartner, 2010. Gartner Survey Indicates More Than 95 Percent of Organizations Expect to Maintain or Grow Their Use of SaaS Through 2010. Available at: http://www.gartner.com/it/page.jsp?id=1361216 [Accessed June 9, 2011].

Kerin, R., A., Harvey, M., G. and Rothe, J., T., 1978. Cannibalism and new product development. *Business Horizons*, 5(21), 25–31.

Lehmann, S., and Buxmann, P., 2009. Pricing strategies of software vendors. *Business & Information Systems Engineering*, *1*(6), 452–462.

McGrath, M., E., 2001. *Product strategy for high technology companies: accelerating your business to web speed*. McGraw-Hill.

Newman, J., W., 1967. *Marketing management and information: a new case approach*. R. D. Irwin.

North, M., J., Howe, T., R., Collier, N., T., and Vos, J., R., 2007. A Declarative Model Assembly Infrastructure for Verification and Validation. *Advancing Social Simulation: The First World Congress* (pp. 129-140).

North, M., J., and Macal, C., M., 2007. *Managing business complexity: discovering strategic solutions with agent-based modeling and simulation*. Oxford University Press, USA.

Shalom, N., and Perry, G., 2008. Economies of Non-Scale. *Nati Shalom's Blog*. Available at: http://natishalom.typepad.com/nati_shaloms_blog/2008/06/economies-of-no.html [Accessed June 26, 2011].

SIIA, 2001. *Software as a service: Strategic backgrounder* (Vol. 21). Retrieved from http://www.siia.net/estore/ssb-01.pdf.

Tesfatsion, L., 2002. Agent-based computational economics: growing economies from the bottom up. *Artificial life*, *8*(1), 55-82.

Wainewright, P., 2009. Ariba's Journey to Software as a Service - The Connected Web. *ebiz*. Available at: http://www.ebizq.net/blogs/connectedweb [Accessed June 17, 2011].

Warfield, B., 2007. Interview: Concur's CEO Steve Singh Speaks Out On SaaS/On-Demand. *SmoothSpan Blog*. Available at: http://smoothspan.wordpress.com [Accessed June 17, 2011].