Technology Management, Economics, and Policy Discussion Paper Series

SEOUL NATIONAL UNIVERSITY
College of Engineering
Technology Management, Economics, and Policy Program

Impact of Pricing Schemes on a Market for Software-as-a-Service and Perpetual Software

Juthasit Rohitratana, Jörn Altmann

TEMEP Discussion Paper No. 2012:88

서 울 대 학 교
공 과 대 학, 기술경영경제정책 대학원과정
151-742 서울시 관악구 관악로 599

Technology Management, Economics, and Policy Program
College of Engineering, Seoul National University
599 Gwanak-Ro, Gwanak-Gu, Seoul 151-742, South-Korea
Phone: ++82-2-880-9140, Fax: ++82-2-880-8389

The *Technology Management, Economics, and Policy Program (TEMEP)* is a graduate program at Seoul National University. It includes both M.Sc. and Ph.D. programs, which are integrated into the graduate programs at the College of Engineering.

The *TEMEP Discussion Paper Series* is intended to serve as an outlet for publishing research about theoretical, methodological, and applied aspects of industrial economics, especially those related to the institute's areas of specialization, namely management of technology, information and communication technology, telecommunication, services, health industries, energy industries, as well as infrastructures for development. In particular, paper submissions are welcome, which analyze current technology and industry related issues, discuss their implications, and suggest possible alternative policies. The objective is to gain insights into important policy issues and to acquire a balanced viewpoint of policy making, technology management, and economics. It will enable us to identify the problems in industries accurately and to come up with optimal and effective guidelines. In addition, another important aim of this series is to facilitate communication with external research institutes, individual researchers, and policy makers worldwide.

Research results disseminated by TEMEP may include views on policies for information and communication technologies, technology management, telecommunication, energy, health, and development economics, but the institute itself takes no institutional policy position. If appropriate, the policy views of the institute are disseminated in separate policy briefs. Thus, any opinion expressed in the TEMEP Discussion Paper Series is those of the author(s) and not necessarily the opinion of the institute.

Finally, the current editor of this series would like to thank Prof. Dr. Almas Heshmati for establishing this working paper series in January 2009 and for guiding its development during the course of the first 55 issues. In particular, Prof. Heshmati provided invaluable contributions for setting up the local IT infrastructure, registering the series with RePEc, disseminating the working papers, and setting the quality requirements.

Jörn Altmann, Editor

Office: 37-305
Technology Management, Economics, and Policy Program
College of Engineering
Seoul National University
599 Gwanak-Ro, Gwanak-Gu
Seoul 151-742
South-Korea

Phone: +82-70-7678-6676
Fax: +1-501-641-5384
E-mail: jorn.altmann@acm.org

Impact of Pricing Schemes on a Market for Software-as-a-Service and Perpetual Software

Juthasit Rohitratana[1], Jörn Altmann[1]

[1] Technology Management, Economics, and Policy Program (TEMEP)
Department of Industrial Engineering
College of Engineering
Seoul National University
Seoul, South-Korea
juthasit@temep.snu.ac.kr, jorn.altmann@acm.org

April 2012

Abstract: In this paper, we present an agent-based simulation system that allows modeling the interactions between software buyers and vendors in a software market. The market offers Software-as-a-Service (SaaS) and perpetual software (PS) licenses under different pricing schemes. Four dynamic pricing schemes are analyzed: derivative-follower pricing, demand-driven pricing, skimming pricing, and penetration pricing. Customer (buyer) agents respond to these prices by selecting the most appropriate software license scheme based on four criteria using the Analytic Hierarchy Process (AHP) decision support mechanism. The four decision criteria relate to finance, software capability, organization, and vendor. The simulation results show that the demand-driven pricing scheme is the most effective method but hard to implement since it requires perfect knowledge about market conditions. As an alternative, penetration pricing and skimming pricing could be used. In addition to this, it can be stated that SaaS is most attractive for small enterprises while PS is attractive for large enterprises.

Keywords: SaaS, Software-as-a-Service pricing, perpetual software pricing, agent-based simulation, Analytic Hierarchy Process (AHP), dynamic pricing, decision support, demand-driven pricing, derivative-follower pricing, penetration pricing, skimming pricing.

JEL Classification Numbers: C15, C44, C51, C53, C61, C63, D40, D45, D81, L24, L86, M15.

1. Introduction

In the real world, pricing schemes are various. Software vendors, which try to maximize profit, might use information about customers, competitors, and market demand to set the price. Detailed information on customers' software selection criteria can help software vendors to design detailed pricing schemes that are attractive to customers. In addition to this, software vendors try to obtain a large market share, using network effects and customer lock-in mechanisms, especially if the potential of product differentiation is narrow [10]. Software vendors, who obtain little information about customers and competitors, can only set their prices based on product development cost and delivery costs [11]. Simple pricing such as cost-plus pricing and derivative-follower pricing are also appropriate pricing schemes for these software vendors [16].

This paper examines a software market, in which Software-as-a-Services (SaaS) vendors and perpetual software (PS) vendors implement different pricing schemes. These pricing schemes are applied on top of the basic pricing concept of the SaaS licensing model and the basic pricing concept of PS licensing models. The basic pricing concept of SaaS is pay-per-use pricing. The price is set for one unit of use. The total use of the software is measured either as the number of accesses to the software, the number of transactions initiated, or the time period of use. The time period pricing usually gives unlimited access to the software for a certain time period (e.g., a month). The basic pricing concept of PS sets a price for a software license, which grants unlimited use of the software and does not expire.

For the analysis of the software market, we develop an agent-based simulation system for dynamic software pricing, focusing on SaaS and perpetual software licensing models. Each vendor agent attempts to maximize its profit by applying one of the following pricing schemes: demand-driven (DD) pricing, derivative-follower (DF) pricing, penetration (PN) pricing, and skimming (SK) pricing. Customer (buyer) agents compare (SaaS and PS) software application offers made by vendors, using the Analytic Hierarchy Process (AHP) as decision support mechanism. The four main criteria used for selecting the software are: finance, software capability, organization, and vendor. The criteria design is based on previous software selection literature [4][5][6][7][8][9][25][26] [27]. The data used for describing organizations and their IT infrastructures is also based on the result of a literature survey [21][28]. Software product/service information is collected from offerings of SaaS and PS software products in the real-world market [34][35].

The simulation results obtained with our agent-based simulation system demonstrate how the software selection criteria impact the decision making and how different pricing schemes influence the success of the software vendors in the market. In detail, it illustrates the revenues from SaaS and perpetual software, using the four pricing schemes. The overall results show that the DD pricing scheme is the most effective method but hard to implement. It requires perfect knowledge about the market conditions. As an alternative for a real-world implementation, the PN pricing scheme and the SK pricing scheme can be considered. Our simulations results also illustrate that the customer base has a significant impact on the success of the pricing plan and the (SaaS and PS) licensing model.

The remainder of this paper is organized as follows. In section 2, previous research on software selection methods and dynamic pricing is discussed. Section 3 describes the proposed simulation model and its agents (i.e., customer agents and vendor agents). The specification of the customer agents includes the AHP-based decision making method and the software selection criteria. The specification of the vendor agent comprises the definition of five pricing schemes. Section 4 presents the simulation results about the four different pricing schemes. Finally, section 5 concludes the paper with a summary.

2. State-of-the-Art

2.1. Software Selection Methods

The following two sub-sections describe decision support tools. After a general overview about decision support tools that have been proposed for hardware, software, and Internet service purchases in literature so far, the Analytic Hierarchy Process (AHP), as the decision support tool of choice, is introduced in more detail.

2.1.1 Decision Support Tools

Many decision support tools for supporting hardware, software, and Internet service purchases have been suggested during the past years [18][20][24][25][26][27]. Altmann and Varaiya proposed a decision support tool for selecting Internet access services that match needs and preferences of customers [18]. Risch and Altmann introduced an extension to WS-Agreement, called computing resource definition language (CRDL), which is a framework for specifying computing resources in service level agreements (SLAs) [20]. The framework contains specifications of hardware, software, pricing schemes, and vendor guarantees, which are important for selecting Cloud computing services. Ullah et al. developed a decision support method for classifying variants of a single software product into product portfolios based on customer preferences [26]. Dias-Neto and Travassos proposed a decision support model for selecting software using a model-based testing (MBT) approach [24]. Kwong et al. introduced a solution, based on a genetic algorithm, for selecting software components [25]. Ayala et al. investigated the software selection from a practical perspective by interviewing practitioners [27].

2.1.2 Analytic Hierarchy Process

The Analytic Hierarchy Process (AHP) method has been developed by Saaty [13]. It is a criteria weighting method, which differs from methods such as Direct Weighting, SMART, SWING, and SMARTER [29]. Direct Weighting requires setting weights for each sub-criterion directly, instead of calculating those weights as AHP requires. The Simple Multi-Attribute Rating Technique (SMART) lets decision makers choose the least important attribute and rate it with 10 points. Then, the decision maker assigns points (> 10 points) to the remaining attributes based on a comparison with the least important attribute that had been chosen before. Using the SWING technique, decision makers give 100 points to the most important criterion first and, then, assign points (< 100 points) to the remaining less

important criteria. Afterwards, for both methods (SMART and SWING), the decision makers rank the attributes in increasing order from the worst to the best attribute.

Jadhav and Sonar evaluated the AHP technique as a flexible and powerful tool for evaluating qualitative and quantitative criteria [5]. They identified as weakness the time-consuming calculation of pair-wise comparisons that increases quadratically as the number of alternatives and criteria increases. Karlsson et al. compared six different methods for prioritizing software requirements (i.e., AHP, Hierarchy AHP, minimal spanning tree, bubble sort, binary tree search, and priority groups) and found that AHP is the most promising method [30].

In addition to this, AHP has been used in many decision making tools for selecting software [4][6][7][8][10]. Godse and Mulik applied the AHP method for evaluating software-as-a-service (SaaS) offerings [6]. In this context, they identified the criteria of SaaS adoption, which are functionality, architecture, usability, vendor reputation, and cost. Lai et al. adopted the AHP method for selecting a multimedia authorizing system [7]. Colombo and Francalanci implemented AHP for selecting customer relationship management (CRM) packages, using criteria that are based on architectural, functional, and cost requirements [4]. Rohitratana and Altmann aggregated both customer and vendor criteria and used these criteria in their AHP-based decision making model for estimating the impact of energy cost on the decision-making process [10]. Mulebeke and Zheng implemented a decision support tool for selecting pen production software. They used Web-HIPRE, an online decision support tool, which allows weighting of criteria according to different weighting methods (including AHP) [8].

2.2 Dynamic Pricing

Dynamic pricing schemes have been investigated with respect to different objectives, ranging from determining price equilibria, achieving stability, controlling demand, and maximizing profit [10][12][14][15][16][17][19][23].

Matsuda and Whang studied pricing mechanisms for networks, maximizing the net utility of a network [14]. In particular, they characterized the equilibrium and the stability conditions of three dynamic pricing schemes.

Altmann et al. investigated the demand change under different pricing schemes within an empirical study [19]. They compared the differences in demand for Internet access service with respect to different usage-based pricing schemes and the flat rate pricing scheme. In a subsequent work, a decision support tool for helping Internet service providers (ISPs) to analyze pricing schemes, called Dynamic Netvalue Analyzer, was introduced by Altmann and Rhodes [17].

In a simulation conducted by Kephart et al., intelligent agents called pricebots automatically adapt product prices to changing market conditions, using a dynamic pricing algorithm [12]. In addition to this, they investigated shopbots. Shopbots respond to prices set by pricebots. They compare product prices, rank products based on customer preferences, and then make purchase decisions.

To capture the fact that perfect market knowledge is almost impossible to obtain in the real world, vendors with limited knowledge about customers and competitors can implement the derivative-follower pricing scheme [12]. Using this pricing scheme, vendor agents incrementally increase (or decrease) prices until they experience a drop in profit. Then, vendor agents decrease (or increase) prices. Dasgupta and Das developed a simulation that focuses on the derivative-follower pricing algorithm [16]. They proposed an algorithm that enhances the derivative-follower pricing by re-estimating the price-profit relationship for vendor agents for each time period. This work has been extended by Dasgupta and Hashimoto, who applied collaborative filtering for analyzing customer preferences. Their algorithm uses the result of the collaborative filtering as input to a dynamic pricing algorithm, which calculates the profit-maximizing price [15].

In another research paper by Lehmann and Buxmann, a pricing algorithm uses knowledge about customer preferences [10]. Because of that, the algorithm, which is referred to as a demand-driven pricing algorithm, achieves profit maximization. Precisely identifying customer preferences becomes a critical success factor for the demand-driven pricing algorithm [23]. However, acquiring perfect information of customer preferences requires huge amount of resources. These resources are usually only available to large firms.

In order to attract customers with low reservation prices, the penetration pricing scheme can be implemented. This pricing scheme initially sets prices lower than the prices of competitors, and then gradually increasing price [10]. The objective is to benefit from lock-in effects, i.e., to benefit from customers whose cost for switching to a new provider is high. Penetration pricing scheme is aimed at customers with high price sensitivity [22].

The polar opposite of penetration pricing is called skimming pricing. A skimming pricing scheme targets customers with high reservation prices or inelastic demand [22]. This pricing scheme creates high prices for products that are released newly, and then continuously lowers prices to capture customers with lower reservation prices [10]. High-tech vendors or innovative software vendors use this pricing scheme to sell software.

3. Simulation Model

3.1. Simulation Scenario

The simulation scenario represents a software market, in which customers purchase software licenses, and vendors offer software licenses at prices that depend on the pricing scheme used. Customers make decisions on purchasing either a Software-as-a-Service (SaaS) solution or a software solution with a perpetual license (PS) from one of the software vendors. A customer cannot opt for not choosing any of those two solutions.

Consequently, there are two types of agents within the simulation model: customer agents and software vendor agents. Customer agents are characterized through criteria values that reflect their organization size. Based on these criteria values and prices offered by vendor agents, customer agents make purchase decisions on the most appropriate software license solution. The software vendor agents are further categorized into two sub-

types: PS vendor agents, and SaaS vendor agents. Both vendor agents offer their software licenses at prices that follow a specific pricing scheme.

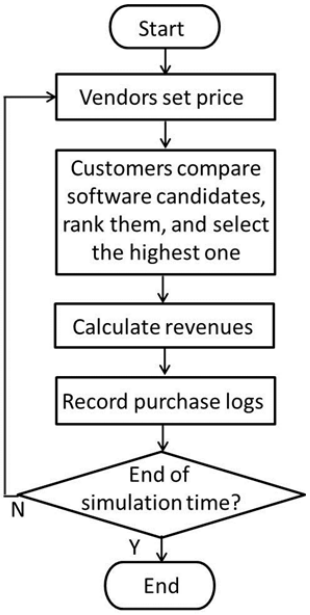

Figure 1. Simulation workflow.

The steps of the simulation are shown in Figure 1. After the simulation has started, each vendor agent calculates a price for its software solution using its dynamic pricing algorithm. Customer agents respond to the prices by calculating scores for both software solutions (i.e., SaaS and PS solutions) based on a decision support model. Taking the scores obtained, the customer agents select the software option with the highest score. Consequently, the revenue of one of the vendor agents is increased. For analyzing the development of this software market off-line, the scores, the selections, and the revenues are recorded during the simulation run.

As Figure 1 also shows, the previously described steps are executed repeatedly until the end of the simulation time (i.e., a pre-set number of iterations) has been reached. Depending on the type of our investigation, the number of iterations is 300 or 3200 iterations.

3.2. Customer Agents

3.2.1. Customer Agent Types

Based on the customer classification that was described by Cai et al. [21], customer agents in this simulation are grouped into three categories: small enterprises, medium-sized enterprises, and large enterprises. The difference between these classes is the number of employees of these enterprises: less than 20 employees are employed in a small enterprise, between 20 and 250 employees are employed in a medium-sized enterprise, while a large

enterprise has at least 250 employees. These values are based on the European Commission's SME guide [28].

Depending on the number of employees, these classes of enterprises have a different demand for software resources. The need for the type of applications, however, is assumed to be identical for all categories of customers. The range of application types goes from basic applications (e.g., office suite software, multimedia software, games, education software, browser, anti-virus) to software for customer relationship management (CRM), finance management, supply chain management, data mining, and data center management. Consequently, we can define the demand for software resources as a percentage value of software usage (Table 1). The higher the number of employees in the company is, the higher the software usage. The 66% for large enterprises is based on the percentage values of software use that have been discussed in literature. These values range between 60% and 80% for perpetual software [2].

Table 1. Customer agent categories and their parameter values.

Category	Number of Employees	Usage Time of Software
Small enterprises	< 20 (average 10)	12 / 36 months (30%)
Medium-sized enterprises	≥20, ≤250 (average 100)	24 / 60 months (40%)
Large enterprise	> 250 (average 1000)	48 / 72 months (66%)

The depreciation periods of software purchases for small enterprises, medium-sized enterprises, and large enterprises is set to 36 months, 60 months, and 72 months, respectively. The reason is the different long-term perspective of these types of enterprises.

3.2.2 Decision Making of Customer Agents

Depending on the organization size (i.e., small, medium, or large organizations), each organization has different preferences. Therefore, customer agent parameters are populated according to the size of the organization. The parameters and the values of these parameters have been identified in existing literature about software selection and SaaS adoption [1][3][4][5][6][7][8][9]. Based on these parameters and values, customer agents score each software solutions and, then, make a purchase decision for the software solution with the highest score. The chosen software product (i.e., either SaaS or perpetual software) is the most suitable product according to the customer agents' profile.

3.2.2.1 Criteria

For scoring each software solution, customer agents use the Analytic Hierarchy Process (AHP) method. Following this method, all criteria are organized in a decision tree, which shows the hierarchy between the criteria. The criteria and the decision hierarchy of all criteria of our simulation model are illustrated in Figure 2.

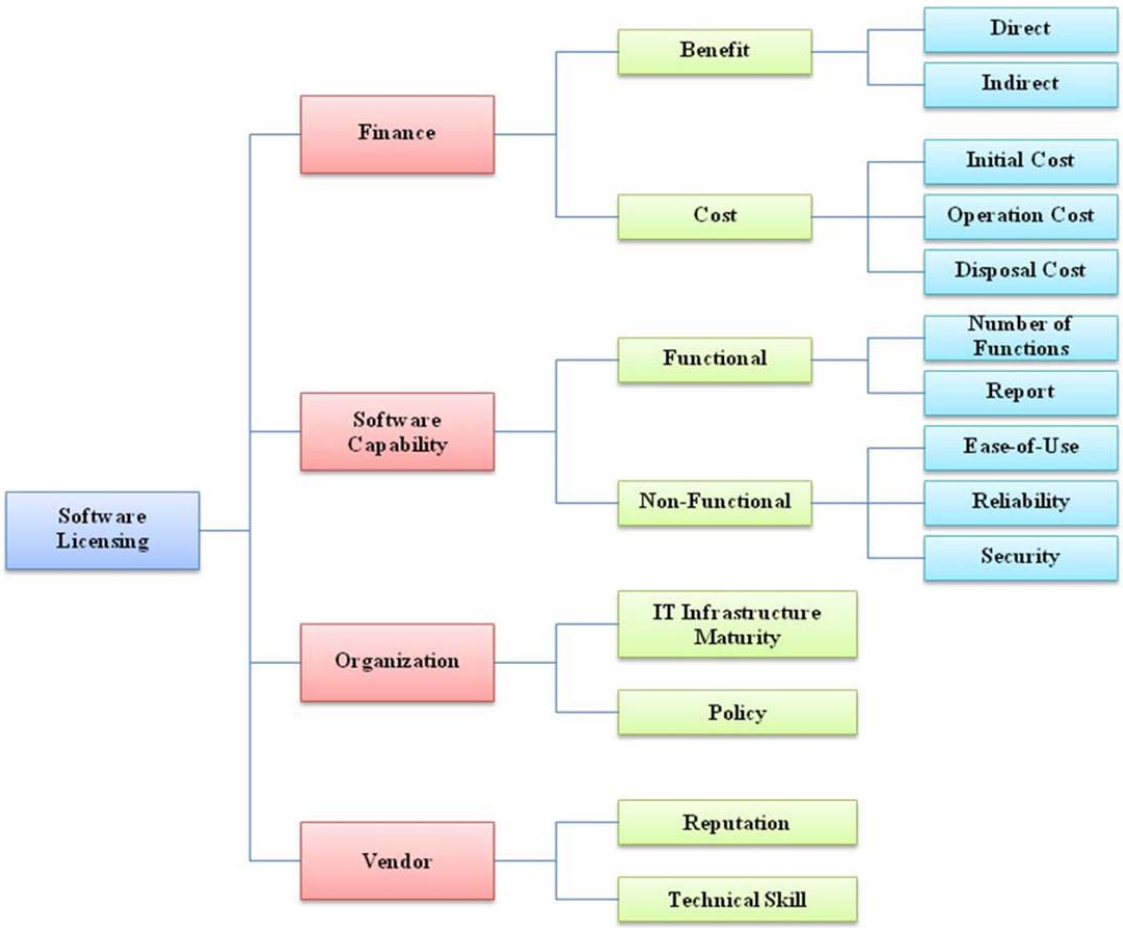

Figure 2. Decision hierarchy of software license criteria.

The four main groups of criteria in the software selection decision making are: Finance, Software Capability, Organization, and Vendor.

Finance is usually the top priority for chief information officers (CIOs) of companies. The Finance criterion includes the Benefit criterion and the Cost criterion. The Benefit criterion refers to the value that buyers gain from using the software. It can be classified into Direct Benefits and Indirect Benefits. The Direct Benefits criterion refers to the utility that a customer gains from using the software. The utility function U is defined as a product:

$$U\,(T_u\,, N_u\,, N_f\,, V_f\,, u) \;=\; f\,(T_u \cdot\; N_u \cdot N_f \;\cdot\; V_f \;\cdot u) \qquad (1)$$

where T_u is the software usage period (measured in units of months) before the software gets replaced, N_u is the number of users that use the software within the company, N_f is the number of software functions, V_f is the average utility of all software functions, u

represents the percentage of use of the software solution, and *f* is the function for making the product unit-less.

The indirect benefit criterion captures the gain from using a new technology. In our case, it includes the increase in public image of the organization through the adaption of the new SaaS technology. Besides, by adopting SaaS, firms can pay for the software cost monthly instead of having a one-time, large, initial payment. Although these benefits do not give directly monetary returns to an organization, they positively impact the firm as a whole. The indirect benefit for perpetual software comes from the authority in data management as well as the ownership of hardware and software.

The cost (i.e., total cost) of software includes three criteria: Initial Cost (capital cost), Operation Cost, and Disposal Cost. The initial cost criterion refers to a one-time payment to a vendor. The disposal cost criterion denotes the expenditure, if users abandon the use of the software solution or switch to a new software solution. The operation cost criterion refers to cost that incurs during the software usage period. The operation costs for perpetual software and for SaaS are defined as:

$$C_{OP\ (PS)} = T_u \cdot C_{server} \tag{2}$$

$$C_{OP\ (SaaS)} = T_u \cdot N_u \cdot (C_n + C_{sub}) \tag{3}$$

where C_{server} is the monthly operation cost for operating a server running perpetual software. C_n is the monthly cost for transferring data across a computer network, and C_{sub} is the monthly subscription rate (i.e., pay-per-use cost) for the software service.

The software capability criterion refers to software performance. It includes Functional and Non-Functional requirements. The functional requirements criterion refers to software attributes that are related to software output, which can be classified through the Number of Functions available through the software or the Number of Reports that are produced by the software. The non-functional requirements criterion includes the criteria Ease-of-Use, Reliability, and Security. They specify the simplicity of handling the software, the reliability of the software, and the software protection against misuse and theft.

The Organization criterion refers to factors within an organization that affects new software implementations. These factors can be grouped into IT Infrastructure Maturity and Policy. The IT infrastructure criterion considers factors such as staff knowledge, computer equipment, facilities for computers, firm culture, and network connectivity. The policy criterion describes the organization's policy towards IT, which can range from low priority to high priority.

The final criterion, the vendor criterion, includes Reputation and Technical Skill. As an indication of the vendor reputation, the number of clients of the vendor can be taken [6]. The technical skill criterion includes factors such as client training quality, technical support quality, and their experience in developing IT products or IT services.

3.2.2.2 Pairwise Comparison

Using AHP, each customer agent has also to perform pairwise comparisons of all selection criteria, in order to calculate the weights for each criterion [13]. For the pairwise comparison, the agents assign a value for each pair of criteria, ranging from 1 (equally favored) to 9 (extreme favored). The value describes the relative importance of one criterion over another criterion. In our simulation, the pairwise comparisons that are performed by the different customer agent types (small, medium, and large firms) differ depending on their preferences. The preferences for each criterion are based on general firm attributes as found in literature [28].

After obtaining the results of all pairwise comparisons, the regular AHP method is applied [13]. The results of the AHP method are the local weights and the global weights for each of the criteria. They are illustrated in the following three tables.

Table 2. Criteria weights of a small enterprise.

Criteria	Global weight	Sub-criteria	Local weight	Global weight	Sub-criteria	Local weight	Global weight
Finance	0.473	Benefit	0.333	0.158	Direct	0.750	0.1185
					Indirect	0.250	0.0395
		Cost	0.667	0.315	Initial	0.709	0.2233
					Operation	0.214	0.0674
					Disposal	0.077	0.0243
Capability	0.311	Functional	0.750	0.233	Functions	0.750	0.175
					Reports	0.250	0.058
		Non-functional	0.250	0.078	Ease-of-use	0.537	0.042
					Reliable	0.195	0.015
					Security	0.268	0.021
Organization	0.086	IT Infra. Maturity	0.833				0.072
		Policy	0.167				0.014
Vendor	0.130	Reputation	0.250				0.033
		Technical skill	0.750				0.097

Table 3. Criteria weights of a medium-sized enterprise.

Criteria	Global weight	Sub-criteria	Local weight	Global weight	Sub-criteria	Local weight	Global weight
Finance	0.457	Benefit	0.333	0.152	Direct	0.833	0.127
					Indirect	0.167	0.025
		Cost	0.667	0.305	Initial	0.405	0.124
					Operation	0.480	0.146
					Disposal	0.115	0.035
Capability	0.284	Functional	0.750	0.213	Functions	0.800	0.170
					Reports	0.200	0.043
		Non-functional	0.250	0.071	Ease-of-use	0.532	0.038
					Reliable	0.102	0.007
					Security	0.366	0.026
Organization	0.061	IT Infra. Maturity	0.833				0.051
		Policy	0.167				0.010
Vendor	0.198	Reputation	0.167				0.033
		Technical skill	0.833				0.165

Table 4. Criteria weights of a large enterprise.

Criteria	Global weight	Sub-criteria	Local weight	Global weight	Sub-criteria	Local weight	Global weight
Finance	0.162	Benefit	0.250	0.041	Direct	0.833	0.034
					Indirect	0.167	0.007
		Cost	0.750	0.121	Initial	0.214	0.026
					Operation	0.709	0.086
					Disposal	0.077	0.009
Capability	0.547	Functional	0.750	0.410	Functions	0.875	0.359
					Reports	0.125	0.051
		Non-functional	0.250	0.137	Ease-of-use	0.090	0.0123
					Reliable	0.354	0.0485
					Security	0.556	0.0762
Organization	0.218	IT Infra. Maturity	0.750				0.164
		Policy	0.250				0.054
Vendor	0.073	Reputation	0.250				0.018
		Technical skill	0.750				0.055

The difference of the global weights of the criteria, as shown in Table 2, Table 3, and Table 4, are caused by the difference in the pairwise comparison of the criteria for small enterprises, medium enterprises, and large enterprises. The global weights illustrate that small and medium-sized enterprises set the finance criterion as the most important criterion. This is a result of a low budget for IT resources. The finance criterion is followed by the criteria capability, vendor, and organization. With respect to large enterprises, the capability and organization criteria are prioritized the highest. The finance criterion has not the highest priority as large enterprises have a large IT budget available.

In our simulation, a customer agent is assigned the weights of Figure 3, Figure 4, and Figure 5, depending on whether it represents a small, medium, or large enterprise. Then, when a customer agent has to evaluate the software options offered by vendor agents, it uses its criteria weights.

3.3 Vendor Agents

3.3.1. Parameter Settings of Vendor Agents

The vendor agent parameters in the simulation are set according to information gained from two well-known SaaS and PS providers. The SaaS vendor parameters are based on the CRM product of Salesforce.com (CRM Professional Edition) [34], while the PS vendor parameters are based on Microsoft's CRM on-premise software [35].

There are two important parameters for vendor agents: the one-time price and the subscription rate. The initial, one-time price refers to upfront payment that a customer has to pay when purchasing a perpetual software license. The initial subscription rate refers to

the monthly price that a customer has to pay to a software vendor for leasing and operating the application. Furthermore, we assume that there is no initial one-time price to pay for SaaS products and no subscription price to pay for PS products. These parameter values of the vendor agents are shown in Table 5.

Table 5. Vendor agent categories and their parameter values.

Category	Initial, One-Time Price	Initial Subscription Rate
SaaS	-	$65
PS	$800	-

In order to avoid the complication in calculating revenues, we assume that the development costs and the maintenance costs for vendors are zero.

3.3.2. Dynamic Pricing Schemes of Vendor Agents

To ensure success in business, software vendors (i.e., SaaS vendors and PS vendors) need to have solid pricing schemes. In particular, software vendors require pricing instruments that work for selling technology products, which get quickly obsolete. In order to address this issue, dynamic pricing schemes are an option. These pricing schemes take an initial price (as set in section 3.3.1) and adapt them to the market situation continuously.

In the following subsections, four dynamic pricing schemes are introduced that have been frequently investigated in academic research and are applied in the software industry. These pricing schemes are the derivative-follower (DF) pricing scheme, the demand-driven (DD) pricing scheme, the penetration (PN) pricing scheme, and the skimming (SK) pricing scheme. In addition to this, the flat-rate (fixed-price) pricing scheme (FP) is used as a reference pricing scheme. Under this pricing scheme, the price does not change.

These pricing schemes are used to understand the interplay between these dynamic pricing plans in the market and to demonstrate how software vendors can benefit from dynamic pricing schemes.

3.3.2.1 Derivative-Follower Pricing Scheme

The derivative-follower pricing scheme (DF) is suited for vendors, who have little knowledge about the market condition and the actual demand of customers [16]. Under this pricing scheme, vendors just need to increase or decrease their prices depending on the results of their sales. Using this trial-and-error approach, they can try to optimize their revenues.

In detail, if a vendor offered his customers a price p_t at time interval t and a price p_{t-1} at time interval $t-1$, where $p_t > p_{t-1}$, and the revenue at time t is larger than the revenue at time $t-1$ ($R_t > R_{t-1}$), then the vendor would increase its price p_{t+1} in time interval $t+1$ as well. Otherwise, the vendor would decrease its price p_{t+1}. Therefore, the price p_{t+1} at time interval $t+1$ is defined as follows:

$$p_{t+1} = p_t + \delta \text{ if } R_t \geq R_{t-1} \text{ at time interval } t, \text{ and}$$

$$p_{t+1} = p_t - \delta \text{ if } R_t < R_{t-1} \text{ at time interval } t \quad (4)$$

where δ is a constant though different for different products, and R_t is revenue at time interval t. This algorithm repeats infinitely.

From the initial price setting of SaaS and PS (Table 5) for our simulation, we assume that SaaS products have a smaller price range than PS products. Therefore, the default value for δ_{SaaS} is set to 1, while the default value δ_{PS} is set to 10.

3.3.2.2 Demand-Driven Pricing Scheme

Using demand-driven (DD) pricing schemes, vendors set prices based on their knowledge about the perceived values of customers. In order to calculate the optimal price, this pricing scheme requires complete information on customer preferences and competitor behaviors.

Demand-driven pricing schemes are market-specific. Customers in different markets have different value perceptions [23]. The key success factor for this pricing scheme is that vendors can clearly identify customer requirements and customers' willingness to pay. Therefore, vendors have to balance the tradeoff between the cost for acquiring knowledge and the potential revenue obtained from this [23]. Since capturing of customer's perceived value is costly, large vendors have an advantage over smaller vendors because of their large amount of resources available for market research.

Examples of software vendors, who implemented this pricing scheme, are the leading software vendors. Microsoft implements a consumption-based pricing plan for its cloud computing product, called Microsoft Azure [31]. Salesforce, Netsuite, and Workday use this pricing scheme for their SaaS products. These software vendors categorized their software offerings such that a category matches the needs of a specific group of customers in the market. This classification resulted in numerous software versions (e.g., individual user version, professional version, enterprise version), targeting specific customer groups. Each software version comprises a different set of features.

Formally, the price p_{t+1} at time interval $t+1$ can be defined as shown in the following equation:

$$p_{t+1} = \arg \max_{pM} (D(\textit{Customer Preferences, Competitor Behavior, } p_M) * p_M) \quad (5)$$

where $D()$ denotes the customer demand for a software version at a certain price p_M, assuming a specific set of customer preferences and competitor behaviors.

For our simulation, we assume that the vendor agent has knowledge about the customer preferences (i.e., knowledge about the three criteria weight settings as described in section 3.2.2.2). The vendor agent selects the customer preferences depending on its target market. With respect to the competitor behavior, we only consider the prices of the competitors in

the current round t in our simulation. It allows comparing the vendor's price with the prices of the competitors.

For setting the price such that the maximum revenue is obtained, the software vendor runs the AHP model of the software consumer agents with prices p_M in the range between $p_t - \alpha$ and $p_t + \alpha$, where α is constant but different for SaaS products (α_{SaaS}) and PS products (α_{PS}). α_{SaaS} is set to 10 and α_{PS} is set to 100, since PS has a larger price range than SaaS. The price p_t of the previous time period is taken as a starting point. The price p_t, which results in the highest revenue, will be taken as the new price p_{t+1}.

3.3.2.3 Penetration Pricing Scheme

The principle behind the penetration pricing scheme (PN) is to use low prices for gaining market share. Penetration pricing schemes target customers, who have a high level of price sensitivity and a low reservation price [22]. This kind of customers comprises the largest segment of the software market. After reaching market share and successfully creating lock-in effects, the software vendor can upgrade its software and generate additional revenue [10].

Penetration pricing schemes are frequently used in business-to-consumer (B2C) scenarios. Walmart with its motto "always low prices" is a good example. In the software industry, some large enterprises also implemented this pricing scheme. For example, Google offers a free SaaS version of an Office suite, in order to compete with Microsoft on its products. Microsoft also lowered the price of its Window and Office suits (or provided less powerful versions) in order to capture new markets (i.e., users who would not be able to purchase the full software at full price) [32].

The disadvantage of this pricing scheme becomes obvious, if the lock-in effect fails and prices are increased. In this case, the vendor is at risk of losing customers. Customers would leave, since the software product was expected to stay cheap.

In our simulation, the initial prices of PN vendors are set lower than its competitors. When PN vendors reach its target market share, the prices are increased to generate higher revenues. The lock-in effect can be modeled as a bonus L in the current price $(p_t = p_t - L)$. In our simulation, it is set to $L = 0$. Besides, if PN vendors raise their prices until their market shares S_t are lower than they expect, the vendors start reducing their prices again. This penetration pricing plan can be formally expressed with the following equation:

$$p_{t+1} = p_t + \delta \text{ if } S_t >= S_T \text{ at time interval } t, \text{ and}$$

$$p_{t+1} = p_t - \delta \text{ if } S_t < S_T \text{ at time interval } t \quad (6)$$

where δ is a constant, S_t is the market share at the last time period t, and S_T is a constant that represents the target market share set by the software vendor. Market share S_t can be calculated with the following equation:

$$S_t = n_t / N \quad (7)$$

where n_t is the number of sales or number of customers at time t, and N is the sample population or the total number of customers.

3.3.2.4 Skimming Pricing Scheme (SK)

The skimming pricing scheme sets initially a high price and, then, gradually reduces the prices [10]. The objective of this pricing scheme is to target customers, who have the highest willingness to pay. The initially targeted customers tend to be insensitive to prices [22]. This pricing scheme is the polar opposite to the penetration pricing scheme.

The skimming pricing scheme is typically found in selling prestige goods or premium goods. To implement this strategy, the software product quality and image should be high enough to match the high price. Vendors intend to gain maximum profit from a market for hi-tech or new innovative products (e.g., HDTV, game software, and MP3 player, camera). Typical examples are found in the game industry. Game software companies set high initial prices to sell their software to forward-looking customers and, then, reduce prices later to sell to customers with a lower reservation price [33].

The disadvantage of this pricing scheme is that, if vendors lower the prices too fast early buyers could get negative feelings about being ripped-off.

To simulate the skimming pricing scheme, we set the initial price of SK venders higher than prices of their competitors. If their sales cannot reach target market share, SK vendors start reducing its price. The equation is formed as described below:

$$p_{t+1} = p_t - \delta \text{ if } S_t < S_T \text{ at time interval } t, \text{ and}$$

$$p_{t+1} = p_t + \delta \quad \text{if } S_t >= S_T \text{ at time interval } t \tag{8}$$

where δ is a constant, S_t is the market share at the last time period t, and S_T is a constant that represents the target market share set by a software vendor. The market share S_t can be calculated with the following equation:

$$S_t = n_t / N \tag{9}$$

where n_t is the number of sales or number of customers at time t, and N is the sample population or the total number of customers.

4. Analysis of Simulation Results

In order to investigate the performance of each pricing scheme, we use our simulation model of the software market, in which customers purchase software from SaaS vendors or from PS vendors. For this, we compare the different pricing schemes (Section 3.3.2) for SaaS products and PS products by letting different types of customer agents make purchase decisions on these software products.

The value settings of the constants used in the pricing schemes of our simulations are summarized in Table 6.

Table 6. Constant values used in the simulations.

Constant	Description	Value
δ_{SaaS}	Constant for increasing and reducing prices for SaaS	1
δ_{PS}	Constant for increasing and reducing prices for PS	10
α_{SaaS}	Constant for setting price range in DD simulations for SaaS	10
α_{PS}	Constant for setting price range in DD simulations for PS	100
$S_{T\,(PN)}$	Target market share for PN simulations	0.3
$S_{T\,(SK)}$	Target market share for SK simulations	0.2

Since the prices of SaaS products are usually lower than PS products, the values of the constants for SaaS are lower than those of PS as well. The target market share of SK is set a little bit lower than the target market share of PN. It makes sense to assume that the number of customers, who are attracted by SK pricing (i.e., are willing to pay a high price), is lower than the number of customers, who are the target of the PN pricing and prefer to pay lower prices.

The number of agents in our simulation depends on the simulation experiment. The rule applied is that there is exactly one software vendor agent for each software option offered and pricing scheme used. Therefore, the number of vendor agents can vary between 2 and 10 agents. The number of customer agents corresponds to the number of customer types used in the simulation run. This number can range between 1 agent and 3 agents.

4.1 Pricing Scheme Comparison in Single Product Markets

To understand how effective the different pricing schemes are, the five different pricing schemes are compared in markets under controlled conditions. That means, we distinguish between markets, in which only SaaS products or PS products are traded, and markets, in which only a specific set of customer types exists. Figure 3 illustrates the results of the pricing scheme simulation of three markets, in which only SaaS products are traded. The three markets shown differ with respect to the customer base. There is a market with small enterprises as customers only, a market with large enterprises as customers only, and a market with all three types of customer agents. In these markets, five SaaS products are offered at five different pricing schemes.

Figure 4 presents the simulation results of three markets, in which PS products are traded only. The three markets differ in the same way as in Figure 3. There is a market with small-sized enterprises only, a market with large-sized enterprises only, and a market with all three types of enterprises. The five PS products are offered at five different pricing schemes.

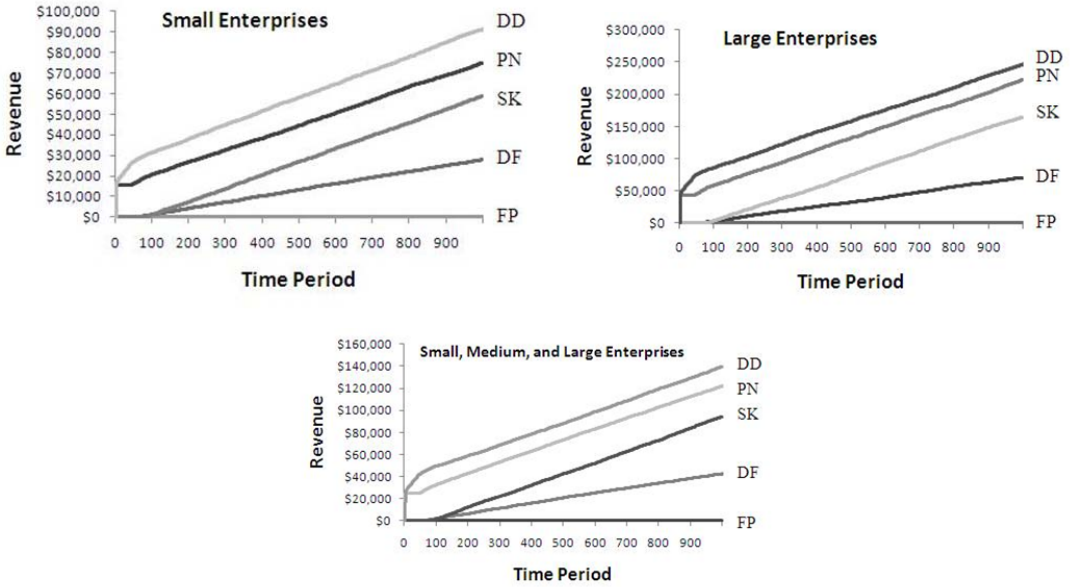

Figure 3. Pricing scheme comparison for three different markets (i.e., market with small-sized enterprises, market with large-sized enterprises, and market with all types of enterprises), in which five SaaS products are offered using five different pricing schemes (i.e., FP, DF, PN, SK, DD).

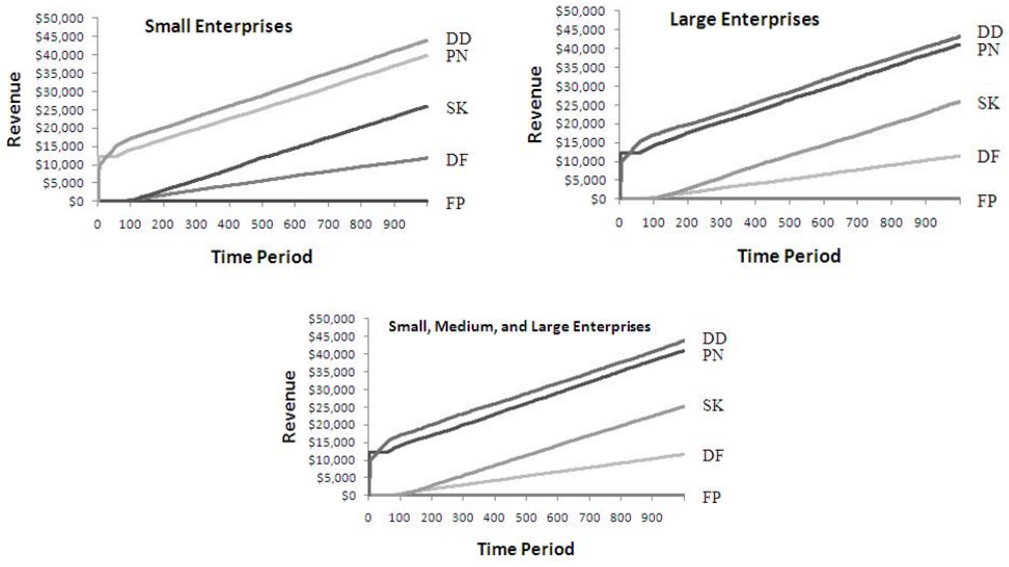

Figure 4. Pricing scheme comparison for three different markets (i.e., market with small-sized enterprises, market with large-sized enterprises, and market with all types of enterprises), in which five PS products are offered using five different pricing schemes (i.e., FP, DF, PN, SK, DD).

The simulation results of Figure 3 and Figure 4 show similar results. Although they differ in their absolute values, the shapes of the curves are similar. This finding is expected, since only one type of good is being traded. In this case of no competition, the results represent the workings of the pricing schemes only.

Moreover, the most effective pricing scheme for both products (i.e., SaaS and PS), is *DD*. It generates the highest revenue for both products. The order of the succeeding pricing schemes is *PN*, *DF*, *SK*, and *FP*.

4.2 Comparison of All Pricing Schemes in a Market for SaaS and PS Products

After having shown the effects of the five pricing schemes on the revenue in a market with a single product, this simulation experiment investigates how SaaS and PS products compete in the same market. The main objective of this simulation is to find the most effective combination of pricing scheme and software licensing option.

The SaaS and PS products are offered in five pricing schemes in a market with small, medium, and large enterprises as customers. Therefore, the total number of vendor agents is 10. Five vendor agents offer a SaaS product and five vendor agents offer a PS product. Each of the five vendors offers its product under a different pricing scheme. These 10 vendors are denoted with FP_{SaaS}, FP_{PS}, DF_{SaaS}, DF_{PS}, PN_{SaaS}, PN_{PS}, SK_{SaaS}, SK_{PS}, DD_{SaaS}, and DD_{PS} (Table 7).

Table 7. Vendor agent types.

Vendor	Type	Scheme	Vendor	Type	Scheme
FP_{SaaS}	SaaS	Fixed Price	PN_{PS}	PS	Penetration
FP_{PS}	PS	Fixed Price	SK_{SaaS}	SaaS	Skimming
DF_{SaaS}	SaaS	Derivative Follower	SK_{PS}	PS	Skimming
DF_{PS}	PS	Derivative Follower	DD_{SaaS}	SaaS	Demand Driven
PN_{SaaS}	SaaS	Penetration	DD_{PS}	PS	Demand Driven

Figure 5 illustrates the result of the pricing scheme simulations for the duration of 3200 time periods. The figure shows the increase in revenue for each time period.

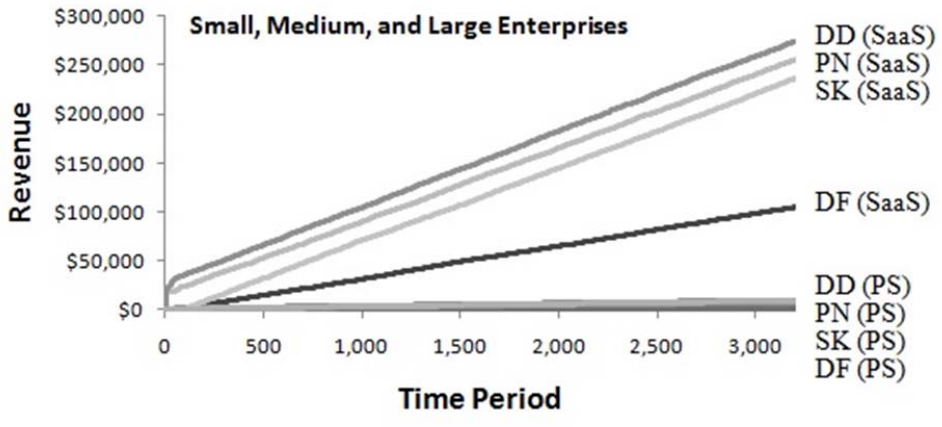

Figure 5. Comparison of all combinations of pricing schemes and software licensing types in a market with small, medium, and large enterprises.

Figure 5 illustrates that only 8 software vendors (DF_{SaaS}, DF_{PS}, PN_{SaaS}, PN_{PS}, SK_{SaaS}, SK_{PS}, DD_{SaaS}, and DD_{PS}) of the 10 software vendors can attract customers. The most successful vendor is DD_{SaaS}. The subsequent ones are PN_{SaaS}, SK_{SaaS}, DF_{SaaS}, DD_{PS}, PN_{PS}, SK_{PS}, and DF_{PS}. Both FP vendors (FP_{SaaS} and FP_{PS}) cannot generate any revenues.

In the beginning of the simulation (i.e., < 100 time periods), PN_{SaaS} successfully penetrates the market with its initial low price offerings. As PN_{SaaS} gradually raises its prices, since it reached its market share as planned, the other vendors start cutting their prices. At the point when PN_{SaaS} charges prices higher than DD_{SaaS}, customer agents start to purchase software from DD_{SaaS}. Vendor DD_{SaaS} continues generating revenue, while the other vendors reduce their prices to be able to compete with DD_{SaaS}. For the rest of the simulated time periods, the DD_{SaaS} performs always a little better than PN_{SaaS}. Vendor SK_{SaaS}, who does not get any revenue in the beginning due to prices that were higher than the prices of the other SaaS vendors, later gains portions of the market and is ranked third in the end of the simulation. Vendor DF_{SaaS} generates less revenue due to its trial-and-error approach and ends up in the fourth place. The PS vendors, except for FP_{PS} that does not generate any revenue, are only able to obtain small portions of the overall revenue.

In order to identify the impact of the customer type on the simulation results, we repeat the previous simulation with only one customer type. We simulate the competition between vendors in a market with small enterprises only as well as the competition between vendors in a market with large enterprises only. The simulation results are shown in Figure 6 and Figure 7, respectively.

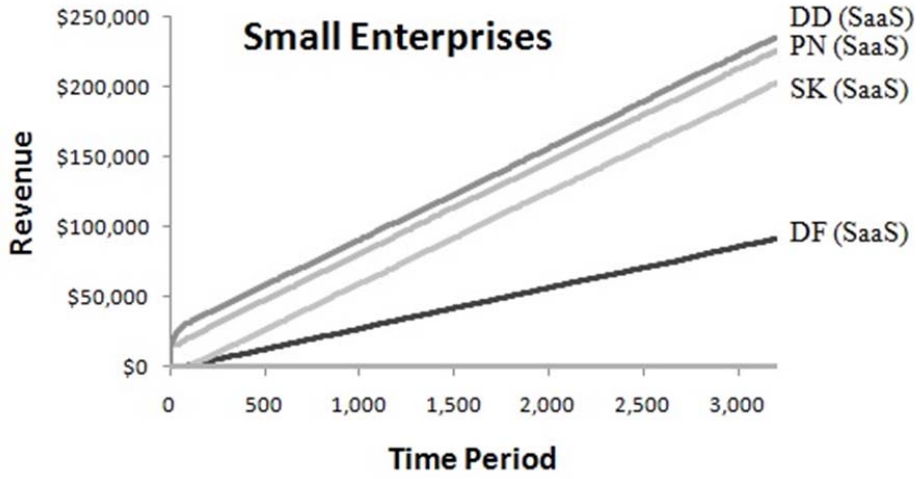

Figure 6. Comparison of all combinations of pricing schemes and software licensing types in a market with small enterprises only.

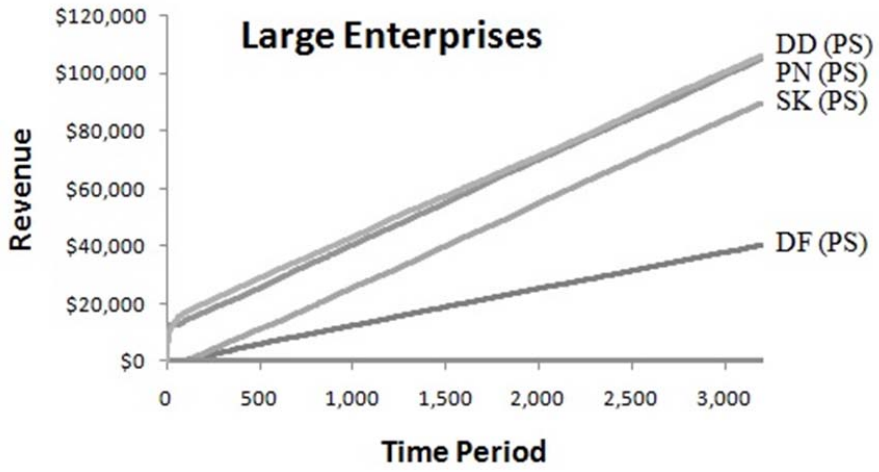

Figure 7. Comparison of all combinations of pricing schemes and software licensing types in a market with large enterprises only.

The results shown in Figure 6 and Figure 7 are similar to the result of the simulation in Figure 5. The most effective pricing scheme is still DD, followed by PN, SK, and DF. In addition to this, Figure 6 also indicates that small enterprises prefer purchasing SaaS products. Figure 7 indicates that large enterprises favor PS products.

In addition to this, as can be seen from a comparison of Figure 6 and Figure 7, the absolute values of revenue differ between SaaS product pricing and PS product pricing. While the revenue reaches only $110,000 in the market with large enterprises, the revenue reaches $240,000 in the market with small enterprises. These absolute values are a

consequence of the parameter setting of enterprises and their demand. Therefore, if PS vendors would decrease the price of their products or the SaaS vendors would increase their prices, the outcome of the simulation run (Figure 5) would result in a mixture of the order of SaaS and PS pricing schemes.

Based on the results depicted in Figure 6 and Figure 7, we can add another interpretation of the simulation results shown in Figure 5. The reason that all SaaS vendors generate more revenues than the PS vendors is simply a consequence of the customer base composition. The medium-sized enterprises are modeled to consume rather SaaS products than PS products. Therefore, in order to attract more medium-sized enterprises, the prices for PS products would have to be lowered.

4.3 Comparison of Penetration Pricing (PN) and Demand Driven Pricing (DD)

As a result of Section 4.2, the PN pricing scheme and the DD pricing scheme have been identified as the top 2 most effective pricing schemes among the five investigated. Consequently, the question rises what would happen, if all vendors implement these two pricing schemes only. To answer this question, we conduct simulations with small, medium, and large enterprises using only the PN pricing scheme and the DD pricing scheme in a market with small, medium, and large enterprises as customers. Figure 8 presents the result of this simulation.

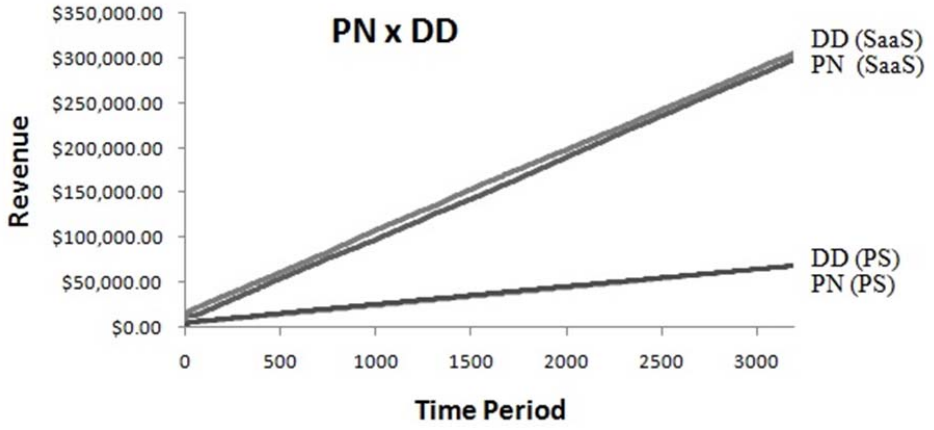

Figure 8. Comparison of the penetration (PN) pricing scheme and the demand driven (DD) pricing scheme in a market with small-sized, medium-sized, and large-sized enterprises.

Figure 8 shows that DD_{SaaS} generates the most revenue, followed by PN_{SaaS}, DD_{PS}, and PN_{PS}. The interesting fact is the little gap between the revenue of DD_{SaaS} and the revenue of PN_{SaaS}. Therefore, we can assume that these two pricing plans are substitutable. Based on the simulation results of Figure 8, we can also state that DD is still the most effective pricing scheme. However, since the implementation of DD is costly and not achievable to 100%, PN is a suitable alternative in real-world scenarios.

4.4 Comparison of All Pricing Schemes Excluding Demand Driven Pricing

The results shown in the previous sections illustrate that the DD pricing scheme is the most effective scheme in our simulations. However, implementing DD in reality is difficult, since it is almost impossible to collect perfect knowledge on both customers and competitors. To study the market evolution without the impact of the dominating strategy of using the DD pricing scheme, we simulate the pricing competition without vendors that apply the DD pricing scheme. The result is shown in Figure 9.

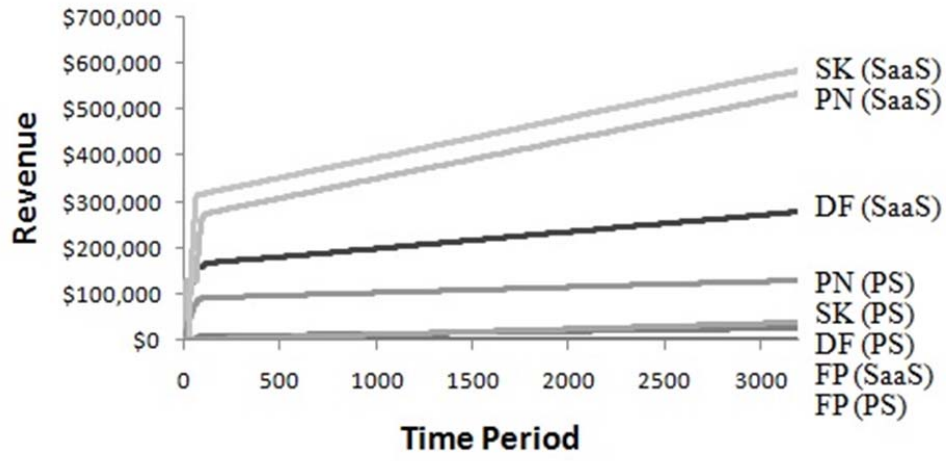

Figure 9. Comparison of all pricing schemes excluding the demand-driven pricing scheme in a market with small-sized, medium-sized, and large-sized enterprises.

Figure 9 illustrates that the most effective pricing scheme in this simulation is SK_{SaaS}. PN_{SaaS} is on the second rank only and DF_{SaaS} on the third rank. In the beginning, the PN pricing schemes did well on penetrating the market and generating revenues. The pricing scheme SK_{SaaS} started generating revenues later than PN_{SaaS} but overcame PN_{SaaS} quickly. The reason is that PN_{SaaS} did not reach its market share in this simulation quickly enough and, therefore, could only follow the SK_{SaaS} pricing scheme. The pricing plan DF_{SaaS} comes in the third place. Since DF_{SaaS} changes its prices every time period (due to its algorithm that changes prices according to the gain or loss of revenue in the last period), there are chances that DF_{SaaS} raises its prices higher than competitors and, therefore, loses customers. SK_{SaaS} and PN_{SaaS} have less dynamic pricing (i.e., its prices are set depending on a market share threshold), which helps avoiding customer loss. Consequently, DF_{SaaS} generates less revenue than SK_{SaaS} and PN_{SaaS} later on in the simulation. With respect to the PS vendors, the PN_{PS} pricing scheme generates the highest revenue compared to other pricing schemes, as it can reach its market share quicker. The SK_{PS} pricing scheme shows the same behavior as the PN_{PS} pricing scheme after a slower start. The FP pricing schemes do not generate any revenues, as happened in the previous simulations (Figure 5).

From this pricing scheme simulation, we can conclude that the revenue threshold and the market share threshold have a significant impact in the beginning of the price setting simulation when prices did not converge to their stable values yet.

4.5 Comparison of Penetration (PN) and Skimming (SK) Pricing Scheme

The results in Section 4.4 suggest that SK and PN compete successfully in the market. Therefore, in this section, we generate a pricing competition between the PN pricing scheme and the SK pricing scheme only. The results of the simulation are shown in Figure 10.

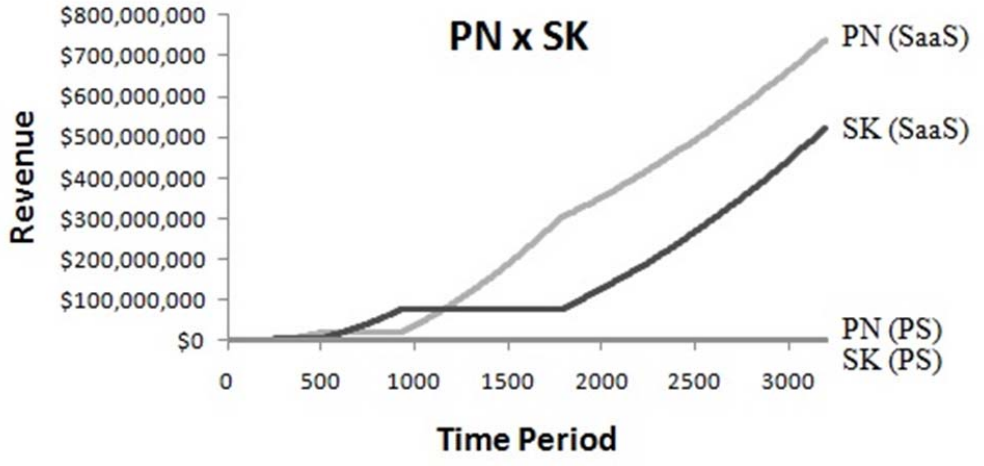

Figure 10. Comparison of the penetration (PN) pricing scheme and the skimming (SK) pricing scheme in a market with small-sized, medium-sized, and large-sized enterprises.

The result shows that PN_{SaaS} pricing scheme gains the most revenue in this simulation experiment. Although there is some time periods, in which SK_{SaaS} generates more revenues than PN_{SaaS}, PN_{SaaS} gains more revenue than SK_{SaaS} in the end. At the end of the simulation, the rate of their revenue generation is identical. As for PS vendors, their revenue is very small compared with the revenue of SaaS vendors, since customers only select them occasionally.

The reason for this behavior is that PN_{SaaS} pricing scheme gets its market share early on and, therefore, increases the price. At the time period 500, however, SK_{SaaS} prices are lower than the prices of PN_{SaaS}. During the following time periods, SK_{SaaS} continues to increase its revenue until time period 890. At this point PN_{SaaS} lost its market share and, therefore, lowers its prices. The prices became even lower than the one of the SK_{SaaS} pricing scheme. Therefore, PN_{SaaS} gains revenue during the following time periods until time period 1800. At this point, SK_{SaaS} lost its market share and starts offering lower, competitive prices. PN_{SaaS} can continue to sell with little lower prices. From this point onwards, the two

pricing schemes are sometimes briefly below the threshold and sometime briefly above the threshold, resulting in a almost linear curve. The slight convex shape of the curve also indicates a slight increase in the price over time. Considering the entire behavior of both pricing schemes, we can conclude that the penetration (PN) pricing scheme and the skimming (SK) pricing scheme show the same behavior in the long run. They can be considered equivalent.

5. Discussion and Conclusion

In this study, we developed an agent-based simulation of a software market that allows trading of two types of software licensing models (SaaS and PS) under four different dynamic pricing schemes. The four dynamic pricing schemes that have been considered are: derivative-follower (DF) pricing scheme, demand-driven (DD) pricing scheme, penetration (PN) pricing scheme, and skimming (SK) pricing scheme.

The simulation involves two types of agents: customer agents and vendor agents. The task of customer agents is to score and rank software options offered by vendor agents, using the Analytic Hierarchy Process (AHP) method. The customer agent parameters are set based on the European Commission's SME guide. There are three categories of customer agents, representing small-sized, medium-sized, or large-sized enterprises. Vendor agents set prices for their software offerings according to the pricing scheme deployed. Vendor agent parameters are based on real world products from popular vendors such as Salesforce.com and Microsoft. Vendor agents are further categorized into SaaS vendors and PS vendors.

Our simulation results reveal that the DD pricing scheme is the most effective pricing scheme compared to the other pricing schemes (Figure 5). However, in a real market, obtaining perfect information about customers and competitors is almost impossible, making the DD pricing scheme difficult to implement. Vendors can alternatively implement PN or SK, since they are simple to implement and almost as effective as the DD pricing scheme (Figure 8 and Figure 9).

In a simulation situation where the DD pricing scheme is excluded, SK or PN become the best-performing pricing schemes (Figure 9). Depending on which of the two pricing schemes reaches the market share threshold first, one of these pricing schemes will come out first with respect to revenue generation. Therefore, the penetration pricing scheme and the skimming pricing scheme can be considered equivalent.

Besides, in our simulation, SaaS vendors always generate revenues, which are higher than the revenues of PS vendors (Figure 5). This is simply explained by the composition of our customer base, which tends to have more small and medium-sized enterprises. If the customer base is composed of smaller firms, the customer population prefers a SaaS product over a PS product (Figure 6). Otherwise, the customer population prefers a PS product (Figure 7).

In the future, we will use the results from our simulation and our simulation model to find the equilibrium point between revenues of SaaS vendors and the revenues of PS vendors.

Acknowledgement

This work is supported by the National Research Foundation of the Ministry of Education, Science and Technology of Korea under the grant K21001001625-10B1300-03310.

References

[1] Wailgum, T.: What to Ask Before Saying Yes to SaaS, Cloud Computing. In: The New York Times, October 27, 2008 (2008).

[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: Above the Clouds: A Berkeley View of Cloud Computing. UC Berkeley, Reliable Adaptive Distributed Systems Laboratory (2009).

[3] Kaplan, J.: SaaS Survey Reveals Real World Experience, http://itmanagement.earthweb.com/features/article.php/3873121/SaaS-Survey-Reveals-Real-World-Experience.htm, accessed on December 2011 (2011).

[4] Colombo, E., Francalanci, C.: Selecting CRM Packages Based on Architectural, Functional, and Cost Requirements: Empirical Validation of a Hierarchical Ranking Model. In: Requirements Engineering. vol.9, pp.186-203 (2004).

[5] Jadhav, A.S., Sonar, R.M.: Evaluating and Selecting Software Packages: A Review. In: Information and Software Technology. vol.51, pp.555-563 (2009).

[6] Godse, M., Mulik, S.: An Approach for Selecting Software-as-a-Service (SaaS) Product. In: IEEE International Conference on Cloud Computing (2009).

[7] Lai, V.S., Trueblood, R.P., Wong, B.K.: Software Selection: A Case Study of the Application of the Analytical Hierarchical Process to the Selection of a Multimedia Authoring System. In: Information & Management. vol.36, pp.221-232 (1999).

[8] Mulebeke, J.A.W., Zheng, L.: Analytical Network Process for Software Selection in Product Development: A Case Study. In: Journal of Engineering and Technology Management. vol.23, pp.337-352 (2006).

[9] Rohitratana, J., Altmann, J.: Software Resource Management Considering the Interrelation between Explicit Cost, Energy Consumption, and Implicit Cost: A Decision Support Model for IT Managers. In: MKWI2010, Multikonferenz Wirtschaftsinformatik (2010).

[10] Lehmann, S., Buxmann, P.: Pricing Strategies of Software Vendors. In: Business and Information Systems Engineering. vol.6, pp.452-462 (2009).

[11] Marn, M.V., Roegner, E.V., Zawada, C.C.: Pricing New Products. The McKinsey Quarterly, http://www.mckinseyquarterly.com/article_print.aspx?L2=16&L3=19 &ar=1329.

[12] Kephart, J.O., Hanson, J.E., Greenwald, A.R.: Dynamic Pricing by Software Agents. In: Computer Networks. vol.32, pp.731-752 (2000).

[13] Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980).

[14] Matsuda, Y., Whang, S.: Dynamic Pricing for Network Service: Equilibrium and Stability. In: Management Science. vol.45, pp.857-869 (1999).

[15] Dasgupta, P., Hashimoto, Y.: Multi-attribute Dynamic Pricing for Online Markets using Intelligent Agents. In: Third International Joint Conference on Autonomous Agents and Multiagent Systems, pp.277-284, New York (2004).

[16] Dasgupta, P., Das, R.: Dynamic Pricing with Limited Competitor Information in a Multi-Agent Economy. In: 7th International Conference on Cooperative Information Systems. pp.299-310 (2000).

[17] Altmann, J., Rhodes, L.: Dynamic Netvalue Analyzer - A Pricing Plan Modeling Tool for ISPs Using Actual Network Usage Data. In: IEEE WECWIS2002, International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems, Newport Beach, USA (2002).

[18] Altmann, J., Varaiya, P.: INDEX Project: User Support for Buying QoS with Regard to User's Preferences. In: IWQOS'98, Sixth IEEE/IFIP International Workshop on Quality of Service , pp. 101-104, Napa, USA (1998).

[19] Altmann, J., Rupp, B., Varaiya, P.: Internet Demand under Different Pricing Schemes. In: ACM EC99, ACM Conference on Electronic Commerce. Denver, Colorado, USA (1999).

[20] Risch, M., Altmann, J.: Enabling Open Cloud Markets Through WS-Agreement Extensions. In: Service Level Agreements in Grids Workshop, in conjunction with GRID2009, CoreGRID Springer Series, Banff, Canada (2009).

[21] Cai, H., Chung, J.Y., Su, H.: Relooking at Services Science and Services Innovation. In: Service Oriented Computing and Applications, vol.2, pp.1-14 (2008).

[22] Monroe, K. B.: Pricing: Making Profitable Decisions. 3rd Edition, New York, McGraw-Hill Publishing Company (2003).

[23] Harmon, R., Demirkan, H., Helfley, B., Auseklis, N.: Pricing Strategies for Information Technology Services: A Value-Based Approach. In: Proceedings of the 42nd Hawaii International Conference on System Sciences (2009).

[24] Dias-Neto, A. C., Travassos, G. H.: Model-Based Testing Approaches Selection for Software Projects. In: Information and Software Technology, vol. 51, pp. 1487 – 1504 (2009).

[25] Kwong, C. K., Mu, L. F., Tang, J. F., Luo, X. G.: Optimization of Software Components Selection for Component-Based Software System Development. In: Computer & Industrial Engineering, vol. 58, pp. 618 – 624 (2010).

[26] Ullah, M. I., Ruhe, G., Garousi, V.: Decision Support for Moving from a Single Product to a Product Portfolio in Evolving Software Systems. In: The Journal of Systems and Software, vol. 83, pp. 2496 – 2512 (2010).

[27] Ayala, C., Hauge, O., Conradi, R., Franch, X., Li, J.: Selection of Third Party Software in Off-The-Shelf-Based Software Development – An Interview Study with Practitioners. In: The Journal of Systems and Software (2010), doi:10.1016/j.jss.2010.10.019.

[28] European Commission: The New SME Definition: User Guide and Model Declaration (2003).

[29] Mustajoki, J., Hämäläinen, R.: Web-HIPRE: Global Decision Support by Value Tree and AHP Analysis. In: INFOR, Vol. 38, pp. 208 - 220 (2000).

[30] Karlsson, J., Wohlin, C., Regnell, B.: An Evaluation of Methods for Prioritizing Software Requirements. In: Information and Software Technology, vol. 39, pp. 939 - 947 (1998).

[31] Foley, M.-J.: Azure Pricing: How Low will Microsoft Go? http://www.zdnet.com/blog/microsoft/azure-pricing-how-low-will-microsoft-go/3235?tag=mantle_skin;content, accessed on December 2011 (2011).

[32] Burrows, P.: Microsoft's Aggressive New Pricing Strategy. In: Business Week, July 27, 2009 (2009).

[33] Nair, H.: Intertemporal Price Discrimination with Forward-Looking Customers: Application to the US Market for Console Video-Games. In: Quantitative Marketing and Economics, vol. 5, pp. 239 – 292 (2007).

[34] Salesforce.com inc.: CRM Professional Edition, http://www.salesforce.com/crm_test_blitz/editions-pricing/professional-edition/, accessed on December 2011 (2011).

[35] Microsoft: Dynamics CRM (formally known as 'On-Premise CRM'), http://crm.dynamics.com/en-us/on-premises, accessed on December 2011 (2011).

TEMEP Discussion Papers

2010-42: Ruzana Davoyan, Jörn Altmann and Wolfgang Effelsberg, "Exploring the Effect of Traffic Differentiation on Interconnection Cost Sharing"

2010-43: Ruzana Davoyan and Jörn Altmann, "Investigating the Role of a Transmission Initiator in Private Peering Arrangements"

2010-44: Ruzana Davoyan, Jörn Altmann and Wolfgang Effelsberg, "A New Bilateral Arrangement between Interconnected Providers"

2010-45: Marcel Risch and Jörn Altmann, "Capacity Planning in Economic Grid Markets"

2010-46: Dang Minh Quan and Jörn Altmann, "Grid Business Models for Brokers Executing SLA-Based Workflows"

2010-47: Dang Minh Quan, Jörn Altmann and Laurence T. Yang, "Error Recovery for SLA-Based Workflows within the Business Grid"

2010-48: Jörn Altmann, Alireza Abbasi and Junseok Hwang, "Evaluating the Productivity of Researchers and their Communities: The RP-Index and the CP-Index"

2010-49: Jörn Altmann and Zelalem Berhanu Bedane, "A P2P File Sharing Network Topology Formation Algorithm Based on Social Network Information"

2010-50: Tai-Yoo Kim, Seunghyun Kim and Jongsu Lee, "The Gene of an Accelerating Industrial Society: Expansive Reproduction"

2010-51: Almas Heshmati and Sangchoon Lee, "The Relationship between Globalization, Economic Growth and Income Inequality"

2010-52: Flávio Lenz-Cesar and Almas Heshmati, "Agent-based Simulation of Cooperative Innovation"

2010-53: Erkhemchimeg Byambasuren and Almas Heshmati, "Economic Development in Mongolia"

2010-54: Almas Heshmati and Subal C. Kumbhakar, "Technical Change and Total Factor Productivity Growth: The Case of Chinese Provinces"

2010-55: Bory Seng and Almas Heshmati, "Digital Divide and Its Variations Amongst OECD, NIE and ASEAN Countries"

2010-56: Kibae Kim, Jörn Altmann and Junseok Hwang, "The Impact of the Subgroup Structure on the Evolution of Networks: An Economic Model of Network Evolution"

2010-57: Kibae Kim, Jörn Altmann and Junseok Hwang, "Measuring and Analyzing the Openness of the Web2.0 Service Network for Improving the Innovation Capacity of the Web2.0 System through Collective Intelligence"

2010-58: Alireza Abbasi and Jörn Altmann, “A Social Network System for Analyzing Publication Activities of Researchers”

2010-59: Jörn Altmann, Costas Courcoubetis and Marcel Risch, “A Marketplace and its Market Mechanism for Trading Commoditized Computing Resources”

2010-60: Fatmawati Zifa and Jörn Altmann, “A empirically validated framework for limiting free-riding in P2P network through the use of social network”

2010-61: Ashraf Bany Mohammed, Jörn Altmann and Junseok Hwang, “Cloud Computing Value Chains-Understanding Business and Value Creation in the Cloud”

2010-62: Ashraf Bany Mohammed and Jörn Altmann, “A Funding and Governing Model for Achieving Sustainable Growth of Computing e-Infrastructures”

2010-63: Junseok Hwang, Jihyoun Park and Jörn Altmann, “Two Risk-Aware Resource Brokering Strategies in Grid Computing: Broker-driven vs. User-driven Methods”

2010-64: Juthasit Rohitratana and Jörn Altmann, “Agent-Based Simulations of the Software Market under Different Pricing Schemes for Software-as-a-Service and Perpetual Software”

2010-65: Radoslaw R. Okulski and Almas Heshmati, “Time Series Analysis of Global Airline Passengers Transportation Industry”

2010-66: Alireza Abbasi and Jörn Altmann, “On the Correlation between Research Performance and Social Network Analysis Measures Applied to Research Collaboration Networks”

2010-67: Kibae Kim, Jörn Altmann and Junseok Hwang, “An Analysis of the Openness of the Web2.0 Service Network Based on Two Sets of Indices for Measuring the Impact of Service Ownership”

2010-68: Bory Seng, “The Driving Forces Underlying the Growth of Total Factor Productivity in Cambodia, Quantitative and Qualitative Studies”

2010-69: Michael Maurer, Vincent C. Emeakaroha, Ivona Brandic, and Jörn Altmann, “Cost and Benefit of the SLA Mapping Approach for Defining Standardized Goods in Cloud Computing Markets”

2010-70: Khin Swe Latt and Jörn Altmann, “A Cost-Benefit-Based Analytical Model for Finding the Optimal Offering of Software Services”

2010-71: Jung Eun Lee, Younghoon Kim, Yeonbae Kim and Donghyuk Choi, “The Impact of Technology Licensing Payment Mechanisms on Firms’ Innovative Performance”

2010-72: Dongook Choi and Yeonbae Kim, “Effects of Piracy and Digital Rights Management on the Online Music Market in Korea”

2011-73: Tai-Yoo Kim, Jihyoun Park, Eungdo Kim and Junseok Hwang, "The Faster-Accelerating Digital Economy"

2011-74: Ivan Breskovic, Michael Maurer, Vincent C. Emeakaroha, Ivona Brandic and Jörn Altmann, "Towards Autonomic Market Management in Cloud Computing Infrastructures"

2011-75: Kiran Rupakhetee and Almas Heshmati, "Rhetorics vs. Realities in Implementation of e-Government Master Plan in Nepal"

2011-76: Alireza Abbasi, Jörn Altmann and Liaquat Hossain, "Identifying the Effects of Co-Authorship Networks on the Performance of Scholars: A Correlation and Regression Analysis of Performance Measures and Social Network Analysis Measures"

2011-77: Ivona Brandic, Michael Maurer, Vincent C. Emeakaroha and Jörn Altmann, "Cost-Benefit Analysis of the SLA Mapping Approach for Defining Standardized Cloud Computing Goods"

2011-78: Kibae Kim and Jörn Altmann, "A Complex Network Analysis of the Weighted Graph of the Web2.0 Service Network"

2011-79: Jörn Altmann, Matthias Hovestadt and Odej Kao, "Business Support Service Platform for Providers in Open Cloud Computing Markets"

2011-80: Ivan Breskovic, Michael Maurer, Vincent C. Emeakaroha, Ivona Brandic and Jörn Altmann, "Achieving Market Liquidity through Autonomic Cloud Market Management"

2011-81: Tai-Yoo Kim, Mi-Ae Jung, Eungdo Kim and Eunnyeong Heo, "The Faster-Accelerating Growth of the Knowledge-Based Society"

2011-82: Mohammad Mahdi Kashef and Jörn Altmann, "A Cost Model for Hybrid Clouds"

2011-83: Daeho Lee, Jungwoo Shin and Junseok Hwang, "Application-Based Quality Assessment of Internet Access Service"

2011-84: Daeho Lee and Junseok Hwang, "The Effect of Network Neutrality on the Incentive to Discriminate, Invest and Innovate: A Literature Review"

2011-85: Romain Lestage and David Flacher, "Access Regulation and Welfare"

2011-86: Jörn Altmann, Martina Meschke and Ashraf Bany Mohammed, "A Classification Scheme for Characterizing Service Networks"

2012-87: Bory Seng, "The Introduction of ICT for Sustainable Development of the Tourism Industry in Cambodia"

2012-88: Juthasit Rohitratana and Jörn Altmann, "Impact of Pricing Schemes on a Market for Software-as-a-Service and Perpetual Software"