SaaS architecture and pricing models

Gabriella Laatikainen and Arto Ojala
Department of Computer Science and Information Systems
University of Jyväskylä
Jyväskylä, Finland
{gabriella.laatikainen, arto.k.ojala}@jyu.fi

***Abstract*— In the new era of computing, SaaS software with different architectural characteristics might be priced in different ways. Even though both pricing and architectural characteristics are responsible for the success of the offering; the relationship between architectural and pricing characteristics has not been studied before. The present study fills this gap by employing a multi-case research. The findings accentuate that flexible and well-designed architecture enables different pricing models; however, poorly designed architecture limits also the pricing. Scalability and high level of modularity are the major enablers of a great variety of pricing models. Using public cloud services may lead to introducing usage-based pricing or in the contrary, making the pricing simpler. Applying multi-tenancy lowers the customizability, consequently the customers' negotiation power decreases. Pricing may give special requirements to the architectural design, such as scalability, customizability and additional components.**

Keywords- cloud; pricing; SaaS; SaaS architecture

I. Introduction

Software-as-a-Service (SaaS) is both a delivery and business model defined by software architectural and business model characteristics. Recent literature describes SaaS as a multi-tenant, virtual, scalable and configurable application that is accessible through browser [1]–[4]. On the other hand, the SaaS model is understood as offered through a different revenue logic compared to the traditional licensed software, such as subscription-based and/or usage based pricing [2], [5]–[7].

The emerging diversity of SaaS pricing models [8] has an impact on the product development; including technical architecture and product design. While traditional software products might be priced at the final stage of the product development, pricing of SaaS has to be considered at the early design phase [9]. Consider the example of Google ad-financed pricing model. In this case, integration of advertisements into the software's front-end has to be designed early enough to allow monetarization. More common examples include incorporating usage measurement into the software because of the usage-dependent pricing units.

Besides the impact of pricing on the architecture, in some cases, the software architecture is responsible for limiting or enabling the use of different pricing model alternatives. For example, if the users' resource usage is difficult to estimate (e.g. the architecture does not have proper logging or there is no clear usage pattern), then the company might perform financially better with a usage-based pricing model, even though the sales department prefers fixed monthly fee (since customers wish to have an estimable budget). Thus, architecture and pricing are interrelated and the success depends also on the harmony between the software architecture and pricing model.

The interrelation of architectural and pricing characteristics has an impact also on the product development and management processes. To date, architectural decisions are usually made by technical staff of the company (architects and developers), while pricing related decisions belong to the responsibilities of business managers (product managers, product line managers, directors, sales managers, etc.) [9]. In many cases, these two units of the company do not interact with each other on daily basis; thus, the unsuitability of the software's pricing model and architecture might come to light too late causing avoidable losses. Hence, in cases when the software's architecture and its pricing are closely related, the knowledge of these interrelations is vital for both the technical lead and the business managers of the company.

Surprisingly, the connection between architectural and pricing characteristics of SaaS software has not been studied before. To fill this gap, the aim of this research is to understand the impact the architecture and pricing models have on each other. As a result of a multi-case study of 5 companies, we aim to answer the following research questions: (1) How does software architecture enable and limit pricing models? and (2) What is the impact of pricing on the software architecture?

The contribution of the research is two-fold. First, the research proposes a theoretical model that describes the relationship between software architecture and its pricing models. Secondly, the managerial implications provide insights into (1) what technical details are important in decision making about pricing and (2) what pricing aspects may have an impact on the architectural decisions.

The structure of this article is as follows. In the next section, we give an overview on recent work related to SaaS architecture and cloud pricing. In Section III, we describe the research methodology used in this article. In Section IV we present the findings of our research. We conclude our paper with a summary and discussion in Section V.

II. Literature Review

In this section, first the current literature on cloud computing and SaaS architecture is presented. Thereafter, recent work on cloud maturity models is summarized. Then, the parameters of cloud pricing models are described. At the end of the section, the findings and the motivation for this article are discussed.

A. Cloud computing and SaaS architecture

Cloud computing provides access to computing resources, storage space, and software applications via internet as a service. Cloud computing can be divided roughly into three service layers. These consist of (i) Infrastructure as a Service (IaaS), which provides computation and storage capacity, (ii) Platform as a Service (PaaS), which provides software development tools and an application execution environment, and (iii) Software as a Service (SaaS), which provides applications on top of PaaS and IaaS [10], [11].

The SaaS model evolved from Application Service Provisioning (ASP) in the late 1990s [4], [12]. ASP was developed as an alternative to on-premise software. It offered the possibility for clients to outsource the hosting and maintenance of the software to an ASP vendor [4]. The ASP model was based on a single-tenant architecture in which each customer had a customized version of the software in the ASP provider's server [13].

SaaS architecture is similar to service-oriented architecture [2], [13]. SaaS is a delivery model, software that is available through the network. Marston et al. reported virtualization (presenting an abstract, emulated computing platform to the users instead of the physical characteristics), multi-tenancy (a single instance of an application software serves multiple clients) and web service (communication over the HTTP protocol) as core architectural characteristics of SaaS software [1]. A well-defined SaaS architecture should be configurable (the application's appearance and behavior can be altered by the users), multi-tenant and scalable (maximized concurrency, effective use of application resources) [2].

As a key characteristic, multi-tenancy is a requirement for a SaaS vendor to be successful [14]. In a multitenant architecture, a single instance of common code and data is shared between multiple tenants [15]. Besides the requirements of shared hardware resources, shared application and shared database instance, Bezemer et al. requires also high degree of configurability in look-and-feel and workflow from multitenant software [16]. Some researchers consider also multi-instancy as a form of multi-tenancy [14], where vendors host separate instances for each customer within shared hardware [13], [14].

Multi-tenancy has many advantages. First of all, it improves the utilization rate of hardware resources and it eases the deployment and maintenance of the software. It also opens new data aggregation opportunities. These benefits result in lower maintenance costs that allow the provider to target also small and medium-size enterprises and thus to catch the "long tail" of the market. [2], [15], [16]

On the other hand, there are also disadvantages of multi-tenancy. Since the tenants share hardware resources, a problem caused by one of the tenants have an impact also on other tenants. Sharing the same database also increases the importance of scalability, security and zero-downtime requirements. Because of increased configurability and thus more complex code, the development work might require more efforts than in case of single-tenant application [16]. Multi-tenant architecture does not allow high customization, since customer-specific configurations can only be made at the meta-data layer [17].

B. SaaS software maturity

Some research groups suggest that a mature SaaS model can be achieved in an incremental way, and the maturity level of SaaS software depends on the level of SaaS architectural characteristics [2], [13] or architectural and business characteristics [18], [19]. Regarding the architectural properties, these maturity models consider multi-instance, customer-specific ASP architectures as the least cloud mature architectures and they call scalable, configurable, and multi-tenant-efficient applications as the most mature ones [2], [13]. However, different components can be at different maturity level. In the cloud maturity model of Kang et al., 16 different maturity levels are identified along Service Component and Maturity Level axes [19]. Besides the Data, System and Service components, the Business characteristics are also taken into account in this model. In Forrester's maturity model, 6 levels are identified that considers also the firm strategy as a key factor in cloud maturity. In this model, outsourcing resides at the lowest level, while firms at the most mature level offer dynamic business applications as a service [18].

Other research groups focus on classifying the SaaS providers into different business archetypes, such as "pure-SaaS" and "enterprise-SaaS" [20], [21]. Pure SaaS refers to software that is simple to use and has low or no requirements for customization [22]. According to Benlian et al., pure-SaaS products also have lower strategic significance in a customer's business processes compared to enterprise-SaaS products [20]. In addition, pure-SaaS products, such as office systems, may have lower inimitability [20]. Enterprise-SaaS, on the other hand, refers to software which is more complex and which may require support, involving integration with customers' existing IT systems [22]. According to Benlian et al. enterprise-SaaS, such as ERP systems, has high strategic value for customers, and inimitability is high [20]. They also found that the adoption of enterprise-SaaS has a high level of uncertainty. As a third group, Luoma et al. introduced the notion of a self-service SaaS archetype, which presents highly standardized applications with easy adoption [22]. In self-service SaaS, customers themselves find applications from the Internet, and evaluate and deploy the software. These applications are mainly targeted at individual consumers [22]. Berman et al. reveal three business archetypes representing the extent to which organizations use cloud computing: "Optimizers" use the technology to enhance the value proposition and improve efficiency, "Innovators" create new streams of revenues or even change their role in the value network, and "Disruptors" may generate totally new customer needs and segments, possibly even new value chains [23].

As a summary, SaaS application cannot be classified in a discrete number of maturity types. The researchers accentuate that targeting the highest maturity level is not necessarily the best fit for every vendor. Software vendors should decide the service components that are shared across the customers and also the level at which these components are shared. Decision makers should take into account many factors, such as the business needs, the targeted customers, architectural characteristics, financial and operational considerations. In some cases, entering a higher level of SaaS maturity is not possible because of confidentiality and security aspects, or since customers may have legal or cultural resistance to multi-tenancy. Some applications can't be moved because the migration is not beneficial cost-wise or the nature of the product/ service requires isolated data and code. In some cases, it may be difficult to guarantee the SLA obligations (e.g. downtime, support options, disaster recovery).

C. *Pricing models*

SaaS software may be priced in many different ways. Even though one of the key conditions for commercial success of cloud services is the clearness and transparency of pricing for both customers and providers [3], [24], SaaS price models are very diverse and complex [8]. In software industry the most common revenue streams are: i) monthly or annual subscription fees, ii) advertising based revenue, iii) transaction based revenue (customers are charged based on the number of transactions they perform), iv) premium based revenue (revenue is generated from charging for premium versions besides the free versions), v) revenue from implementation and maintenance services and vi) software licensing [25]–[28].

Software pricing in these above introduced revenue models may base on different aspects. The software pricing model parameters of Lehmann and Buxmann [29] and the SBIFT model of Iveroth et al. [30] are taken into account in the classification of cloud pricing models that describes these models along 7 dimensions [31]:

1. *Scope* represents the granularity of the offering, whether it is priced as a package or different prices are given for different functionalities.
2. *Base* represents the information base the price is set on. The price might be decided based on cost considerations, the competitors' prices, based on performance or customer value.
3. *Influence* represents the ability of buyers and sellers to influence the price, and it contains the options Pricelist, Negotiation, Result-based price, Pay-what-you-want, Auction and Exogenous pricing.
4. *Formula* represents the connection between price and volume, and it contains different variations of fix and variable price components.
5. *Temporal rights* represent the length of service's usage period, and it can be Perpetual, Subscription-based or Pay-per-use.
6. *Degree of discrimination* represents the level of price variety depending on the buyer. The product or service may be priced differently for different regions, for different time of buying. The price can depend on the acquired volume or the quality, or it might be even customer-specific.
7. *Dynamic pricing strategy* represents the strategy of dynamic price change over time. Penetration, skimming or hybrid pricing strategies belong to this dimension.

D. *Summary*

In summary, well-designed SaaS architecture requires virtualization, multi-tenancy, web service, scalability and configurability. Cloud pricing characteristics include scope, base, influence, formula, temporal rights, degree of discrimination and dynamic pricing strategies. Choosing the right maturity level requires architectural, business and operational considerations. Even though researchers paid increasing attention to both SaaS architecture and pricing models, there is no research paper focusing on the relationship between software architecture and pricing models.

III. METHODOLOGY

The aim of the research is to enable an in-depth investigation of a complex phenomenon in a real-life environment, where architectural and pricing decisions are made. Since the study aimed to understand the behavior of a firm rather than quantitative measurement, the case study method is suitable for this purpose [32], [33]. The research setting for the study consists of five software firms marked with A-E (see TABLE I). In order to gain a deep understanding on the phenomena, the following multiple criteria is used to select the cases: (i) the case firms develop software for different industries; (ii) the sample includes both recently established and relatively old firms; (iii) the sample includes both traditional software firms and SaaS companies; (iv) the sample includes SaaS with different cloud maturity level; (v) researchers have good access to the required information, as recommended by [34].

TABLE I
OVERVIEW OF THE CASE FIRMS

Firm	**Year of establishment**	**Number of employees**	**Target industry**
A	1997	cca. 980	Finance, public sector, telecom and other industries
B	1998	30	Telecom operators, Component manufacturers and service providers for telecom networks
C	2011	3	Public and private sector
D	2008	12	Large and medium sized corporations
E	2006	30	Furniture chains and furniture manufacturers

Multiple sources are used to gather data on each case firm. The data is collected primarily through semi-structured interviews with multiple decision makers of the case companies. In TABLE II, the number of interviews is presented with representatives of the case firms. The interviewees consist of Chief Executive Officers, vice presidents, sales managers, architects, technical leads and project managers. The interviews last cca. 60 minutes and they are all recorded and transcribed. Thereafter, the complete transcripts are sent back to the interviewees for review. Some of them commented on the content while other interviewees accepted the transcripts as they were. In addition to face-to-face meetings, information is gathered through phone calls and emails. Besides these, secondary information is gathered about the cases through web pages, brochures and press releases.

TABLE II
NUMBER OF INTERVIEWS

Firm	A	B	C	D	E
Nr. of interviews	1	8	4	6	8

We utilized content analysis as the data analysis method. The case data analysis consisted of three concurrent flows of activity [35]: (i) data reduction, (ii) data displays and (iii) conclusion-drawing/verification. In (i) data reduction phase, the data were given focus and simplified through compilation of a detailed case history of each firm. This is in line with Pettigrew [36], who suggests that organizing incoherent aspects in chronological order is an important step in understanding the casual links between events. Thereafter, on the basis of interviews and other material collected from the case firms, we used tables to identify and categorize the unique patterns of each case under subtopics derived from the research questions. In addition, we used checklists and event listings to identify critical factors related to the phenomena encountered [35]. In (ii) the data display phase, we arranged the relevant data drawn from the findings of the earlier phase into new tables. In (iii) the conclusion drawing and verification phase first we concentrated on identifying the aspects that appeared to have significance for this study. At this stage we noticed regularities, patterns, explanations and causalities related to the phenomena. After conclusion drawing, we verified the results and carried out discussions in order to avoid misunderstandings. In the last stage of the research, we sent the manuscript to the representatives of each firm for review.

IV. FINDINGS

A. *Overview of the case firms and relationship between architecture and pricing*

In TABLE III, we provide a short overview of the case firms regarding the software architecture and pricing model of their main product or service. It can be seen from the table, that the companies are very different in terms of cloud maturity. However, subscription-based pricing is a common pricing model that is applied by each case firm.

TABLE III
ARCHITECTURE AND PRICING MODEL - OVERVIEW

Firm	Product/Service	Architecture	Pricing model
A	Development tool and Backend As A Service	Traditional software and PaaS. Use of public cloud services.	Premium pricing model (traditional software) and subscriptions with usage based pricing (PaaS)
B	Planning and optimization software for telecom operators	ASP architecture by design. No public cloud provider is used. Virtualization, scalability, high level of modularity.	Offered (i) traditionally, (ii) through subscriptions and (iii) as part of consultancy services
C	Software for user guides, commercials and media description	mobile web application, ASP architecture	One-time fixed fee and monthly subscription fee
D	Entitlement management software	SaaS, Service Oriented Architecture	Offered both with traditional licenses and subscriptions
E	Interactive 3D sales software	Designed as an ASP software. Migration to SaaS architecture is in progress.	One-time fixed fee, monthly subscription fee and usage-dependent hosting fee

In TABLE IV, the relationship between architecture and pricing is presented. The first two columns refer to the first research question regarding the impact of architecture on pricing models. In the last column, the effect of pricing on the architecture is presented. It can be seen from the table that the relationship between architecture and pricing varies case by case.

TABLE IV
RELATIONSHIP BETWEEN ARCHITECTURE AND PRICING

Firm	Architecture enables pricing models	Architecture limits pricing models	Pricing affects architecture
A	Flexible and configurable architecture enables different pricing models. Use of public cloud provider's services implies introducing usage-based pricing.	Traditional desktop application can't be migrated to SaaS and offered through subscription.	Fixed priced projects lead to poorer design. Pricing requires extra components. Creating premium functionalities on top of open source software implies use of specific technologies.
B	Scalable and highly modularized architecture enables different pricing models (subscription-based model, licenses, different bundling options, usage-based pricing).	Well-designed architecture does not limit the pricing.	Pricing requires scalable and customizable architecture, use of public cloud resources, different delivery modes and additional components.

C	Loose connection between architecture and pricing (startup firm)	Loose connection between architecture and pricing (startup)	Pricing will entail automatic billing tools and other configuration tools.
D	Loose connection between architecture and pricing (small firm focusing only on software development).	The architecture does not limit pricing, but gives the criteria on how to price.	Pricing does not affect architecture (small firm focusing only on software development).
E	Migration to SaaS architecture and use of public cloud services implies fixed price instead of usage-based pricing. Change in architecture enables change in pricing. Introducing SaaS architecture lowers the prices.	Multi-tenancy lowers the customizability, the product/service becomes more standardized; therefore the firm is less willing to negotiate with customers.	Usage-based pricing requires functionalities that enable pricing. Prices have to be lowered, therefore maintenance costs have to be decreased; thus, introducing SaaS architecture is needed.

Architecture and pricing are closely related in case of firms A and E, where the firms migrate their software to public cloud resources. For the case firm A, introducing a new PaaS service with usage based pricing model is under progress. Likewise, the case firm E currently migrates its product to SaaS architecture, and as a consequence, the pricing model becomes simpler and prices will be lowered. Representatives of case firms A and E affirmed that both architecture and pricing characteristics are considered in decisions related to pricing and technical details. Pricing and architecture are interrelated also at the firm B, where well-designed architecture enables many different pricing models. In return, these pricing models require high quality software. However, it is difficult to say, has the architecture impact on pricing or the other way around:

"Architecture affects pricing, pricing affects architecture … it is the egg-chicken problem … architecture and pricing are in symbiosis."

Conversely, the relationship is loose in case of firms C and D. Firm C is a startup company of 3 employees, where neither the architecture nor the pricing is yet mature; currently the firm's main goal is to attract new customers and get references. The simplicity of both the architecture and pricing model does not allow close relationship between architecture and pricing. The firm concentrates on short-term goals like working software and appropriate pricing for the customers, where architecture and pricing are independent from each other.

Likewise, firm D is a small company consisting of only 8 employees. The firm's main activity is software development, while the channel partners are responsible for end customer service, helpdesk support and pricing. Firm D is not involved in pricing issues regarding the end customers; thus, architecture and pricing do not interact in case of their software. One of the interviewees representing firm D stated:

"Pricing has no impact on architecture. The architecture does not limit the pricing, but it gives the criteria to find the right price level. If I would give the price level first, and then the development costs should match this level, it wouldn't be good."

B. How does software architecture enable and limit pricing models?

The interviewees all agreed that flexible and well-designed architecture enables different pricing models; however, poorly designed architecture limits also the pricing. A representative of company A accentuates the flexibility and configurability as key architectural characteristics that enable different pricing models.

"In this case architecture does not limit pricing. It is a very flexible architecture that enables configurability.[…] Having a cloud offering enables us to have monthly fees. Since the customer is closer to us, we know what he does and we are able to develop our service much more. Additional sales becomes easier."

Scalability and high level of modularity are the most important characteristics for company B that let them offer the same software with different pricing models.

"Architecture does not limit pricing, but enables it. Scalability and modularity enables many [pricing] possibilities."

"Architecture enables different models, such as SaaS and service packages, even licenses, everything is possible. If something happens and we can't estimate the users' resource needs then it would probably affect the pricing. Then probably we would apply usage-based pricing."

Using public cloud services may lead both to introducing usage-based pricing or getting rid of it and make the pricing simpler. In the first case, company A introduces the SaaS service as a new service and expects the customers to pay for the public resources based on their usage. However, company E sees the cloud providers' usage log data as an advantage since it enables the firm to simplify its pricing. Customers prefer simple pricing where they know the fee in advance. With the excessive logging and monitoring data, the company can estimate the customers' usage more easily.

However, software architecture limits pricing models as well. Company A offers its software as a desktop application that has to be installed on customers' machine. The license-based software is sold with a yearly fee per user. In addition to this software, the company wants to introduce a new cloud based service with subscription-based pricing model. Besides the monthly fees, the customers will pay for the used resources based on their usage.

A representative of company E mentioned that introducing SaaS architecture and multi-tenancy lowers the negotiation power of the customers. Customization is more challenging in a multi-tenant architecture and with the architectural change the strategy of the company moves from creating customized software towards offering standard software with configuration possibilities. This drawback of multi-tenancy has to be accepted for smaller maintenance costs in turn.

"This new architecture makes customization much more limited. At the moment we have different instances for different customers, so we can customize it much more. It is

challenging to decide where is the border line where we move, what kind of customizations we will be able to do."

In the future, the company wants customers to configure and maintain different part of the software themselves. The new architecture allows customers to configure the user interface and the business logic rules; they can create new users, new elements and maintain the old ones through a simple web interface. The goal of the company is to restrict the cost of customer-based maintenance work to zero. However, the low level of customers' influence on setting the price limits the pricing. As a result, the company's value proposition is communicated through a pricelist.

C. What is the impact of pricing on the SaaS architecture?

The representative of firm E stated that the need for great architectural changes comes from the sales department of the company. The architectural change is needed for two reasons. First of all, prices and therefore maintenance costs have to be lowered. The customers' higher expectations, the need of lower prices and the high competition place new demands on software architecture and development.

"If we ask from the sales persons, they want to lower the price in order to get more new customers. On the technology side the maintenance costs should be lowered. But the final pricing decisions come from the marketing and sales. [...] So yes, the pricing has an impact on the architecture: the starting price is smaller now, the maintenance and other costs have to be lowered as much as possible. Usually we have technical and sales persons and we think together about pricing."

Secondly, the company wants to extend its services to another large market with different pricing needs than the countries where the company operates now. In this new country customers want short contracts with no fix fee in the beginning. This requires architecture where new customers can benefit from the services without long installation and integration costs.

"Here everybody understands that the initial project work has to be paid. [...] But in this new market customers are used to have only monthly fees, no long contracts. [...] The starting fee is commitment that makes the sales work more difficult and slow. [...]There we can sell our software with monthly fee only, no fixed starting fee."

Although the cost of the planned architectural changes is very high, it is seen as a required investment that has long-term benefit for the company and the only possible way that allows the company to grow.

Pricing may give special requirements to the architectural design. In case of company B scalability, high customizability and the use of public cloud providers' resources were the most important requirements from the sales department towards the technical team. In this company the software was priced already in the requirements specification phase of the development life cycle; therefore the requirements for pricing were taken into account in the architectural design. This finding is in line with Choudhary's finding: in most of the cases the subscription-based licensing leads to higher architectural requirements and greater investment in product development and that implies higher software quality [5].

In case of companies A and C, change in pricing model may require additional components, such as different infrastructure, automatic billing or configuration tools. Thus, the technical team has to be consulted before implementing changes in pricing model to give work estimation.

Pricing models have an impact also on work prioritization. Interviewees of companies A and E discussed that in case usage-based pricing is introduced, first those functionalities have to be developed that enable usage-based pricing (company E) and those that generate more resource usage (company A).

V. CONCLUSIONS

SaaS is a multi-tenant, virtual, scalable and configurable web application [1]–[4] that is offered through subscription-based and/or usage based pricing [2], [5]–[7]. The interrelation of architectural and pricing characteristics implies that basic knowledge on the offering's architecture is required for pricing decisions and pricing also affects the architecture. In this paper we presented the results of our multi-case study on the relationship between architecture and pricing.

Findings in this study reveal that architecture and pricing are tightly related in cases where the firm's value proposition resides at high cloud maturity level and hosting is outsourced to a public cloud provider. In this case architecture is an important factor in pricing, while pricing gives special architectural requirements. However, in case of startup firms, or smaller companies that focus only on software development, architecture and pricing have no or limited impact on each other.

Concerning the first research question, we found that flexible and well-designed architecture enables different pricing models; however, poorly designed architecture limits also the pricing. Scalability and high level of modularity are important characteristics that enable a great variety of pricing models. The decision of using public cloud services requires redesigning the pricing model as well. In some cases, the use of public resources leads to introducing usage-based pricing, while in other cases the pricing model becomes even simpler. Introducing multi-tenancy lowers the customizability of the software; thus also the negotiation power of the customers decreases.

The relationship between SaaS architectural and pricing characteristics is visible in Figure 1. In the figure, only those architectural and pricing characteristics are visible that have an impact on each other based on the findings. For example, the Base, Degree of Discrimination and Dynamic Pricing Strategy pricing characteristics are not affected directly by the architectural decisions; thus, these dimensions might be more influenced by the strategic decisions of the company and do not depend on the software architecture. In the figure, solid arrows represent enabling relationship between architectural and pricing characteristics, while dashed arrows are used to show the limiting relationship.

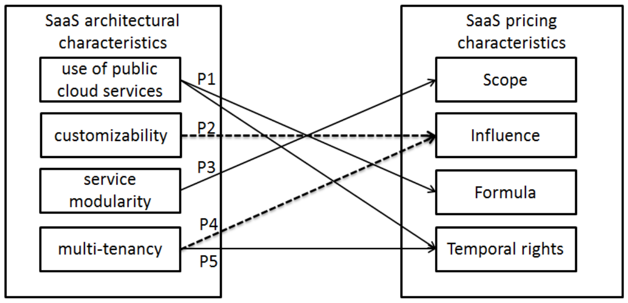

Figure 1. The impact of architecture on pricing models.

The relationships marked in the picture with P1-P5 can be described as follows:

P1. *Using public cloud services has an impact on both the Temporal rights and Formula dimensions.* For example in case of company A, introducing a new service that uses public cloud resources implies use of subscription-based pricing model (Temporal rights dimension) with usage-dependent pricing metrics (Formula dimension). Conversely, migration to public cloud at the company E makes the pricing simpler and the pricing model with usage-dependent pricing metrics is replaced with subscription-based revenue logic (Temporal rights and Formula dimensions).

P2. *The level of customizability has an impact on the possible influence of the customers on the pricing model (Influence dimension).* As an example, migration to a multi-tenant architecture lowered the level of customizability for case company E. This affects the pricing as well: standardization of the product/service leads to less negotiation.

P3. *High level of service modularization enables many different bundling options (Scope dimension).* For example, the company B's software is a scalable and highly modularized application that is offered through three different revenue model: traditionally, through subscription and as part of consultancy services.

P4. *Multi-tenancy limits the customizability level that may limit the negotiation power of the customer (Influence dimension).* For example in case of company E, the lower customizability implies less negotiation between the customers and the provider.

P5. *High level of multi-tenancy enables different options for the Temporal rights dimension of the pricing model.* For example, migration to multi-tenant architecture at the company E enables subscription-based revenue logic.

Concerning the second research question, pricing related decisions should be communicated early enough in the software development life cycle since pricing may give special requirements to the architectural design, such as scalability, high customizability and the use of public cloud providers' resources. This finding is in line with Choudhary's finding: in most of the cases the subscription-based licensing model leads to higher architectural requirements, thus, to greater investment in product development and that implies higher software quality [5]. Additionally, the findings reveal that if the pricing model changes, additional components may be required, such as different infrastructure, automatic billing or configuration tools. The pricing model has also an impact on the work prioritization.

The present study contributes also to the literature on understanding the business model changes due to migration from on premise to cloud services. We found that the use of cloud resources has an impact on the pricing model; namely, due to migration, the prices might be lowered and the pricing model might become simpler, or, usage-based pricing components might be introduced. Even though migration to cloud requires great investments, companies see this path the only way to survive under heavy competition in the market.

Due to the methodological circumstances, the findings of this study cannot be fully generalized. However, they can be used for further quantitative testing. Besides, the impact of pricing on the internal processes of the companies is also left for further studies.

References

[1] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, "Cloud computing—The business perspective," Decis. Support Syst., vol. 51, no. 1, pp. 176–189, 2011.

[2] F. Chong and G. Carraro, "Architecture strategies for catching the long tail," MSDN Libr. Microsoft Corp., pp. 9–10, 2006.

[3] C. Weinhardt, D.-I.-W. A. Anandasivam, B. Blau, D.-I. N. Borissov, D.-M. T. Meinl, D.-I.-W. W. Michalk, and J. Stößer, "Cloud computing–a classification, business models, and research directions," Bus. Inf. Syst. Eng., vol. 1, no. 5, pp. 391–399, 2009.

[4] A. Benlian and T. Hess, "Opportunities and risks of software-as-a-service: Findings from a survey of IT executives," Decis. Support Syst., vol. 52, no. 1, pp. 232–246, 2011.

[5] V. Choudhary, "Comparison of software quality under perpetual licensing and software as a service," J. Manag. Inf. Syst., vol. 24, no. 2, pp. 141–165, 2007.

[6] S. Stuckenberg, E. Fielt, and T. Loser, "The impact of software-as-a-service on business models of leading software vendors: experiences from three exploratory case studies," in Proceedings of the 15th Pacific Asia Conference on Information Systems (PACIS 2011), 2011.

[7] P. Tyrväinen and J. Selin, "How to sell SaaS: a model for main factors of marketing and selling software-as-a-service," in Software Business, Springer, 2011, pp. 2–16.

[8] M. A. Cusumano, "The changing labyrinth of software pricing," Commun. ACM, vol. 50, no. 7, pp. 19–22, 2007.

[9] H. B. Kittlaus and P. N. Clough, Software product management and pricing: Key success factors for software organizations. Springer, 2009.

[10] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "A view of cloud computing," Commun. ACM, vol. 53, no. 4, pp. 50–58, 2010.

[11] M. H. Hugos and D. Hulitzky, Business in the cloud: what every business needs to know about cloud computing. Wiley. com, 2010.

[12] M. Xin and N. Levina, "Software-as-a-service model: Elaborating client-side adoption factors," in Proceedings of

the 29th International Conference on Information Systems, R. Boland, M. Limayem, B. Pentland,(eds), Paris, France, 2008.

[13] J. Ju, Y. Wang, J. Fu, J. Wu, and Z. Lin, “Research on key technology in SaaS,” in Intelligent Computing and Cognitive Informatics (ICICCI), 2010 International Conference on, 2010, pp. 384–387.

[14] C. J. Guo, W. Sun, Y. Huang, Z. H. Wang, and B. Gao, “A framework for native multi-tenancy application development and management,” in E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007. The 9th IEEE International Conference on, 2007, pp. 551–558.

[15] C.-P. Bezemer and A. Zaidman, “Multi-tenant SaaS applications: maintenance dream or nightmare?,” in Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), 2010, pp. 88–92.

[16] C.-P. Bezemer, A. Zaidman, B. Platzbeecker, T. Hurkmans, and A. t Hart, “Enabling multi-tenancy: An industrial experience report,” in Software Maintenance (ICSM), 2010 IEEE International Conference on, 2010, pp. 1–8.

[17] M. Cusumano, “Cloud computing and SaaS as new computing platforms,” Commun. ACM, vol. 53, no. 4, pp. 27–29, 2010.

[18] S. Ried, Forrester’s SaaS maturity model: Transforming vendor strategy while managing customer expectations (2008). .

[19] S. Kang, J. Myung, J. Yeon, S. Ha, T. Cho, J. Chung, and S. Lee, “A general maturity model and reference architecture for saas service,” in Database Systems for Advanced Applications, 2010, pp. 337–346.

[20] A. Benlian, T. Hess, and P. Buxmann, “Drivers of SaaS-adoption–an empirical study of different application types,” Bus. Inf. Syst. Eng., vol. 1, no. 5, pp. 357–369, 2009.

[21] K.-W. Huang and M. Wang, “Firm-Level Productivity Analysis for Software as a Service Companies,” ICIS 2009 Proceedings, 2009.

[22] E. Luoma, M. Rönkkö, and P. Tyrväinen, “Current Software-as-a-Service Business Models: Evidence from Finland,” in Software Business, Springer, 2012, pp. 181–194.

[23] S. J. Berman, L. Kesterson-Townes, A. Marshall, and R. Srivathsa, “How cloud computing enables process and business model innovation,” Strategy Leadersh., vol. 40, no. 4, pp. 27–35, 2012.

[24] A. Anandasivam and M. Premm, “Bid price control and dynamic pricing in clouds,” ECIS 2009 Proceedings, 2009.

[25] A. D’souza, J. Kabbedijk, D. Seo, S. Jansen, and S. Brinkkemper, “Software-As-A-Service: Implications For Business And Technology In Product Software Companies,” PACIS 2012 Proceedings, 2012.

[26] M. A. Cusumano, “The changing software business: Moving from products to services,” Computer, vol. 41, no. 1, pp. 20–27, 2008.

[27] A. Ojala, “Software renting in the era of cloud computing,” in Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on, 2012, pp. 662–669.

[28] A. Ojala, “Software-as-a-Service Revenue Models,” IT Professional , vol.15, no.3, pp.54,59, 2013.

[29] S. Lehmann and P. Buxmann, “Pricing Strategies of Software Vendors,” Bus. Inf. Syst. Eng., vol. 1, no. 6, pp. 452–462, 2009.

[30] E. Iveroth, A. Westelius, C.-J. Petri, N.-G. Olve, M. Cöster, and F. Nilsson, “How to differentiate by price: Proposal for a five-dimensional model,” Eur. Manag. J., 2012.

[31] G. Laatikainen, A. Ojala, and O. Mazhelis, “Cloud Services Pricing Models,” in Software Business. From Physical Products to Software Services and Solutions, Springer, 2013, pp. 117–129.

[32] R. K. Yin, Case study research: Design and methods, vol. 5. Sage, 2003.

[33] G. Paré, “Investigating information systems with positivist case research,” Commun. Assoc. Inf. Syst., vol. 13, no. 1, p. 18, 2004.

[34] R. E. Stake, “The art of case study research,” Sage, 1995.

[35] M. B. Miles and A. M. Huberman, Qualitative data analysis: An expanded sourcebook. Sage, 1994.

[36] A. M. Pettigrew, “Longitudinal field research on change: theory and practice,” Organ. Sci., vol. 1, no. 3, pp. 267–292, 1990.