Bridging the gap between software architecture and business model development: A literature study

Sami Hyrynsalmi
Computing Sciences
Tampere University
Pori, Finland
sami.hyrynsalmi@tuni.fi

Sampsa Rauti
Department of Future Technologies
University of Turku
Turku, Finland
sampsa.rauti@utu.fi

Erkki Kaila
Department of Future Technologies
University of Turku
Turku, Finland
erkki.kaila@utu.fi

***Abstract*—The software architecture plan describes the high-level structure and logic of a software system. The architectural plan acts as a constitution and dictates the fundamental principles of the system; therefore, the plan also eventually determines which kinds of business models the software system can support. In the modern mercury business, there is need for experimentation in business model and flexibility in architecture. This paper uses a systematic literature review method to collect primary studies from the extant literature addressing business models and software architectures. The aim is to summarize the current knowledge. The selected primary studies (n=10) are qualitatively analyses and synthesized. The results show that the area remain mostly unaddressed and there is need to develop new methods to support flexible architecture design, tools and development methods.**

***Index Terms*—software architecture; business model; mercury business; literature review; software-intensive business**

I. Introduction

In the modern industry, the speed of changes has been drastically increased [1]. This is even more evident in the software-intensive businesses [2, 3], where ability to swiftly react to changes and adopt new innovations can be a crucial competitive advantage. For example, Järvinen et al. [4, p. 65] describe the new phenomenon as a 'mercury business' where software-intensive companies are actively looking to find *"new grooves where to flow to grow new business"*. As pointed out by Järvinen, et al. [4], for a company to be able to work in mercury business, changes are requested into organization, leadership and culture of the company.

In addition to the organizational issues and capabilities in the software development processes, mercury business also requires that *software architecture* of a product or a service needs to support incremental development and changes in the business model of a product [4]. The software architecture description of a software system denotes the elements that constitute the software system, their properties as well as the relationship between the elements [5]. Thus, the software architecture, in the end, also dictates which kind of a *business model* can be implemented with the software system.

While there are plethora of studies focusing on how an architecture design should meet the (fixed) business goals [c.f. 6], to the authors' best knowledge, a more limited research stream has focused on discussing how an architectural design can support business model experimentation. To map the existing academic knowledge, this paper presents a systematic literature review on works addressing relationships between business models and software architectures. More specifically, we address the following research questions:

RQ1 What extant literature report on the relationship between software architecture and business model?

RQ2 How software architecture design, according to the extant literature, can support business model experimentation?

To answer the presented questions, we performed a literature study. The primary studies were collected with systematic literary study method [7] and addressed via thematic analysis. The final set of primary studies consists of 10 articles published between 2001 and 2017. We restrict our focus specifically on software architectures and we do not included, e.g., system architecture studies.

The remaining of this paper is structured as follows. Section II reviews literature and defines the central concepts for this study. It is followed by the description of used research process in Section III and the results in Section IV. Finally, Section V presents the analysis and discussion of the results while Section VI concludes the study.

II. Background

A. Software architecture

Software architecture, which describes high-level structure and logic of a software system, has several different definitions in the existing literature. According to the IEEE Standards Association, a software architecture is *"The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution."* [8] Therefore, an architecture is not only about the static structure, but it also includes the functionalities and the dynamic structures of the system. It describes how the parts of the system interact with each other. The software architecture is also a guideline for the developers: it describes rules and principles regarding how to develop a system using the given architecture and imposes some restrictions on the future evolution of the system.

Another way to look at software architectures is to note that the process of designing an architecture always involves making several important design decisions. Following this reasoning, Bosch considers the software architecture as a set of major design decisions [9]. Because the architecture aims to fulfill the quality requirements for the system, the most pivotal design decisions are usually quality-related. A good architectural description should explain how the architecture meets the technical and the business expectations set for the system through the design decisions. Justifications for the major decisions should also be included.

For different purposes, the architecture can be described on several levels of detail and from different points of view. Architectural models and diagrams can either show how the system has been divided into parts (static description) or how it behaves during execution (dynamic description). Different points of view in architectural description often serve specific stakeholder groups. For example, in order to implement the system, a developer wants to see how the system has been organized into software modules (logical view). On the other hand, a system engineer might want to see what the topology of the component on the physical layer of the system looks like (physical view). [10] This way, a well-documented software architecture also facilitates communication between stakeholders [11].

Software architectures often make use of architectural styles. An architectural style refers to *"a general, reusable solution to a commonly occurring problem in software architecture within a given context"* [12]. The concept is very similar to the idea of design patterns, but architectural styles have broader coverage in the architecture. An architectural style defines and delineates some essential elements and principles in a given software architecture, guiding the implementation process and keeping the developers from exercising too much unnecessary creativity. For example, in the layered architecture style, components within a system are organized into horizontal layers, and each layer has a specific role within the system. A layer forms an abstraction around its functionality, so a layer does not need to be concerned with the internal implementation of any other layer [13].

B. Business model

The literature of business models spring from the field of strategic management. To define the concept simply, a business model is a blueprint of a business organization. That is, the business model aim to illustrate how a company operates and makes money [14].

More formally, a business model can be defined as *"[...] the rationale of how an organization creates, delivers and captures value"* [15]. That is, the business model of a company should describes the ways how a company produces value (e.g., what benefit the product or the service brings to the customer or what problem it solves); how the offered solution is delivered and distributed to the customers; and how the company makes profitable business out of the offered goods or services.

While there are dozens of different business modelling notations and tools presented [14, 16], the Business Model Canvas by Osterwalder and Pigneur [15] is the most often used. The canvas consists of nine building blocks: *(a)* Key partners, *(b)* Customer segments, *(c)* Value propositions, *(d)* Channels, *(e)* Customer relationships, *(f)* Revenue streams, *(g)* Key resources, *(h)* Key activities, and *(i)* Cost structure.

Business models have been reduced to, e.g., a set of actors, activities, and interfaces [17]. Furthermore, often seen simplification equates the term 'business model' with, e.g., the revenue or pricing model of a product [c.f. 18]. In the terminology used in this paper, the revenue model of a company consists of one or more revenue streams [16]. A revenue stream defines how the company is getting compensated from the services or products offered [19].

As the terminology related to the business models is not established, also the concept of 'pricing model' is sometimes used to depict similar or partly overlapping concepts [c.f. 20]. In general, the pricing model defines how the service or a product is priced (eg., what components are included into pricing, whether payment flow is frequent or single). As in the case of SaaS domain, often studies have been addressing pricing models enabled by the architecture [cf. 21].

C. Related work

To the best of authors' knowledge, no previous literature study has been presented in this area. However, there are a few studies addressing the gap. For example, the book by Luke Hohmann [22], titled *Beyond Software Architecture*, divided a software system into technical (i.e., *tarchitecture*) and market architecture (i.e., *marketecture*). According to Hohmann [22, p. 51], marketecture *"is the business perspective of the system's architecture. It embodies the complete business model, including the licensing and selling models, value propositions[...]"*.

Furthermore, Arsanjani [23] argued the need to bridge the gap between business modelling and software architectures already in 2001. Losavio et al. [24], for example, have argued that architecture of a system needs to be agile enough to support changes in business model.

Finally, Laatikainen and Ojala [25] as well as Laatikainen [21] have addressed the relationship between software architecture and pricing models in SaaS solutions. Furthermore, they noted the lack of work addressing this area. As put by Laatikainen [21, p. 13], *"[a]lthough the relationship between a software architecture and its pricing model is important to understand, this topic has not yet been investigated in the literature."* This study is motivated by their observation, yet the scope is extended from pricing models to business models.

III. Research Process

In order to map the current academic knowledge on the phenomenon at hand, we decided to use systematic literature study to collect the material. The systematic literature study is, according Kitchenham and Charters [7], a method to systematically and in a replicable way collect primary studies from the extant literature. SLRs are used to summarize existing evidence, pointing gaps of knowledge and provide a framework and position for future research activities [7].

The systematic literature studies can be categorized into two groups [26]: On the one hand, Systematic literature reviews aim to collect all available evidence to answer to a specific research question. Systematic mapping studies, on the other hand, aim to offer a broad review of evidence that is available on the specific topic. As this study's aim is to point out current knowledge of bridging the gap between the software architecture and the business model research streams, this study uses the latter approach. Furthermore, as this study's aim is to broadly map the current knowledge, we decided to use electronic searches on large publication databases instead of manual searching form the selected venues.

In this study, we follow the guidelines given by Kitchenham and Charters [7] for systematic studies in software engineering. We designed the research process which accommodated the following six phases:

1. **Prestudy.** In this phase, we performed initial searches as well as selected from the authors' previous experience a set of papers that are discussing about the phenomenon on the hand. The prestudy phase showed that there should be enough primary studies for the literature review. In addition, the prestudy phase produced a set of papers that are used calibrate the search term. The selected set consisted of three peer-review papers.
2. **Search term formation.** In this phase, we constructed the search term to be used in the searches. Based on experimenting with different keyword and search string combinations in order to find maximum number of related works (including the pre-selected publications that should be found with the searches), we ended up using the following Boolean search term:

   ```
   ("software architecture" OR "SaaS
   architecture") AND ("revenue model"
   OR "pricing model" OR "business
   model")
   ```

   That is, the left-hand side of the search term aims to include the software architecture research papers. We selected to use both 'software architecture' and 'SaaS architecture' terms as in the SaaS literature stream, this term is often used to depict the architectural decision instead of the base concept. The right-hand side of the search term similarly aims to include different the business model aspects. As discussed in Section II, the terminology is a bit muddled and we decided to use three different terms in order to capture studies addressing different aspects of business models.

   Finally, we decided to focus the searches the abstract, title and keywords. Based on the testing with full content, too many false positive results were returned in the test searches.
3. **Electronic searches.** We decided to use the following peer-reviewed literature databases:
   - Elsevier's Scopus database
   - IEEE Xplore Digital Library
   - ACM Digital Library
   - ScienceDirect

   From the electronics search engines suggested by Brereton's et al. [27], we did not use Google Scholar (as its results are not replicable), EI Compendex and Inspec (lack of access). Naturally, the search terms used were adopted to the capabilities of each electronic search engines. The searches were performed in January 2019.
4. **Selection.** We decided to use the following inclusion criteria when selecting the articles: 1) Studies addressing both software architectural as well as business model (as the terms are explained and understood in Section II) and their interplay; 2) Peer-reviewed articles; and 3) Articles written in English were included.

   Similarly, we used the following exclusion criteria: 1) Commentaries, prefaces, panel summaries, presentation notes, etc. non-peer reviewed content; 2) Studies not clearly discussing about the phenomena from the field of software production; and 3) Studies focusing on business models that can be presented as an abstract model and transformed from the model to an executable code; while these kind of studies can be useful for future considerations, they often consider business model in a lower abstraction level than what is mean in this study. These kinds of papers are excluded.

   The decision to include or exclude is done in two separate parts.

   4.1 Abstract and title. Firstly, we evaluated the papers based on their titles and abstracts. If a paper cannot be judged to be included or excluded based on these attributes, it was included for the next step.

   4.2 Full paper. Secondly, the full studies were evaluated.

   I both evaluation parts, the same aforementioned inclusion and exclusion criteria were used.
5. **Snowballing.** After the usage of inclusion and exclusion criteria, the select primary articles are read through. From all articles, further relevant primary studies are selected with the backward snowballing technique [28]. In this, we evaluated the bibliography of each selected articles and studied all potentially related articles by hands. The same inclusion and exclusion criteria are used than in the previous phase.
6. **Analysis.** In the last phase, the final set of primary articles are studied. In this phase, all duplicates are removed. In this study, we use conceptual-analytical approach [29]. That is, the articles are read through and summarized. The reporting in the following section is based on the synthesized results.

IV. Results

A. Descriptive statistics

The total number of studies matched in the four electronic databases is 277 (includes duplicates). Table I shows the

TABLE I
THE NUMBER OF ARTICLES FOUND IN ELECTRONIC SEARCHES (SEE PHASE 3), AND NUMBER OF ARTICLES SELECTED BASED ON TITLES AND ABSTRACTS (SEE PHASE 4.1) AND STUDIES SELECTED BASED ON EVALUATION OF FULL PAPER (SEE PHASE 4.1). THE TOTAL NUMBERS INCLUDE DUPLICATES.

Database	Search	Abstracts selects	Full paper
Scopus	144	61	8
Xplorer	99	37	4
ACM	29	1	1
ScienceDirect	5	2	1
Total	*277*	*101*	*14*

TABLE II
THE SELECTED PAPERS.

ID	Authors	Ref.	Year
P01	Gruhn & Weber	[30]	2005
P02	Ketola	[31]	2014
P03	Knodel & Manikas	[32]	2016
P04	∗Laatikainen & Ojala	[25]	2014
P05	Lechner & Schmid	[33]	2001
P06	Liu et al.	[34]	2008
P07	Moore & Mahmoud	[35]	2009
P08	Naab	[36]	2011
P09	∗Ojala	[37]	2016
P10	∗Silvander et al	[38]	2017

∗ denotes papers used in the seed set.

numbers of matched found in different search engines as well as the number of studies evaluated based on abstracts and full studies.

A large number of studies (101, including duplicates) were included into the full paper review phase due to the ambiguousness of the terms used. For example, several papers used the concept 'business model' to loosely describe any business activities that are modelled. For example, a business process of any kind and its model are referred as a business model.

The final set of papers selected for this study contains 10 unique studies. All of the studies used in the seed set for calibrating the search term were among the included articles. Furthermore, snowballing did not produce new hits to the final set. The selected papers are listed in Table II. All papers have been published from 2001 to 2017.

B. Summary of selected studies

In the following, we will shortly summarize the selected ten primary studies and how they address the questions at hand. Gruhn & Weber [30] shares the observation that there is a lack of work addressing the gap between revenue model and architecture. As a result of their paper, they propose a reference architecture for a certain kind of subscription-based revenue model.

Ketola [31] proposes of using the measurement approach, more commonly known in the mobile game development field, to adapt the business model of a product and guide also the software development process.

Knodel & Manikas [32] discuss, in their position paper, on the benefits of a reference architecture of a software ecosystems to stakeholders, activities and aspects, including the business model.

Laatikainen & Ojala [25] also note the lack of previous work addressing the connections between SaaS architecture and business models. They address how a software architecture design enables or limits the use of different pricing models in 5 cases. Their case study is among only empirical studies in this domain.

Lechner & Schmid [33] address that how an online business should take communities and their organizing into account and that the system's architecture should support this.

Liu et al. [34] discuss on the business models and software architecture in the context of e-business services. They illustrate how a traditional three-layered architecture works well, in a simple setting, to implement a business model of a common e-business service.

Moore & Mahmoud [35] propose a reference architecture that implements the needs of trusted broker for SaaS applications. The architecture is based on the needs of the business model.

Naab [36] notes that the current software architecture design methods are not able to provide flexibility required by modern business models. Thus, he outlines a proposal how to develop a method that takes the flexibility requirements better into account.

Ojala [37] studies software renting, as a pricing model, in the cloud computing with 37 interviews. Based on the study, he shows that some software architecture decisions can either limit or enable different pricing and revenue models of the product. Furthermore, he discuss that changes in revenue model are easier to implement than changes in the architecture.

Silvander, Wilson and Wnuk [38] propose on the use of context descriptions for improving architectural flexibility for being able to react to changes in business model. They report anecdotal evidence supporting their proof of concept.

V. DISCUSSION

A. Key findings

We summarize our key findings in the following:

(*i*) The area is basically unaddressed. Despite a number of studies observing the lack of work during the last two decades [eg. 23, 25, 30, 37], there is only a few studies even discussing the topic. Only two studies are based on empirical evidence.

(*ii*) The little focus, that there has been, is scattered. Some of the studies addressed specific domains and solutions with a reference architecture, some focused on pure SaaS products and their revenue models. A more holistic approach is totally lacking.

(*iii*) Also, there is total lack of usable tools, processes and models for practitioners. While business flexibility and the need for change has been noted in several papers, none has offered a practical guidelines for the professionals.

Thus, to answer to the presented two research questions: First, the extant literature does not report much on the interplay between business models and software architecture (RQ1). A few studies have been presented, yet their focus has either been on niche areas or in restricted solutions. Second, the extant literature does not report much for supporting business model experimentation (RQ2). Naab [36] notes the lack of flexibility in the methods and Silvander et al. [38] propose a method to improve flexibility. Yet, the latter work seems still to be in its starting phases and no concrete support is yet offered. However, the discussed approach seems to be promising and it will likely to lead some new oppenings in the area.

As this study showed that the intersection of business model and software architecture research remains basically unaddressed with only a handful studies even discussing the area, there are lots of promising research avenues to study.

For example, the software architecture should support experimentation. As noted in several of the paper reviewed, often the flexibility in architecture and lack of its support in the current model was seen as a problem. This is even more evident area to improve when the latest development in the minimum viable product (MVP) concept is acknowledged [cf. 39]. Thus, new openings supporting fast and flexible architecturing—and building MVPs efficiently—are needed.

B. Limitations

Naturally, there are certain issues limiting this study. First, a literature study is as good as its primary material and used search terms. For example, authors of the primary studies might use different keywords than those identified in this study. In the computing discipline, authors are known to use witty titles as well as complex and inharmonious terminology [c.f. 40], which cause both false positives (i.e., studies included in the search phase while they should not be) as well as false negatives (i.e., studies excluded in the search phase while they should not be).

Secondly, this study is limited by the used databases. For example, only ACM Digital Library has indexed the book by Hohmann [22], which remains still as the largest authoritative reference for crossing the gap between business models and software architecture. Thus, the selection of the used publication databases is crucial for any systematic literature study. In this study, we tackled this limitation by using four large publication databases. Yet, some work might have been omitted due to the used selection.

Thirdly, a somewhat large number of false positives were found due to the popularity and ambiguous of the terms used. Especially the term ‘business model’ proved to be a general term used to describe several different concepts (cf. ‘contrast’ in Shaw and Gaines’ [41] model). This might indicate that probably not all relevant terms have been included into the search term.

VI. Conclusion

This study explored the intersection of the software architecture and business model research streams by using the systematic mapping method. A total of ten studies were found. This study shows that there is an undeveloped research area addressing the connections between business models and software architecture, yet several scholars have requested that the area should be addressed as the modern business sets new requirements for flexibility and speed.

References

[1] S. Davis and C. Meyer, *Blur: The Speed of Change in the Connected Economy*. Reading: Addison-Wesley, 1999.

[2] J. Bosch, *Speed, Data, and Ecosystems: Excelling in a Software-Driven World*. Boca Raton: CRC Press, 2016.

[3] P. Abrahamsson, J. Bosch, S. Brinkkemper, and A. Mädche, “Software Business, Platforms, and Ecosystems: Fundamentals of Software Production Research (Dagstuhl Seminar 18182),” *Dagstuhl Reports*, vol. 8, no. 4, pp. 164–198, 2018.

[4] J. Järvinen, T. Huomo, T. Mikkonen, and P. Tyrväinen, “From agile software development to mercury business,” in *Software Business*. Cham: Springer International Publishing, 2014, pp. 58–71.

[5] D. Garlan, F. Bachmann, J. Ivers, J. Stafford, L. Bass, P. Clements, and P. Merson, *Documenting Software Architectures: Views and Beyond*, 2nd ed. Addison-Wesley Professional, 2010.

[6] R. Kazman and L. Bass, “Categorizing business goals for software architectures,” Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU/SEI-2005-TR-021, 2005.

[7] B. A. Kitchenham and S. Charters, “Guidelines for performing systematic literature reviews in software engineering. version 2.3.” Keele University, Keele, Staffs, United Kingdom, EBSE-2007-01, July 2007.

[8] “IEEE Standard 1471-2000. Recommended Practice for Architectural Description for Software-Intensive System,” IEEE Standards Assciation, Standard, 2000.

[9] J. Bosch, “Software architecture: The next step,” in *Software Architecture*, F. Oquendo, B. C. Warboys, and R. Morrison, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 194–199.

[10] P. Kruchten, “The 4+1 view model of architecture,” *IEEE Softw.*, vol. 12, no. 6, pp. 42–50, Nov. 1995.

[11] J. Maranzano, S. Rozsypal, G. Zimmerman, G. Warnken, P. Wirth, and D. Weiss, “Architecture reviews: practice and experience,” *IEEE Softw.*, vol. 22, no. 2, pp. 34–43, Mar. 2005.

[12] R. Taylor, N. Medvidovic, and E. Dashofy, *Software Architecture: Foundations, Theory, and Practice*. O’Reilly Media, Inc., 2009.

[13] M. Richards, *Software Architecture Patterns*. O’Reilly Media, Inc., 2015.

[14] E. Luoma, “Examining business models of software-as-a-service companies,” Ph.D. dissertation, University of Jyväskylä, 2013.

[15] A. Osterwalder and Y. Pigneur, *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. Wiley, 2010.

[16] A. Osterwalder, Y. Pigneur, and C. L. Tucci, "Clarifying business models: Origins, present, and future of the concept," *Communications of the AIS*, vol. 16, pp. 1–25, 2005.

[17] J. Gordijn, H. Akkermans, and H. V. Vliet, "Value based requirements creation for electronic commerce applications," in *Proc. 33rd Annual Hawaii International Conference on System Sciences*, 2000, pp. 10.

[18] A. Ovans, "What is a business model?" Harvard Business Review, January 2015.

[19] K.-M. Popp and R. Meyer, *Profit from Software Ecosystems*. Books on Demand GmbH, 2010.

[20] S. Lehmann and P. Buxmann, "Pricing strategies of software vendors," *Business & Information Systems Engineering*, vol. 1, no. 6, p. 452, Oct 2009.

[21] G. Laatikainen, "Financial aspects of business models: reducing costs and increasing revenues in a cloud context," Ph.D. thesis, University of Jyväskylä, 2018.

[22] L. Hohmann, *Beyond Software Architecture: Creating and Sustaining Winning Solutions*. Boston: Addison-Wesley Longman Publishing Co., Inc., 2003.

[23] A. Arsanjani, "A domain-language approach to designing dynamic enterprise component-based architectures to support business services," in *Proceedings of the TOOLS 39th*, July 2001, pp. 130–141.

[24] F. Losavio, L. Chirinos, A. Matteo, N. Lévy, and A. Ramdane-cherif, "Designing quality architecture: Incorporating iso standards into the unified process," *Information Systems Management*, vol. 21, no. 1, pp. 27–44, 2004.

[25] G. Laatikainen and A. Ojala, "SaaS architecture and pricing models," in *2014 IEEE International Conference on Services Computing*, June 2014, pp. 597–604.

[26] B. Napoleão, K. R. Felizardo, É. F. de Souza, and N. L. Vijaykumar, "Practical similarities and differences between systematic literature reviews and systematic mappings: a tertiary study," in *Proceedings of the 29th International Conference on Software Engineering & Knowledge Engineering*, 2017, pp. 85–90.

[27] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of Systems and Software*, vol. 80, no. 4, pp. 571–583, 2007.

[28] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE '14. New York, NY, USA: ACM, 2014, pp. 38:1–38:10.

[29] P. Järvinen, "Research questions guiding selection of an appropriate research method," Department of Computer Sciences, University of Tampere, Tampere, Finland, Series of Publications D–2004–5, December 2004.

[30] V. Gruhn and T. Weber, "From an e-business revenue model to its software reference architecture," in *Challenges of Expanding Internet: E-Commerce, E-Business, and E-Government*. Boston: Springer US, 2005, pp. 33–47.

[31] T. Ketola, "Quantifying software development: Applying mobile monetization techniques to your software development process," in *2014 Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES)*, July 2014, pp. 1–4.

[32] J. Knodel and K. Manikas, "Towards reference architectures as an enabler for software ecosystems," in *Proccedings of the 10th European Conference on Software Architecture Workshops*. New York, NY, USA: ACM, 2016, pp. 26:1–26:4.

[33] U. Lechner and B. F. Schmid, "Communities-business models and system architectures: the blueprint of mp3.com, Napster and Gnutella revisited," in *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, Jan 2001, pp. 10.

[34] L. Liu, S. Wang, and Z. Liang, "Service and intellectual/knowledge trading e-business model," in *2008 Second International Symposium on Intelligent Information Technology Application*, vol. 3, Dec 2008, pp. 185–190.

[35] B. Moore and Q. H. Mahmoud, "A service broker and business model for SaaS applications," in *2009 IEEE/ACS International Conference on Computer Systems and Applications*, May 2009, pp. 322–329.

[36] M. Naab, "Enhancing architecture design methods for improved flexibility in long-living information systems," in *Software Architecture*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 194–198.

[37] A. Ojala, "Adjusting software revenue and pricing strategies in the era of cloud computing," *Journal of Systems and Software*, vol. 122, pp. 40–51, 2016.

[38] J. Silvander, M. Wilson, and K. Wnuk, "Encouraging business flexibility by improved context descriptions," in *Proceedings of the 7th International Symposium on Business Modeling and Software Design*, 2017, pp. 225–228.

[39] S. Hyrynsalmi, E. Klotins, M. Unterkalmsteiner, T. Gorschek, N. Tripathi, L. B. Pompermaier, and R. Prikladnicki, "What is a minimum viable (video) game? towards a research agenda," in *IFIP I3E 2018*. Cham: Springer International Publishing, 2018, pp. 217–231.

[40] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Inf. Softw. Technol.*, vol. 50, no. 9-10, pp. 833–859, 2008.

[41] M. L. Shaw and B. R. Gaines, "Comparing conceptual structures: consensus, conflict, correspondence and contrast," *Knowledge Acquisition*, vol. 1, no. 4, pp. 341 – 363, 1989.