A proposal of a methodology for software ecosystems development

Completed Research

André L. De Gusmão
Universidade Federal do Pará
andredgusmao@gmail.com

Cleidson R. B. De Souza
Universidade Federal do Pará
cleidson.desouza@acm.org

Rodrigo Q. Reis
Universidade Federal do Pará
quites@quites.net.br

Adailton M. Lima
Universidade Federal do Pará
adailton@ufpa.br

Abstract

Software ecosystems have experienced great popularity in the last decade because they help establishing a network of users, partners, and service providers. Large companies such as Apple, Amazon, Google and Facebook have contributed to the popularity of this approach influencing different companies to establish their own software ecosystems. A recent trend is the transition from software products into software ecosystems by opening the product's architecture to allow external actors to engage in product development and, especially, refinement. However, several business, organizational and technological challenges need to be faced when transitioning to a software ecosystem. This paper presents a methodology for the development of software ecosystems based on the evolution of a software product architecture into a software ecosystem architecture. This methodology provides a guide to facilitate the development of an ecosystem architecture minimally prepared for issues that influence its launch, evolution and success.

Keywords

Software ecosystems, Software architecture, Ecosystem architecture, Platform architecture, Platform evolution.

Introduction

Bosch (2009) define a software ecosystem as being composed of "a software platform, a set of internal and external developers and a community of domain experts in service to a community of [end] users that compose relevant solution elements to satisfy their needs". For instance, in the iOS ecosystem, the software platform is the iOS operating system and its associated APIs, internal developers are Apple employees who develop the iOS platform, while external developers and domain experts are responsible for developing apps to the platform which will be purchased by end users.

Different companies are now interested in establishing their ecosystems because of the advantages a software ecosystem approach brings: increasing innovation and as a consequence, attractiveness for new users, reduced time to market, increasing business opportunities, and finally, increasing collaboration among ecosystem partners to share risk and the cost of innovation (Bosch, 2009; Manikas et al, 2016). However, this approach has disadvantages as well, including the risk of low platform adoption (Jansen, 2013), the cost of managing and coordinating platform extenders (Tiwana, 2014), and new challenges in platform evolution (Tiwana, 2010).

While some companies design their ecosystems from the scratch, it is much more common to evolve a software product into a software ecosystem because this enables an organization to expand the value of its core products through integration with other external products or services. Despite this, to the best of our knowledge, only a few papers have studied the transition from a software product into a software ecosystem. Hanssen (2012), for instance, investigates changes in an organization's software development

environment due to its change from a waterfall approach to agile development and, later on, to a software ecosystem approach. Christensen and colleagues (2014) describe a study about the creation of a Danish ecosystem for telemedicine. To guide the development of this ecosystem, the authors propose the concept of software ecosystem architecture, which is composed of three aspects: the business, the technology and the organization.

In general, these studies suggest that establishing a software ecosystem is influenced by organizational, social, business, and technological factors. Furthermore, these studies report isolated cases that are very difficult to replicate and/or generalize. To address this issue, the main contribution of this work is a *methodology* for the establishment of a software architecture appropriate for an ecosystem, i.e., a software ecosystem architecture. This methodology is aimed at the evolution of a software product architecture into a software ecosystem architecture, and takes into account organizational, social, business, and technological aspects. Our methodology is based on previous proposals by Kilamo et al (2011) and Manikas et al (2016) and it is meant to be applied iteratively and incrementally. It provides a guide to facilitate the development of an ecosystem architecture minimally prepared for issues that influence its launch, evolution and success. We describe a case study of our methodology. The results suggest that using our methodology software companies will be able to start with a software product and evolve it into a software ecosystem.

The rest of the paper is organized as follows. The next section briefly describes the related work about transitioning from a software product to a software ecosystem approach. This is followed by our proposed methodology including its steps and tasks. After that, we describe a case study in which we used our methodology. Our results and discussions are also presented. The last section presents our conclusions.

Related work: Transitioning to Software Ecosystems

Ghanam (2012) presents a case study about a company that adopted the software platforms strategy. This resulted in a taxonomy of challenges that can be faced by any company willing to adopt the same strategy. The main categories of this taxonomy are business, people, and organizational and technological challenges. Meanwhile, Hanssen's work (2012) investigates changes in a software development organization that is, initially, moving from the waterfall to an agile approach and, later on, to a software ecosystem strategy. Hanssen concludes that the changes within the organization have led to an increased collaboration between the organization's internal and external collaborators and that shared value in the studied ecosystem consists of two elements: technology and the knowledge / business domain.

Christensen (2014) describes a study about the construction of a Danish telemedicine ecosystem by creating an organization and the associated technology platform. To guide the development of the ecosystem, Christensen uses as support a concept architecture for software ecosystems. This architecture is composed of three "structures": the business, the technology and the organization. The *organizational* structure is related to the governance of the organizational elements of the ecosystem, which are formed by actors, software components and their interactions. The actors have their roles within the ecosystem influencing their objectives within it, so that the ecosystem survival is related to actors achieving their goals. The *business* structure is related to how actors create and deliver value in / from the ecosystem. This structure is important in determining costs, revenues and the sustainability of the software ecosystem. Finally, the software framework (the *technology*) consists of actors and software elements related to the development of applications in the software ecosystem. Developers are the main actors responsible for the platform and the software elements, which are structures like modules, components, and deployment nodes. These structures are important to define quality attributes such as portability, availability and security. Software ecosystem architectures are modeled from architectural descriptions using UML, showing how the platform is developed, how it behaves and how it is deployed.

In our review of the literature we identified two related works that support the development of an ecosystem from an existing product: Kilamo et al (2011) and Manikas et al. (2016). Kilamo et al (2011) presents a study about the problem of creating an open source community from proprietary software. To support the creation of a community, Kilamo discuss the OSCOMM framework that is composed of three phases that cover the process of opening the product architecture while maintaining the community that will be formed. This framework takes into account business, social, organizational and technological

aspects. Manikas et al (2016) proposes a process for designing, developing and establishing an ecosystem. This process contains three steps and a collection of activities associated with each step.

The Proposed Methodology

Figure 1 presents an overview of the proposed methodology. It has four phases, each with a set of tasks, namely: (i) ecosystem design (viability, participants and value), (ii) preparation for launch, (iii) platform engineering (removing impediments, making architectural changes) and (iv) evaluating the architecture and its monitoring process, including defining what will be measured, evaluating the architecture and conducting initial community testing.

The proposed methodology was created based on Manikas' et al (2016) and Kilamo's et al (2012) approaches. Manikas' et al (2016) approach influenced our first and last phases. More specifically, these phases were included motivated by Manikas' phases of (i) pre-analysis and ecosystem design, and (ii) evaluate and monitor. In our case, we created tasks in our first and last phases to analyze the ecosystem viability, identify ecosystem participants, identify possible extensions, identify the value proposition, and, finally, define the aspects to be measured in the last phase of the methodology. Kilamo's framework was also used to support our work, especially in phases 2 and 3 which can be mapped to Kilamo's *hands on* approach. In other words, Kilamo's release readiness rating and open source engineering phases inspired us to include tasks like identify and resolve impediments, define priorities, refactor the software architecture, plan the architectural infrastructure, and define the information to be measured.

It is important to mention that our methodology was also inspired by our previous work (de Gusmão et al, 2016). In this previous work, we describe the transition from a software product architecture to a software ecosystem architecture. In this case, the transition was conducted without a methodology to guide us. However, we documented the steps we conducted and used them to create our methodology by combining them with Manikas' et al and Kilamo's et al approaches.

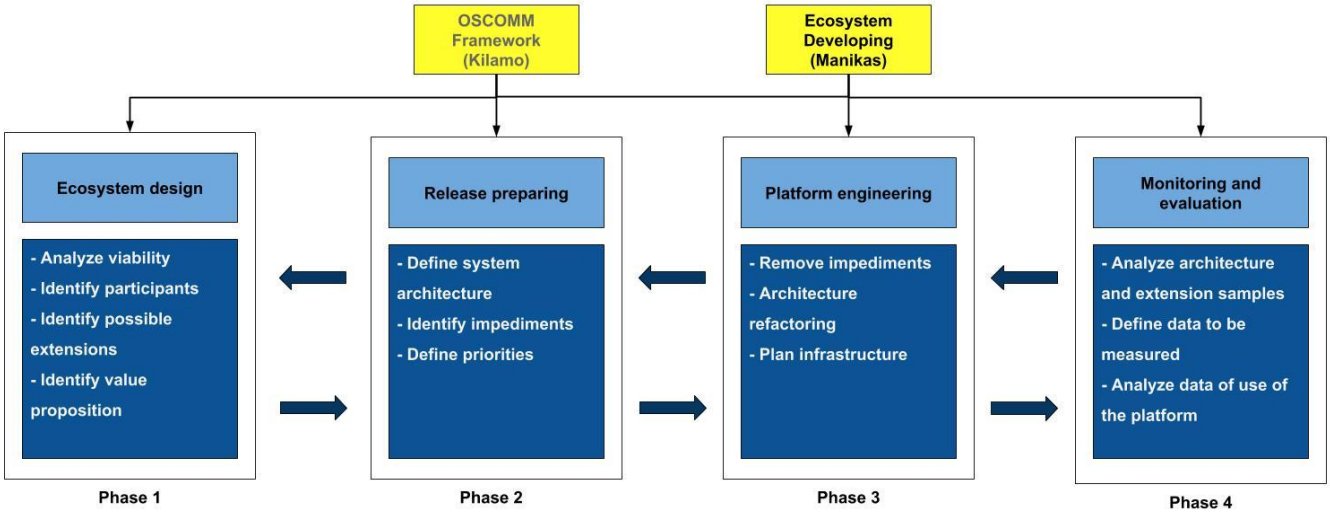

Figure 1. Proposal of a methodology for ecosystem development.

Our proposed methodology is aimed to be carried out in an iterative and incremental way. For instance, during the iterations, intermediate versions of the platform are built until it is ready to be launched. The same is done after the initial launch of the ecosystem. From this point, the next iterations generate intermediate versions –evolutions or corrections – of the platform until its launch.

Regarding the tasks, one can carry out all of them or only the ones (s)he sees as necessary in each phase according to the current status of the ecosystem. However, the execution of all the tasks has the objective of supporting the transition from a software product to a software ecosystem.

In the following paragraphs, each phase of our methodology will be described.

Phase 1: Ecosystem Design

Although there are several software ecosystems, there is no unique reason that leads to the creation of an ecosystem in the literature. However, this phase of the methodology addresses the development of a plan for a future ecosystem based on the product, and it is necessary to identify aspects related to the viability of the ecosystem, aspects that may be related to technological and social demands and opportunities. At this stage it is necessary to identify the participants of this future ecosystem in order to understand their future relationships with the platform. Identifying the value proposition of the ecosystem, as well as the identification of possible extensions of it, are tasks that help understand the opportunities and impediments in the development of the ecosystem.

For building an ecosystem it's necessary to understand its viability, that is, identify its purpose, demands, opportunities or whether the existing product business model would benefit from developments in an ecosystem. Likewise, it is necessary to identify difficulties and impediments that an ecosystem can bring to the product.

Identify the actors that interact with the product, as well as the actors that may exist in the future ecosystem to understand the participants network of it. One way to identify possible participants in the ecosystem would be to use the software supply network (SSN) concept proposed by Boucharas (2009) and used by Costa et al (2013).

Identifying possible extensions is a necessary task to help assess the ecosystem viability. It is not compulsory to think about possible extensions since the ecosystem has not yet been constructed, but in doing so it is possible to identify demands with product users which provides information necessary to evaluate the viability of the ecosystem.

Another task is the identification of value proposition, i.e., what value the ecosystem can offer to the entity responsible for it and, likewise, what value will be delivered from the ecosystem to its (future) participants. In Christensen's work (2014), for example, the business model canvas is used to identify the value proposal that the company can deliver using the ecosystem approach.

In short, in the design phase, it is desirable to construct an initial concept of the ecosystem, including its potential advantages, challenges, participants, interactions, and objectives. Through this initial effort, it is possible to decide whether building an ecosystem is viable and, then, move on to the second phase.

Phase 2: Preparation for launch

The phase of preparation for launch has as its main objective to identify the architectural modifications necessary to evolve the product into a platform, so the tasks involved include defining the current (product) and future (platform) architectures and through them identify possible impediments to the transition, which can be technological, social or business-oriented.

It is necessary to identify architectural modifications in the product architecture until a satisfactory level in the properties of a platform is reached, just as it is interesting that the platform has some quality attributes that might influence the success of the ecosystem (Tiwana, 2014). However, both properties and quality attributes have trade-offs and influence each other, which makes necessary to establish which of these will be priorities.

Another aspect of this task is to create the initial design of the platform architecture. In our previous work, de Gusmão et al (2016), the product architecture was defined using UML component diagrams. This helped the team to better understand the desired architecture and the required communication between the architectural components.

From the definition of the architecture of the product and the design of the architecture of the future platform it's necessary to identify the possible impediments (technological, social or business) for the transition between architectures. For example, what quality attributes or properties of the proposed architecture should be refined to achieve desired levels on a platform (Jansen, 2009). Through the initial data on the ecosystem gathered in the previous phase and tasks, one should select the architectural styles (Taylor, 2013) applicable to the platform. This may contribute to the identification of impediments already known in the context of software ecosystems.

In summary, the launch phase is a planning phase to meet future ecosystem demands. To do so, it is necessary to understand the current product architecture, identify possible impediments, and design an software platform architecture from the information gathered in the first phase. This leads to a list of modifications, and priorities, for the evolution of the architecture of the product into a platform.

Phase 3: Platform Engineering

In this phase, the changes in the software product architecture to establish a software ecosystem architecture have already been prioritized. Thus, it is necessary to remove the identified impediments and to actually make the planned architectural modifications, according to the defined priorities. The first task aims to remove non-architectural impediments such as license, culture and business model issues. Then, the product architecture is refactored to materialize the design made in the previous phase. In this task, architectural decisions are made respecting the possible trade-offs and properties of a platform in so that the new platform is refactored according to the platform's architecture design and the platform owner's objectives. In the last task the infrastructure of the ecosystem is analyzed. Issues such as platform communication with external applications, platform stability, evolution, and ecosystem management are the focus of this task.

This phase of the process focuses on architectural evolution, through architectural changes and the removal of impediments. In addition, in this phase, the platform infrastructure is planned in a way to meet the demands of the ecosystem, thus making it possible to launch the ecosystem.

Phase 4: Evaluation and monitoring

The last phase of our methodology aims to evaluate what has been done so far, allowing to carry out further iterations of the methodology steps if necessary. Another task to be carried out in this phase is to define how the platform will be monitored to verify the success and health of the platform ecosystem. This will make the ecosystem more stable allowing conscious and corrective decisions to be made.

The evaluation of the architecture should be done by the team responsible for it and, if possible, by future ecosystem participants. This allows one to validate whether the planning made in the preparation phase for the launch was followed in the engineering phase. The task of identifying possible extensions in the design phase can also help in this task, allowing one or more examples of extensions to be constructed and integrated into the ecosystem.

Another task in this phase is to define the types of information to be collected in order to measure the success of the ecosystem, its stability, its health and other additional aspects that help make evolutionary and corrective decisions in the platform. This monitoring is important because from the launch of the platform, it is difficult to predict the evolution speed of the ecosystem. This task should take into account the ecosystem objectives and potential ecosystem participants,

The final phase of the proposed methodology is responsible for contributing to the launch of a minimally prepared ecosystem to meet the demands of the organization (owner of the platform) and the other participants in the ecosystem. This methodology was evaluated in a case study whose details are described in the next chapter.

Description of the case study

WebAPSEE

The software product selected for the case study was WebAPSEE, a process and project management tool created at our university. This tool has been used in several projects in the last 10 years. This was important because it implied in available documentation about its software architecture and operation. WebAPSEE also has a commercial version, WebAPSEE Pro, which contains extra features. The Pro version of WebAPSEE has been in the market for more than 10 years, having users with a lot of experience in it. We believe this was important because it meant that these users would have requirements that could be met by a WebAPSEE ecosystem.

In general, WebApsee was chosen as a case by three main factors. First, the owners had an interest in evolving it for a software ecosystem. Second, there were available users with plenty of experience, and potential interest in product extensions. Third and finally, the owners' interest in scientific research, which allowed us to conduct the case study.

Application of the methodology

During the seven months in which the first iteration of the proposed methodology was carried out for the development of the WebAPSEE ecosystem, eight meetings were held with the members of its team. Participants include an undergraduate student, a software developer, an architect and one of the creators of the project. Meetings were described in Table 1. It was also necessary to conduct an interview with the team from a local software company, which uses the WebAPSEE Pro. This interview was associated to the task of analysis of the ecosystem viability that is part of the Phase 1 of our methodology.

	Initial Meeting	**Application of the Methodology**			
When	June 2017	July to August 2017	September to October 2017	November to December 2017	January 2018
Activity	Initial meeting	Phase 1	Phase 2	Phase 3	Phase 4
Participants	Creator of the project, developer	Creator of the project, developer and the team from Bel.	Creator of the project, architect, developer, scientific initiation scholarship student.	Creator of the project, architect, developer, scientific initiation scholarship student.	Creator of the project, architect, scientific initiation scholarship student.

Table 1. Process of data collection and application of the methodology

The following section describes the results of the application of the methodology.

Results

Phase 1: Ecosystem Design

Based on the analysis of the WebAPSEE and the information provided by the interviewees, it was possible to map possible partners and actors involved in the future ecosystem. In this case, we used the software supply network notation (Boucharas, 2009), SSN for short, to document and identify ecosystem opportunities. When presenting the SSN model to the interviewees, we were informed that there were students extending WebAPSEE who would be good candidates for ecosystem participants. In addition, a development company that had been using WebAPSEE for 8 years was regarded as an ecosystem participant. A group interview was conducted with this company's team and according to them WebAPSEE supports most of the company's project management needs. However, there were additional functionalities in which they were interested including team chat and time management features. These functionalities were documented as potential complements to WebAPSEE.

In a later meeting with the WebAPSEE team it was defined that a possible value proposition that the ecosystem would bring to the WebAPSEE would be to meet current demands for features and attract other companies to use the platform. For the ecosystem actors (students and the local company) the value proposition would be the ability to develop its own extensions.

Previous extensions to WebAPSEE were presented by its team: a task manager in Eclipse and a tool for managing requirements (Sales, 2008). These complements had already been implemented in the past. However, integration with WebAPSEE was done differently in each case: a new version of WebAPSEE was create to support the requirements tool, while the task manager implemented Eclipse connectors for communication with the WebAPSEE server. This again suggested that a unique way to create complements was important to foster a WebAPSEE ecosystem.

Phase 2: Preparation for Launch

To discuss the current architecture of WebAPSEE it was necessary to collect all available information about WebAPSEE's software architecture. In order to do so, we analyzed previous publications and selected those that presented details about it. After collecting the available information, a meeting was held with the WebAPSEE team to validate the information found and to understand the architecture in order to create diagrams that represented the current WebAPSEE architecture. In the meeting it was defined that the component diagrams, deployment, layers and the data model would be sufficient to describe the architecture.

Based on the architecture description, the WebAPSEE team discussed the points in the architecture that would be relevant in the transition from a product to a ecosystem. The main point reported by the team was the Façades (Gamma et al, 1995) that performed the communication between the client applications and the implementations available in the WebAPSEE server. These façades had been implemented with the purpose of providing interoperability for the server allowing multiple clients to interact with a WebAPSEE server. So, those façades contain interfaces to perform the main operations within the context of the WebAPSEE including modeling a process, changing the state of a task, uploading artifacts, and so on. Thus, according to the team, architectural modifications to enable the development of extensions could be made on top of these façades, modifying them so that WebAPSEE clients and complements could reuse them.

Although there are previous works developed on top of WebAPSEE, for instance Sales (2009), its learning curve is not simple. So, one of the opportunities envisioned by its team is to create an ecosystem is to reduce the level of knowledge needed to build extensions. In short, the ecosystem software architecture would encapsulate the complexity that WebAPSEE already has.

Phase 3: Platform Engineering

In order to change the current product architecture, a proposal for an API architecture was presented to the WebAPSEE team. The proposal consists of assembling the existing WebAPSEE interfaces (the façades) and making them available as a unique module for accessing WebAPSEE's services, i.e., a single API. This API could be used to create extensions by providing interfaces for integration with WebAPSEE. Figure 2 represents, through a component diagram, the evolution of the architecture: the facades would be encapsulated in an API that would be the integrator component of the extension with the WebAPSEE server.

In order to solve the impediment of the high learning curve of the WebAPSEE, it was assigned to the undergraduate student the task of documenting WebAPSEE facades as part of his research project. This way, this documentation would be useful as a basis for future API documentation of the ecosystem.

One should note that the iteration of the methodology focused exclusively on the design and evolution of architectural aspects for the WebAPSEE ecosystem. This happened because the product team, in the period that this study was done, was focused only on making corrections in it. That is, no plans were made to change the infrastructure of the WebAPSEE ecosystem. However, it should be emphasized that infrastructure planning is a task that must be performed whenever the product will be implemented and will evolve into an ecosystem.

Phase 4: Evaluation and monitoring

Due to limited staff resources in WebAPSEE, the tasks of this phase could not be performed during the iteration because it would involve monitoring the evolution of the community, analyzing software architecture usage and taking corrective actions. What was done in this case study was to define candidates to participate in the WebAPSEE ecosystem, like the local software company and the students involved with WebAPSEE. However, no planning was done for community monitoring.

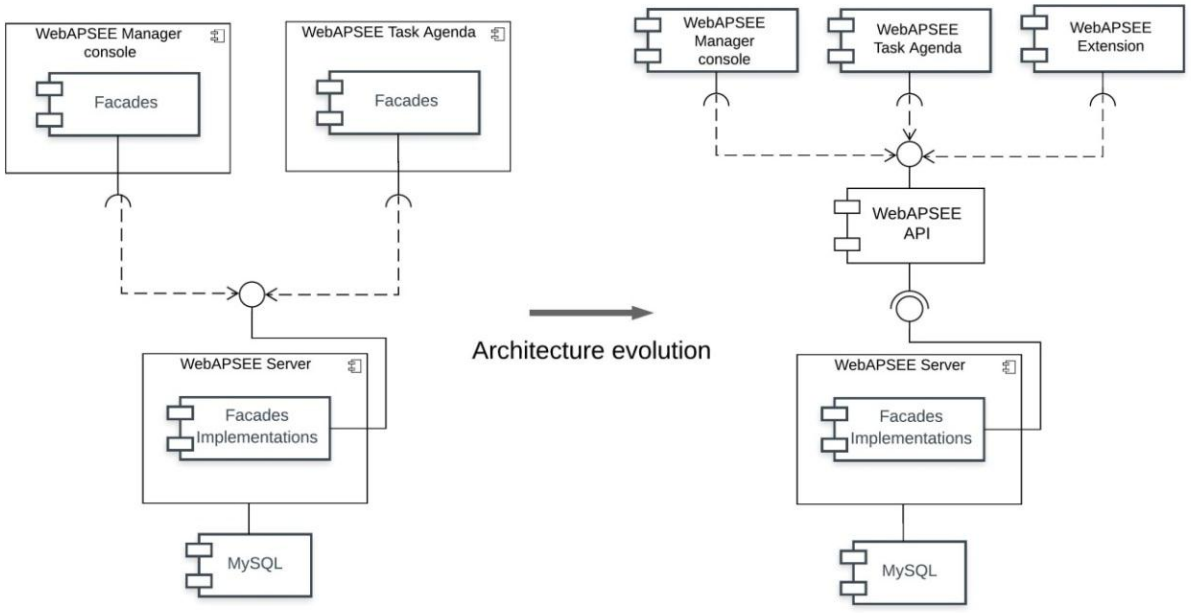

Figure 2. Evolution of the WebAPSEE architecture from a product to an ecosystem

Discussion

The proposed methodology, and most of its steps and tasks, was applied in the WebAPSEE software product. We initially assessed the viability of the ecosystem, then, after having documented the main points of the architecture, the team suggested what architectural changes should be made in the product to launch it as an ecosystem. This was done taking into account the current and future software architectures. This consisted of identifying issues to be solved in the current architecture of the product, choosing architectural styles applicable to the future architecture, and, finally, presenting to the WebAPSEE team the architectural solutions, according to the architectural style chosen, for a discussion about architectural changes. During the execution of these steps, some lessons were learned as well as limitations were observed. These are presented in the following paragraphs.

Use of the Proposed Methodology

The execution of steps and tasks can vary according to the ecosystem or focus, i.e., whether the focus is on the technological infrastructure, the participants of the ecosystem or the business model. In this way, the design phase of the proposed methodology is extremely important to guide the development of an ecosystem allied to the needs and interests of the organization owning the product to be transformed.

We believe that using a process with defined phases and tasks contributed to start evolving WebAPSEE into an ecosystem in an organized way, by defining priorities, identifying problems to be addressed and changes to be carried out in the architecture. This was possible even though, during the case study, the WebAPSEE team had limitations to implement the architectural changes. This means that the methodology was effective because it established the right questions to ask.

We also observed that it was possible, and desirable, to use previous knowledge from the literature to assist in the tasks performed in each phase of the methodology. Examples include the SSN modeling notation from Boucharas SSN (2009); the work on architectural styles from Taylor (2013); and the lessons learned in the work of Kilamo et al (2011), Costa et al (2013) and de Gusmão et al (2016) who all studied the development of new ecosystems.

On the other hand, the proposed methodology still lacks additional validation, i.e., additional iterations should be performed in the WebAPSEE or another product. For this reason, there was a concern that the methodology was generic and reproducible. Some points can still be adjusted, such as the phases and

tasks of the process, where tasks can be inserted referring to ecosystem consistency, orchestration definition within the ecosystem (interaction between participants and the platform), and security issues.

Influence of the ecosystem on the architecture

During the case study, it was also possible to analyze the influence of the ecosystem on the architecture. As a product, WebAPSEE has few people involved in the development and evolution of it making it difficult to develop extensions. A WebAPSEE ecosystem in turn would allow its participants to have much more influence on the evolution of it, as they would act by building new features and evolving existing functionalities.

In the case study, Taylor's (2013) work was also instrumental in defining a way of integrating the platform with participating ecosystem developers through architectural styles, which are a set of principles and responsibilities indicating how platform components can be grouped. The choice of architectural styles by the WebAPSEE team - the identification of impediments and prioritization of the ecosystem - has a great influence on the work of defining the ecosystem. This allows the team to adopt better solutions for the orchestration within the ecosystem, since in the product there is no defined process for the development of extensions.

Limitations

As the proposed methodology and case study had a focus on the architectural evolution of the product, this study was limited in relation to social and business aspects of the ecosystem, including the definition of a business model for the platform, licensing and intellectual property issues, and the definition of a process for the development of extensions. This is an important limitation of this work.

Despite the initial iteration to establish an ecosystem, the last phase of the project was only partially addressed by the WebAPSEE team. Because the team was not available to modify the architecture, the tasks of defining what would actually be measured and monitored were not performed, which means, it would not be possible to use this information with the future ecosystem participants. On the other hand, in subsequent iterations we believe it will be possible for the team to plan the monitoring process, and then launch the platform using the planned metrics, therefore, monitoring the ecosystem and making decisions about changes to the platform.

As already presented, the process still needs validations, especially in the evaluation and monitoring phase, however, this opens opportunities for new work in this area. One opportunity would be to use the AAMEE method (Amorim et al, 2015) during the evaluation phase to align the architectural decisions according to the needs of both, the participants of the ecosystem, and the owners of the platform.

Conclusions

With the goal of supporting the evolution of a product into an ecosystem from an architectural point of view, this work described a methodology to support the establishment of a software ecosystem. This methodology is iterative and incremental with well-defined steps and tasks. It was inspired by previous work in the literature about the development of software ecosystems, namely, Manikas et al (2016) and Kilamo et al (2011), but goes beyond them. Manikas, for instance, studies the creation of two ecosystems based on the actors and in solely the business dimension. Kilamo, on the other hand, analyzes the evolution of a product to an open source ecosystem. Our methodology seeks to extend these previous approaches. This paper also describes a case study carried out to establish a software ecosystem based on the WebAPSEE software product. Results from this case study were positive, although there are limitations, which need to be addressed in future work.

ACKNOWLEDGMENTS

This research has been partially funded by the Brazilian National Council for Research and Development (CNPq), under research grants 440880/2013- 0 and 310468/2014-0.

REFERENCES

Amorim, S., McGregor, J., de Almeida, E., and Chavez, C. 2015. "Tailoring the ATAM for Software Ecosystems", Software Architecture , pp. 372-380(doi: 10.1007/978-3-319-23727-5_30).

Bosch, J. 2009. "From software product lines to software ecosystems", International Software Product Line Conference (13), pp. 111-119.

Boucharas, V., Jansen, S., and Brinkkemper, S. 2009. "Formalizing software ecosystem modeling", Proceedings of the 1st international workshop on Open component ecosystems - IWOCE '09 (doi: 10.1145/1595800.1595807).

Christensen, H., Hansen, K., Kyng, M., and Manikas, K. 2014. "Analysis and design of software ecosystem architectures – Towards the 4S telemedicine ecosystem", Information and Software Technology (56:11), pp. 1476-1492(doi: 10.1016/j.infsof.2014.05.002).

Costa, G., Silva, F., Santos, R., Werner, C., and Oliveira, T. 2013. "From applications to a software ecosystem platform", Proceedings of the Fifth International Conference on Management of Emergent Digital EcoSystems - MEDES '13 (doi: 10.1145/2536146.2536159).

de Gusmão, A., De Souza, C., Reis, R., and Lima, A. 2016. "A study about architectural requirements in a transition from product to software platform", Proccedings of the 10th European Conference on Software Architecture Workshops - ECSAW '16 (doi: 10.1145/2993412.3003388).

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. 1995. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Ghanam, Y., Maurer, F., and Abrahamsson, P. 2012. "Making the leap to a software platform strategy: Issues and challenges", Information and Software Technology (54:9), pp. 968-984(doi: 10.1016/j.infsof.2012.03.005).

Hanssen, G. 2012. "A longitudinal case study of an emerging software ecosystem: Implications for practice and theory", Journal of Systems and Software (85:7), pp. 1455-1466(doi: 10.1016/j.jss.2011.04.020).

Jansen, S. 2013. "How quality attributes of software platform architectures influence software ecosystems", Proceedings of the 2013 International Workshop on Ecosystem Architectures - WEA 2013 (doi: 10.1145/2501585.2501587).

Kilamo, T., Hammouda, I., Mikkonen, T., and Aaltonen, T. 2012. "From proprietary to open source—Growing an open source ecosystem", Journal of Systems and Software (85:7), pp. 1467-1478(doi: 10.1016/j.jss.2011.06.071).

Manikas, K. 2016. "Revisiting software ecosystems Research: A longitudinal literature study", Journal of Systems and Software (117), pp. 84-103(doi: 10.1016/j.jss.2016.02.003).

Manikas, K., and Hansen, K. 2013. "Software ecosystems – A systematic literature review", Journal of Systems and Software (86:5), pp. 1294-1306(doi: 10.1016/j.jss.2012.12.026).

Manikas, K., Hamalainen, M., and Tyrvainen, P. 2016. "Designing, Developing, and Implementing Software Ecosystems: Towards a Step-wise Guide", in International Workshop on Software Ecosystems, Dublin, Ireland: pp. 70-79.

Sales, M. 2008. "Gerência de requisitos integrada à gerência de projetos no ambiente WebAPSEE", Undergraduate, Universidade Federal do Pará.

Taylor, R. 2013. "The role of architectural styles in successful software ecosystems", Proceedings of the 17th International Software Product Line Conference on - SPLC '13 (doi: 10.1145/2491627.2492152).

Tiwana, A. 2014. Platform Ecosystems, (1st ed.) Elsevier, Inc.

Tiwana, A., Konsynski, B., and Bush, A. 2010. "Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics", Information Systems Research (21:4), pp. 675-687(doi: 10.1287/isre.1100.0323).