Are Business Expectations Aligned with the Development Plan Made by the Software Architecture Area? A Case Study on Agile Teams in a Large Company

Marcelo Augusto da Silva[1](✉), Inaldo Capistrano Costa[1], and Eduardo Martins Guerra[2]

[1] Instituto Tecnológico de Aeronáutica - ITA, São José dos Campos, SP, Brazil
marcelo.augusto@ga.ita.br
[2] Free University of Bozen-Bolzano, Bolzano, Italy

Abstract. In the current scenario of digital transformation, understanding the interaction between the areas of business and software architecture is essential for delivering successful projects. This research aims to elucidate perceptions related to both domains, thus seeking a more efficient collaboration in the context of agile software development projects. Based on a qualitative research method, we conducted semi-structured interviews with product owners and software architects. The collected data were analyzed using Thematic Analysis to discover patterns and themes regarding the perceptions of the interviewed professionals. We found out that business areas often have a limited understanding of the technical complexities involved in software architecture, while software architects sometimes have no knowledge about business development plans. However, a continuous iteration process, supported by proper communication channels, could drive better project results. The study also revealed the potential for a proactive, integrated approach to architecture, focusing on continuous education and team alignment. Finally, bridging the knowledge gap and fostering collaboration between the two areas may lead to more efficient and effective software development processes. Future research perspectives could reveal strategies that would improve this collaboration or explore similar dynamics in different organizational contexts.

Keywords: Agility · Software Architecture · Agile Methodology · Case Study · Thematic Analysis

1 Introduction

In the current scenario of software project development, the need for quick delivery and the ability to adapt to constant changes in business requirements are key factors to deliver successful projects. The adoption of agile methodologies emerges as a response to this demand, thus promoting greater flexibility, collaboration, and continuous delivery of value [5]. However, while agile methodologies focus on adaptability and customer interaction, software architecture remains a complex technical aspect that can often be overlooked. This dichotomy between agile adaptability and the need for a solid architecture may lead to misalignments between the expectations of the business area and the software product that has been developed [4].

The alignment between the expectations of the business area and the software development project team is essential to ensure that the technological solution that has been developed is in sync with the business goals [12]. This approach not only enhances the chances of meeting the referred business requirements, but also ensures that the software is efficient, scalable, and sustainable in the long term [13]. Integrating the architecture team into the development process is essential to guarantee that the software can evolve along with the ever-changing demands of the business [6]. Through effective collaboration, the understanding of business objectives becomes clear, and the software architecture team can provide the necessary guidelines for a successful implementation [14]. The lack of such alignment may result in solutions that fail to meet the needs of the business, in addition to presenting technical challenges, thus affecting system availability, performance, and maintenance [15].

This article aims to explore these potential misalignments, taking as a case study the software project development environment of a large cooperative financial system in Brazil - the software developed by the referred organization employs agile development practices and is widely used throughout the country. Thus, through this research, we aim to understand the nature of these eventually existing discrepancies and offer insights that can help development teams harmonize agile practices with the requirements demanded by software architecture, thus ensuring that both walk side by side in favor of more aligned and effective solutions. To investigate this phenomenon, we were guided by the following research questions:

- RQ1: How does the business area perceive software architecture and what relevance do they assign to the development of software projects?
- RQ2: What is the level of knowledge of the software architecture area in relation to the application development plan?
- RQ3: How do the architecture and business teams perceive the iterations in the development process of software projects?

In order to answer our above-mentioned research questions, we conducted ten semi-structured interviews [1] with professionals who are currently working on software development projects in the environment of the referred financial cooperative in Brazil. The study included five professionals who are currently working in the business area and are responsible for stating the requirements that the software must meet and five other professionals who work in the technology area and are responsible for structuring the architecture that the software must follow to be implemented. We then proceeded to an inductive thematic analysis [2,3] of the interview transcripts.

In this study, the results emerge as a deep reflection of the existing dynamics between the areas of business and software architecture in contemporary organizations. Throughout the analysis, we were able to reveal distinct and sometimes conflicting insights about the role and relevance of software architecture in the context of project development. Such findings throw light on areas of misalignment and also identify potential for optimization in the collaborative process between technical and business teams, thus suggesting an intrinsic need for realignment in order to deliver more effective software solutions.

The contributions of this work go beyond the mere identification of these dynamics, providing a practical road map to facilitate effective integration between architecture and business teams. Based on the recommendations proposed herein, this study acts as a guide for organizations seeking to strengthen their collaborative approach, emphasizing the importance of mutual understanding and aligned goals. The insights and strategies presented in this paper can potentially serve as a reference point for organizations willing to align their technical initiatives with their business strategies more effectively.

The article is structured as follows: Sect. 2 presents the Software Architecture theme and its relevance in software projects. Section 3 contextualizes our research by connecting it to similar studies on the subject. Section 4 details the method and tools employed in our data collection and analysis. In Sect. 5 we present and discuss what we found by analyzing the interactions between the referred areas during project development. We come to a conclusion in Sect. 6, where we reflect on our findings and point to possible directions for future research.

2 Software Architecture Relevance

Software architecture can be understood as the structure of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution. It establishes the fundamental organization of a system in terms of its components and their interactions, and is critical to determining software quality, performance, and longevity. The IEEE, in its standard definition, describes software architecture as "the fundamental structure of a system, which consists of software components, their externally visible properties, and the relationships among them" [29].

Bass et al. [6] describe software architecture as the structure of a system that includes software components, the relationship between these components, and the properties of both elements. In this context, software architecture is more than just the structure; it also defines how the components interact and how the structure evolves over time.

According to Shaw and Garlan [21], software architecture is a discipline that provides a structural point of view and provides techniques to help create highly structured and modular systems.

The interaction between software architecture and non-functional requirements (NFRs) plays a crucial role in the software project development process.

NFRs, such as performance, security, and reliability, significantly influence architectural decisions. Certain quality attributes hold pivotal importance in architectural design stages, due to their direct impact on the system's structure and design pattern choices. Additionally, proper categorization of NFRs is necessary for effectively evaluating software architecture. Simultaneously, managing these requirements in specific development contexts, like model-driven development, underscores the need for a systematic approach from the outset. This integration fosters better conditions for the final architecture to align with stakeholders' expectations and the system's operational requirements [6,32,33].

In parallel, agile software development processes have gained prominence due to their ability to provide value in an iterative and incremental way, prioritizing collaboration and response to change. However, aligning architecture strictness with the flexibility found in agile methods can be a challenge. Architectural decisions often require early planning and consideration, while agile methods value adaptation and continuous delivery. Thus, to achieve optimal balance, it is vital that development and architecture teams collaborate closely and adjust their processes and practices in order to align the benefits of robust architecture with the agility of development processes [6].

In the organization studied, software architecture plays a leading role, which is evidenced by the existence of a unit in the IT sector consisting of professionals specialized in this field with the purpose of satisfying the inherent needs of software development projects. This unit actively collaborates with sectors vital to IT such as security, infrastructure, and operations, so that solutions reach adequate standards of security, availability, and robustness. This organization generates solutions at a national scale, serving approximately 8 million users who carry out financial transactions both in person at business units and via self-service channels. It is also important to mention that the organization operates in a highly regulated sector of the economy. Thus, its software projects are often shaped by external influences, which include transactions that must adhere to SLAs determined by regulatory bodies, for example.

In summary, software architecture provides a blueprint for the system, representing its main properties and how they interact. It is the key artifact for understanding any system's large components and how they are orchestrated to work together.

3 Related Works

Upon investigating the existing literature on the alignment between the business and the software architecture areas as well as the impact of organizational models on agile development, several prominent works were identified. These works provide a critical perspective on the challenges, solutions and trends associated with this subject. Within the context of the research questions included in this study, we can highlight the following works.

Rozanski and Woods [7] delve deeply into software architecture and the relationship with stakeholders; they don't specifically focus on "the business perception of software architecture" as an isolated topic. Instead, they provide a comprehensive approach to address the concerns of all stakeholders, including but not limited to the business itself. The main focus of this work is to provide a structured approach to software architecture and communicate this architecture to stakeholders.

Garlan [20] in turn, discussed how software architectures often evolve in response to external pressures. Market changes, new technologies, and the emergence of competing standards may lead to unplanned adjustments in architecture. This study highlights the importance of a flexible and adaptable architecture to address these challenges.

Research by Dingsøyr et al. [8] highlighted that continuous collaboration and frequent iterations are essential for agile development. They noticed that teams that work closely together and review their processes regularly are more likely to understand and implement requirements effectively, which results in higher-quality software.

Kniberg and Ivarsson [18], in their famous white paper about the Spotify model, described how guilds and other organizational structures can promote collaboration and knowledge sharing. Their work provides robust evidence that such frameworks can mitigate challenges that are commonly faced in software development, especially those related to communication between technical and business areas.

The study by Viviani et al. [31] highlights the critical management of NFRs in software projects, emphasizing their propensity for change and late definition, aspects often underestimated in software architecture planning. The research, through responses from professionals with extensive experience, revealed that NFRs undergo significant alterations, often late in the development cycle, highlighting a notable gap in the elicitation, validation, and management of these requirements. This discovery underscores the pressing need for agile approaches that can accommodate such uncertainties and changes, ensuring that the software architecture maintains its integrity and relevance over time, considering that the change and evolution of NFRs are inevitable in the software evolution cycle.

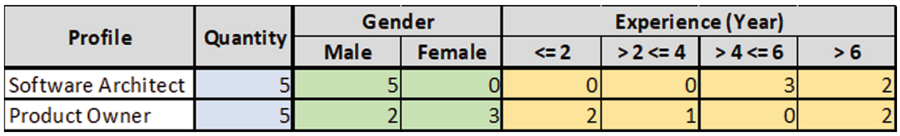
The above-mentioned works highlight the complexity and importance of effective alignment between the business area and the technical teams. Proper integration and continuous communication are essential to ensure that the software developed is aligned with the company's goals and needs.

4 Research Method

To deepen the understanding of misalignments between the business area's expectations and the architectural solutions implemented in software projects within the studied organization, a case study approach was chosen [17] with a qualitative research method. Semi-structured interviews were used as the data collection instrument [2,3] with professionals involved in the software development process.

Interviews. From May 2022 to July 2023, ten interviews were conducted with professionals who work on software projects in the organization studied. Initially, two interviews were carried out: one with a software architect and the other with a product owner. A preliminary analysis of these data was carried out to determine whether they would be adequate to guide our research development. After this initial assessment, the interviews continued. All sessions were conducted online in Portuguese through video conference and lasted about 45 min each.

Participants. In order to assess the perception of professionals in the business area and those responsible for software architecture, five professionals corresponding to each profile were selected. The business professionals interviewed were appointed by managers of business product areas and the IT professionals - all software architects - were appointed by the manager responsible for the software architecture area. After being assigned to participate in the research by their respective managers, all were duly contacted, briefed on the issue under study, and invited to voluntarily participate in the research. All the appointed professionals agreed to participate and therefore the interview session was scheduled. Figure 1 details the interviewees' qualifications. In the interview session, which was recorded with prior authorization from the participants, previously prepared questions were presented to the participants who then expressed their perception about the issue raised.



Profile	Quantity	Gender		Experience (Year)			
		Male	Female	<= 2	> 2 <= 4	> 4 <= 6	> 6
Software Architect	5	5	0	0	0	3	2
Product Owner	5	2	3	2	1	0	2

Fig. 1. Profile of the interviewees.

Research Ethics. During the recruitment process, participants were informed about the purpose of the study, the content of the questions, and the affiliation of the interviewer. In the organization studied, it is widely known that there are professionals on their staff, people who take up a professional master's degree course which is encouraged by the organization itself. Aware of this condition, participants agreed to participate in this study which can bring benefits to the organization's software development process. At the beginning of each interview, the interviewer made sure to announce the purpose of the study and the anonymous nature of its content, in addition to explaining the dynamics of the interview and obtaining verbal consent from the interviewee. Since the interviews were conducted using Microsoft Teams[1], they were recorded with the participant's consent and transcribed automatically by the tool itself during the course of the interview, and the interviewee also viewed the content of the transcript.

[1] https://www.microsoft.com/pt-br/microsoft-teams/log-in.

Data Analysis. To analyze these transcripts, we developed an inductive coding scheme. Inductive coding was used to investigate the participants' insight into the software project development process in the organization studied - the aim was to identify dysfunctions that would create a gap between what the user expects the software to deliver throughout its development and how the project development area prepares the software architecture to meet present and future requirements. In this approach, themes emerge from the data, and codes are signed when concepts become apparent in these data. This means that the researcher encodes the data without trying to fit them into a pre-existing coding framework or their own analytical biases [2].

To develop the analyses, the ATLAS.ti[2] software was used and the thematic synthesis process proposed by Braun and Clarke [2] was followed. A researcher began the analysis by carefully reading the transcripts and getting immersed in the data. Subsequently, specific text segments were identified, labeled, and transformed into initial codes. To ensure coding accuracy and cohesion, a random selection of these codes was submitted to the research group for evaluation. This allowed for a uniform understanding of the codes among the members. The following step involved the conversion of these codes into themes, which were subdivided into sub-themes and higher-order themes. The researcher then thoroughly reviewed all themes and data, ensuring their congruence, which led to the elaboration of a thematic map of the analysis. To add rigor to the process, another researcher was introduced to reassess the codified texts and established themes. The final structure of themes and sub-themes emerging from the analysis can be viewed in Fig. 2 - details and further discussion will be covered in the subsequent section.

5 Results and Discussion

Upon carrying out semi-structured interviews, it was possible to identify key patterns and themes related to the interaction dynamics between the business, development, and software architecture teams in the context of project development. The main findings have been organized into 5 themes, as follows:

5.1 Established Architectural Infrastructure

One of the main findings of this study refers to the existence of a well-established reference architecture in the organization which, in general terms, is aligned with the non-functional requirements of the various software developed and used in the referred environment. This implies the existence of a pre-defined set of standards, principles, and components that are considered standard for the construction and evolution of systems. Reference architecture serves as a blueprint, ensuring that systems are consistent, interoperable, and aligned with organizational strategy. One of the interviewed architects made the following statement:

[2] https://atlasti.com/.

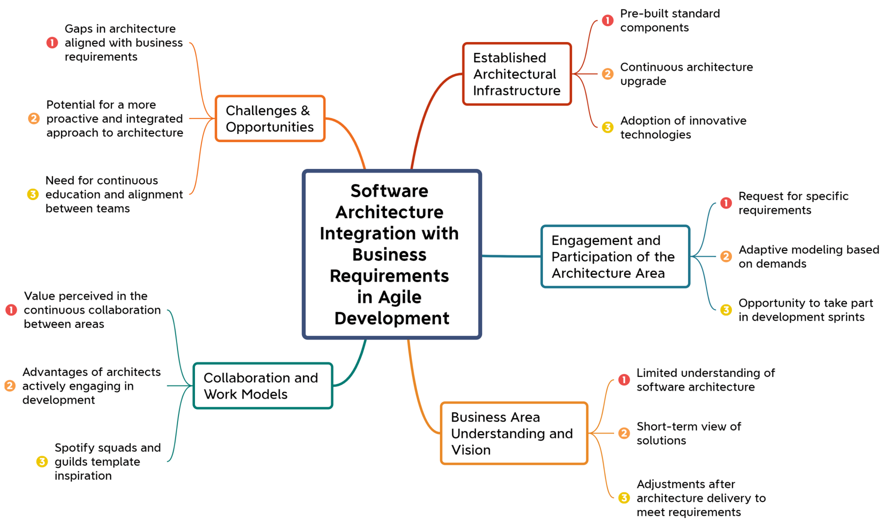

Software Architecture Integration with Business Requirements in Agile Development
Challenges & Opportunities
Gaps in architecture aligned with business requirements
Potential for a more proactive and integrated approach to architecture
Need for continuous education and alignment between teams
Established Architectural Infrastructure
Pre-built standard components
Continuous architecture upgrade
Adoption of innovative technologies
Engagement and Participation of the Architecture Area
Request for specific requirements
Adaptive modeling based on demands
Opportunity to take part in development sprints
Collaboration and Work Models
Value perceived in the continuous collaboration between areas
Advantages of architects actively engaging in development
Spotify squads and guilds template inspiration
Business Area Understanding and Vision
Limited understanding of software architecture
Short-term view of solutions
Adjustments after architecture delivery to meet requirements

Fig. 2. Final thematic map.

"reference architecture would be a guide to good practices. Practices that must be adopted as a norm. As a rule, they are drivers to be applied to business data" [Architect-1].

We must emphasize the importance of reference architectures, as they provide a solid foundation for development and help reduce costs by avoiding rework and speeding up delivery by reusing previously validated components [6]. This architecture may also help ensure regulatory or security standard compliance, in addition to facilitating communication between teams as it creates a common language and shared understanding regarding standard technical solutions [7].

We also identified that this reference architecture has a regular evolution plan that seeks to provide modern solutions, compatible with what is offered by the market, thus keeping the software ready to meet the referred business requirements. Among the various reports on the maintenance of this reference architecture, one of the architects stated: *"We have an overall plan when it comes to creating new components, not a specific architecture plan. There's the creation of new products and everything must be done in a new architectural design"* [Architect-5]. Shaw and Garlan [21] stated that as business needs and the technological scenario evolve, software architecture must be adjusted and reviewed to continue meeting emerging requirements and challenges.

Another relevant matter about architecture maintenance identified in this study refers to prospecting innovative technologies, as mentioned by one of the interviewed architects, *"Among its attributions, the architecture team must prospect new technologies and bring them to the company and, in a way, make them operational, so that these technologies can be used by the development teams"* [Architect-4]. Foote and Yoder [22] mention that evolution and innovation are inseparable in the context of software development. By introducing innovative technologies, it is possible to address new challenges and optimize the systems' performance and efficiency.

Finally, we were able to verify that the architects' statements showed that they are committed to promoting continuous evolution, adopting good practices, and incorporating innovations, which may be an indication of the organization's architectural maturity.

5.2 Engagement and Participation of the Architecture Area

Based on the statements given by the interviewees, it was possible to identify a series of practices and challenges related to the engagement of the architecture area in the development of software projects in the organization featured in this study. These observations are in line with existing discussions in the literature about the role of architecture in agile teams and the integration of architects in development teams.

The participation of architects often begins when there are specific demands that require technical assessments. This can be evidenced in the statement given by one of the interviewed architects: *"We need to assess if [this demand] will have support. So what shall we do? Shall we have a chat about it? Let's call an alignment meeting to discuss some architectural pre-documentation aspects related to software architecture"* [Architect-1]. Figure 3 shows a representation of the identified working model. Kruchten [23] argues that, in agile environments, teams often find themselves in situations where architectural expertise becomes vital, especially when new challenges arise.

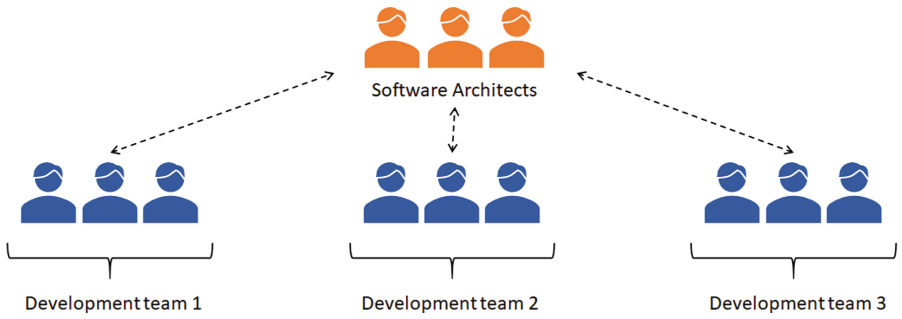


Fig. 3. A Separate Team of Software Architects Works with Multiple Development Teams [30].

Another architect made the following comment: *"a call comes in for us to assess some data, for example, and that's when we become aware of what is being developed"* [Architect-1]. Currently, architects are called upon mainly when specific architectural demands arise. This model may result in late design decisions and possible rework if architectural considerations are not duly identified at the beginning of the development cycle [6].

As for an earlier performance of architects along with the development teams, one of the interviewees made the following comment: *"The software architecture's pre-documentation meeting does not exist. It all comes from the gut feeling of the development team, really. So I would say that the software architecture documentation is the trigger for us to start being aware and acting together with the team, but this informal approach with teams that have more expertise, as I mentioned before, may occur. This is what we must assess"* [Architect-1]. A proactive participation of architects throughout the whole development cycle may facilitate the early identification of architectural challenges, allowing for solutions that are more knowledgeable and aligned with business needs and technical constraints [9]. In addition, their constant presence may serve as an ongoing education channel for the business team, helping them understand the role and importance of software architecture in their projects.

In summary, the interaction between architects and development teams, as observed in the above-mentioned statements, points to a need for greater integration and continuous collaboration. Practices observed in the organization featured in this study, although in line with some published works, also suggest that there are opportunities for a more systematic and continuous approach to architectural engagement. Encouraging closer collaboration between business, development, and architecture areas can not only improve the quality of the delivered solutions but also promote a more harmonious working environment, with fewer conflicts and misunderstandings [10].

5.3 Business Area's Understanding and Views

In the agile development environment, the role of software architecture is often underestimated or misunderstood, especially by business teams [11]. Evidence of that could be identified in the organization studied, where the business team has little clarity on what constitutes software architecture and how relevant this component is to project delivery. This fact was evidenced by the speech of one of the professionals in the business area, as highlighted below, about what software architecture is: *"I'll tell you my understanding of it based on the little contact I've had. I understand that they create a framework and from that framework, they can build something there. I don't know exactly what that is"* [Product Owner-5].

Another feature identified in the study is the absence of a structured, long-term product development plan. In terms of evolution and innovation, responses mentioned incremental deliveries. One of the respondents said: *"On the product itself, I don't see much change for the next 5 years, as a form of business. It basically depends on the Central Bank"* [Product Owner-4]. The tendency to focus on immediate needs and not anticipate changes over a long-term horizon may lead to decisions that are not scalable or flexible [24]. The regulatory role of the Central Bank, as mentioned by the interviewee, also highlights the importance of considering external factors that may influence product decisions and their development.

The observation that architectural adjustment often occurs in response to significant incidents, as mentioned by one of the respondents - *"It usually comes* *after an incident happens. After there's been a lot of fuss over it...”* [Architect-5] - highlights an often delayed response to changing needs. This reactive type of approach may lead to one-off solutions and possibly more costly and complex refactoring operations in the future.

Guerra et al., in their study, present the idea of “architectural triggers”, which are predefined events or conditions that indicate the need for architectural reviews or adjustments [25]. These triggers can be seen as a proactive approach, allowing teams to identify and respond to potential architectural issues before they evolve into significant crises. By incorporating such triggers into the development process, professionals can better anticipate and manage necessary changes, ensuring that the software develops in a more controlled and sustainable manner.

The above-mentioned observations suggest that there is a need for better alignment and communication between the business and technical areas, ensuring that the long-term implications of architectural decisions are well understood and taken into consideration in the development of software projects.

5.4 Collaboration and Work Models

Collaboration and effective communication between different areas of software engineering are critical to ensure that end products are robust, scalable, and meet end-user needs. This collaboration is essential not only between individual members of the software team but also between different teams and areas of expertise [6]. The comment made by one of the interviewees - *“Not just architecture, but also infrastructure and tests should be part of my day-to-day business development. Without silos. Today it's not like that here. I think this [approach] makes things very complicated”* [Architect-1] - on the need for a daily collaboration reflects the opinion of many authors who suggest that integrated and collaborative teams are more effective in delivering high-quality software [26].

Furthermore, the emergence and success of multidisciplinary team models, as highlighted by another interviewee - *“Why did they decide to break up and create these cross-functional teams? Because now the success metric of that whole team, that cross-functional team, happens to be the project, which in the end is what matters to the client”* [Architect-4] - resonate with the advantages perceived in agile development and continuous integration. Figure 4 shows an image that represents the above-mentioned work model. The agile method has become one of the most adopted software development methodologies precisely because it focuses on collaboration, continuous feedback, and adaptation to change [27].

Spotify's model of squads and guilds, also mentioned by one of the interviewees - *“in addition to chapters, you can use the collaboration of other organizations, like, you can organize the chapters, which is done between teams, but you can also have tribes that are larger groups working on a similar business pillar. So, there are still other organizations you can use to redeploy teams”* [Architect-4] - is a particularly successful adaptation of Agile and Lean principles. Not only does it bring together multidisciplinary teams (or “squads”) that have autonomy and responsibility for delivery, but it also allows for effective

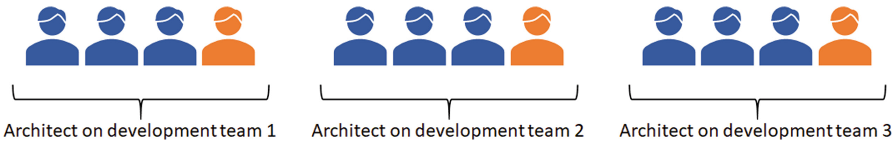

Architect on development team 1
Architect on development team 2
Architect on development team 3

Fig. 4. Each Development Team Has One Software Architect [30].

cross-communication through "guilds", ensuring that knowledge is shared across squads and that there is consistency where necessary [19].

Nevertheless, it is worth noting that while models such as Spotify's may work well for some organizations, a successful implementation of these models will depend on the organization's culture, structure, and goals. Thus, it is essential that companies, such as the one studied in this case, carefully consider their individual needs and contexts before adapting such practices [8].

5.5 Challenges and Opportunities

The role of software architecture in IT projects is crucial not only in terms of technical decisions but also to ensure that the final solution is aligned with the needs of the referred business. However, comments made by the interviewees have highlighted some substantial challenges that must be assessed and addressed.

One of the product owners addressed a common problem in software projects - *"we end up finding it a bit hard to get the expected result. Sometimes when we are talking to the professionals responsible for the requirement analysis process or when we speak to the designer, we create [the product] in some way, and then, when it comes to developing it, we end up finding many barriers in this regard"* [Product Owner-2] - in which there is a disconnect between the initial requirements, the proposed design and the actual deployment [28]. This often leads to rework, project delays, and solutions that do not fully meet the expectations or needs of the business. This lack of alignment emphasizes the importance of clear and effective communication during every stage of the project, as well as the need for a flexible architecture that can adapt to changes as they arise [6].

The observation made by another product owner who participated in the interview suggests a need for greater integration and collaboration between software architects and development teams - *"We spend a lot of time thinking about things like: Is this how we are going to create it? Shall we do it like this? No, but it doesn't have to be like this, or sometimes even developers have some ideas that can make our processes a lot easier. It goes back and forth, multiple times because I feel like there is this gap"* [Product Owner-2]. As noted by Fairbanks [11], a more proactive approach to architecture can lead to more robust and effective solutions. Furthermore, this collaboration may serve as a means of sharing knowledge and best practices, thus ensuring that everyone on the team is on the same page.

Finally, the comment - *"Teams, they work in different ways, which ends up being a complicating factor when we have to deal with several products. We end* *up having to use different approaches with different teams and that ends up complicating our day-to-day work"* [Product Owner-1] - made by another product owner who participated in the interview highlights the challenge of working with multiple teams that may have different methodologies or work patterns. Dingsøyr et al. [8] noted that the standardization of work methods may improve the efficiency and quality of the developed software. However, it is essential to recognize and respect differences between teams and find a balanced approach that allows for flexibility while maintaining a certain level of standardization.

5.6 Validity Discussion

Case studies, especially within the context of software engineering research, face several validity challenges. Runeson and Höst [17] outline various threats to validity in case studies, and these can be extrapolated and applied to the case study carried out in this research, which used qualitative research with thematic analysis. Examining the validity of our study in terms of internal, external, construct validity and reliability, we offer the following considerations:

Internal validity is typically linked to studies aiming to establish causal relationships and elucidate specific conditions or problems [17]. Since our research sought to understand misalignments in software development without emphasizing causal connections, we did not dwell on internal validity. External validity, in turn, assesses whether findings can be generalized beyond the studied contexts. Our results come from a major Brazilian financial institution. To expand the generalization of these insights, additional research in different industries or regions based on more extensive samples is recommended.

Construct validity refers to the alignment of collected data with research questions. In this scope, we prepared a questionnaire and tested it with a product owner and a software architect. Subsequent analysis ensured that the data properly covered the research topic. Moreover, we interviewed experienced professionals from the studied organization, who were appointed by their respective managers.

Lastly, reliability is linked to the objectivity of data analysis, regardless of the researchers involved. At this stage, the first researcher established a case study protocol to ensure consistency in the research methodology. The analysis of the collected data was conducted by the first and second researchers, aiming to ensure a comprehensive view of the software development process and its potential to build an architecture that meets business demands. Consequently, a third researcher reviewed the classifications carried out by the other researchers in order to give them more objectivity and impartiality.

6 Conclusion

The present study aimed to deepen the understanding of the relationship between the business area and software architecture in the context of project development. Through our investigations, we managed to identify that the perception of the business area on software architecture is diverse - many see it as a fundamental structure to build or adapt functionalities, while others have a more limited view, focused on immediate deliveries (RQ1). This perception may vary, but it reinforces the importance of clear and continuous communication between teams in order to guarantee the effectiveness of the project.

When it comes to understanding software architecture in terms of application evolution, we can see that there is an effort to keep up-to-date and aligned with demands and changes in the business plan. However, it is challenging to keep in sync, given the dynamic nature of businesses and the rapid changes in technology (RQ2).

The perception of iterations in the development process revealed the need for closer collaboration between the architecture and business teams. The introduction of agile practices and collaborative models, such as those inspired by Spotify, may be a promising way to improve this integration (RQ3). However, continuous alignment and the formation of cross-functional teams are essential factors to overcome the challenges identified in this study.

Finally, our study found that the software development process in the studied organization is more exposed to misalignments between the expectations of the business area and the developed solution. However, this fact does not imply that the results of the delivered projects fail to meet expectations. Such a phenomenon was not studied in this research. Still, it is important to point out that the development and deployment processes, when not optimized, can lead to multiple iterations, potential rework, and late delivery.

Our findings highlight the importance of mutual understanding between business and software architecture areas, revealing knowledge gaps and friction points in the context of development process iterations. By elucidating these challenges, the study offers insights for organizations to seek closer and more integrated collaboration, thus promoting greater efficiency in project development. This improved understanding may encourage targeted training, adjustments in organizational models, and the introduction of appropriate collaboration tools, thereby leading to heightened performance in both sectors.

We believe that the findings presented in this study may serve as a starting point for future investigations and improvements in the field of software engineering. In future studies, it may be beneficial to delve into practical strategies to improve communication between the areas of business and architecture, explore the impact of different organizational models on effective collaboration, and investigate how tools and technology platforms can be used to facilitate mutual understanding. Additionally, it would be of great value to analyze the evolution of these interactions over time, considering the rapid changes in technology and business demands, as well as to deepen studies on continuous education and alignment mechanisms between teams in agile environments.

References

1. Adams, A., Lunt, P., Cairns, P.: A qualitative approach to HCI research (2008). https://oro.open.ac.uk/11911/1/9780521870122c07_p138-157.pdf. Accessed 31 Aug 2023
2. Braun, V., Clarke, V.: Using thematic analysis in psychology. Qual. Res. Psychol. **3**(2), 77–101 (2006). https://doi.org/10.1191/1478088706qp063oa
3. Cruzes, D.S., Dyba, T.: Recommended steps for thematic synthesis in software engineering. In: International Symposium on Empirical Software Engineering and Measurement, pp. 275–284. IEEE (2011). https://doi.org/10.1109/esem.2011.36
4. Stal, M.: Refactoring software architectures. In: Agile Software Architecture [S.l.]. Elsevier (2014). https://doi.org/10.1016/B978-0-12-407772-0.00003-4
5. Babar, M.A., Brown, A.W., Mistrík, I. (eds.): Agile Software Architecture: Aligning Agile Processes and Software Architectures. Newnes (2013)
6. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. Addison-Wesley (2012)
7. Rozanski, N., Woods, E.: Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives. Addison-Wesley (2012)
8. Dingsøyr, T., Nerur, S., Balijepally, V., Moe, N.B.: A decade of agile methodologies: towards explaining agile software development. J. Syst. Softw. **85**(6), 1213–1221 (2012). https://www.sciencedirect.com/science/article/pii/S0164121212000532. Accessed 31 Aug 2023
9. Kruchten, P.: Contextualizing agile software development. J. Softw. Evol. Process **25**(4), 351–361 (2013). https://doi.org/10.1002/smr.572
10. Ambler, S.W.: Agile architecture strategies for scaling agile development. Agile Model. (2012)
11. Fairbanks, G.: Just Enough Software Architecture: A Risk-Driven Approach. Marshall Brainerd (2010)
12. Laudon, K.C., Laudon, J.P.: Management Information Systems: Managing the Digital Firm (2004)
13. Sommerville, I.: IEEE software and professional development (2016). https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7420501. Accessed 31 Aug 2023
14. Fowler, M.: Patterns of Enterprise Application Architecture (2002)
15. Kruchten, P.: The Rational Unified Process: An Introduction (2003)
16. Yin, R.K.: Case Study Research: Design and Methods. Sage Publications (2013)
17. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. (2009)
18. Kniberg, H., Ivarsson, A.: Scaling Agile @ Spotify with Tribes, Squads, Chapters and Guilds (2012). https://blog.crisp.se/wp-content/uploads/2012/11/SpotifyScaling.pdf. Accessed 31 Aug 2023
19. Smite, D., Moe, N.B., Levinta, G., Floryan, M.: Spotify guilds: how to succeed with knowledge sharing in large-scale agile organizations. IEEE Softw. **36**(2), 51–57 (2019). https://doi.org/10.1109/MS.2018.2886178
20. Garlan, D.: Software architecture: a roadmap. In: The Future of Software Engineering, pp. 91–101 (2010). https://doi.org/10.1145/336512.336537
21. Shaw, M., Garlan, D.: Software Architecture: Perspectives on an Emerging Discipline. Prentice Hall (1996)
22. Foote, B., Yoder, J.: The selfish class. In: Proceedings of the 3rd Conference on Pattern Languages of Programming (1998)

23. Kruchten, P.: Agility and architecture: can they coexist? IEEE Softw. **27**(2), 16–22 (2010). https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5420791. Accessed 31 Aug 2023
24. Boehm, B., Turner, R.: Balancing Agility and Discipline: A Guide for the Perplexed. Addison-Wesley (2004)
25. Wirfs-Brock, R., Yoder, J., Guerra, E.: Patterns to develop and evolve architecture during an agile software project. In: Proceedings of the 22nd Conference on Pattern Languages of Programs, pp. 1–18, October 2015. https://doi.org/10.1145/3424771.3424800
26. Cohen, D., Lindvall, M., Costa, P.: An introduction to agile methods. Adv. Comput. **62**, 1–66 (2004)
27. Highsmith, J., Cockburn, A.: Agile software development: the business of innovation. Computer **34**(9), 120–127 (2001)
28. Boehm, B., Basili, V.R.: Software defect reduction top 10 list. Computer **34**(1), 135–137 (2001)
29. IEEE Standard 1471-2000: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE (2000)
30. Mihaylov, B.: Towards an agile software architecture (2015). https://www.infoq.com/articles/towards-agile-software-architecture/. Accessed 31 Aug 2023
31. Viviani, L., Guerra, E., Melegati, J., Wang, X.: An empirical study about the instability and uncertainty of non-functional requirements. In: Stettina, C.J., Garbajosa, J., Kruchten, P. (eds.) International Conference on Agile Software Development, vol. 475, pp. 77–93. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-33976-9_6
32. Chen, L., Babar, M.A., Ali, N., Chen, X.: A taxonomy of non-functional requirements for software architecture evaluation. IEEE Access **8**, 31753–31769 (2020)
33. Cuesta, C.E., Pérez, J.A., Pérez, M.S.: Dealing with non-functional requirements in model-driven development. In: Advances in Intelligent Systems and Computing, pp. 15–27 (2019)