Outsourcing with distributed teams in large-scale agile environments

Rafael Camara[1], **Marcelo Marinho**[2],**Suzana Sampaio**[2],
Ivaldir Honório[1], **Hermano Moura**[1]

[1]Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 15.064 – 91.501-970 – Recife – PE – Brazil

[2]Departamento de Computação
Universidade Rural de Pernambuco (UFRPE) – Recife, PE – Brazil

(rcf8,ihfj,hermano)@cin.ufpe.br,(marcelo.marinho,suzana.sampaio)@ufrpe.br

Abstract. *Outsourcing projects continue to grow through a more distributed and connected environment among teams from all over the globe. Due to it, many customer-vendor relationships are being formed in multi-site settings around a large-scale agile project. Companies are constantly looking for suppliers to assist and develop their complex products, although large-scale agile projects with many teams require some attention to the outsourcing project. Based on it, this study aims to conduct a systematic literature review regarding how outsourced distributed teams from large-scale agile projects are handling their operation. From 23 studies, it was possible to synthesize six distinct themes and 15 patterns of outsourcing distributed teams working in large-scale contexts.*

1. Introduction

The search for lower production costs, access to global talents, and a 24-hour development cycle led modern software engineering (SE) to look for distributed teams over the globe. Such a trend has been shaping the software development approach of many companies. However, by the time the projects grow and the complexity arises, organizations are constantly outsourcing the development of their solutions to achieve similar benefits of Distributed Software Development (DSD) and also acquire scalability [Batra 2009].

Organizations with distributed outsourced teams are usually involved in large-scale projects which require the tailoring and scaling of agile methods to handle the dynamic environment [Krancher 2020]. However, standard agile methods do not fit well in large-scale outsourced projects [Matthiesen and Bjørn 2017]. Since late changes can generate more costs for the client and fixed-bid contracts can be a problem for a vendor that wants to be open to changes, many companies are looking forward to facing the challenges of outsourcing distributed teams with large-scale projects by tailoring the agile methods and the outsource nature [Bass 2016].

Considering this scenario, this study raises a practical question: "How do large-scale agile projects with outsourced distributed teams handle their operation?". To answer such a question, we conducted a Systematic Literature Review (SLR) [Kitchenham and Charters 2007] that extracted studies from 2001 to 2021 that described the outsourcing operation of client and vendors teams in large-scale contexts. A total of

23 studies were selected. From those studies, six themes and 15 patterns regarding the operation activities of those teams have emerged.

This study contributes both practitioners and researchers to a better understanding of how distributed outsourced teams work and handle the challenges of large-scale environments using agile methodologies. By focusing on enriching the body of knowledge, the patterns are presented in a guidance structure with *Problem*, *Context* where the problem exists, *Solution* that describe actions that solved the problem, and *Consequences* that gather results and trade-offs of the patterns.

The remainder of this study is organized as follows: In Section 2, we introduce the background to the present problem. Section 3 describes the research method and research question. Sections 4 and 5 present the results and implications, respectively. Finally, Section 6 provides some concluding remarks and areas of future research directions.

2. Background

2.1. Distributed software development

Distributed Software Development (DSD) is the name given to designate software projects that are developed by teams spread beyond the company's boundaries and in different locations [Marinho et al. 2019]. Such boundaries could be national that describes DSD or international that refers to Global Software Development (GSD) [Marinho et al. 2018].

Despite the nature of the distribution, companies that seek to distribute their development are looking for shared benefits, such as expenses cuts, access to global and high skilled talents, round-the-clock development, and low labor costs [Marinho et al. 2019]. However, those benefits do not come for free; together with them, most organizations need to deal with challenges related to inadequate communication, cultural differences, team coordination, time-zone differences, and control of the development process [Marinho et al. 2018].

2.2. Large-Scale Agile Projects

Large companies constantly looking to distribute their development are often handling large-scale projects. Sometimes, those large-scale projects are complex and belong to critical sectors. A large-scale project also requires development methodologies to scale and fit the environment settings.

Large-scale is also a metric that requires some definition. This study follows the taxonomy presented by Dingsoyr *et al.* [Dingsøyr et al. 2014] which defines the scale of agile software development projects based on the number of teams involved. Small-scale projects are supposed to have only one team. Large-scale projects can have 2 to 9 teams, and the coordination will require scaling approaches. Finally, projects with more than ten teams can be called very large-scale projects; for those projects, a scaling framework is also recommended.

2.3. Outsourcing in large-scale settings

Outsourcing can be described as the relationship between a client and vendor through a contract that aims to deliver a service or a product [Gupta 2022]. Companies outsourcing their software development are constantly handling distributed teams and large-scale

projects. Similar to the benefits of DSD, organizations that outsource their development are looking for skilled labor for a low cost and without the full responsibility of coordinating the outsourcing teams. However, those benefits bring some problems. First, clients outsourcing their development cannot always correctly monitor the progress of the vendor team, which can lead to quality control problems [Gupta 2022]. Second, clients need to share their code repositories and documents with the vendor, which can expose them to the risk of losing intellectual property [Gupta 2022].

To handle some of the problems that outsourcing may generate, clients establish rules regarding onshore and offshore software development, concentrating all the complex and vital activities such as design architecture, domain expertise in onshore teams [Bass 2016, Matthiesen and Bjørn 2017], while focusing only on software implementation in offshore teams [Hossain et al. 2009, Noordeloos et al. 2012]. Based on those points, this study focused on describing patterns regarding the operation of outsourcing distributed teams from large-scale projects.

3. Research Method

A Systematic Literature Review [Kitchenham and Charters 2007] was conducted in this study to evaluate most of the relevant literature regarding outsourcing approaches in large-scale distributed environments. Our goal was not to uncover all recorded approaches regarding outsourcing since this study is part of more extensive work regarding agile tailoring practices in large-scale distributed settings. However, we aim to collect sufficient information about the outsourcing theme in those challenging contexts.

We applied automatic search in five large bibliographic databases to identify a set of relevant studiess that the research goals should match. The search string aimed to gather the keywords of each research theme by combining the keywords of "distributed software development", "scaling agile", "large-scale", and "outsourcing". Based on this structure, the search string can be accessed in the research protocol (See in `https://bit.ly/3tV65TD`). We used this string to search the metadata relating to journals and conference proceedings in IEEEXplore, ACM Digital Library, SpringerLink, Scopus, and Wiley bibliographic databases.

3.1. Document selection

The search produced 1520 references from 2001 to 2021 (IEEE = 31; ACM = 836; Springer= 191; Wiley = 58; Scopus = 404). The references were limited from 2001 onwards since the Agile Manifesto was published this year. The selection process had three phases: (i) an initial selection of research results that could reasonably satisfy the selection criteria (outlined next) based on a reading of the studies' titles and abstracts; followed by (ii) a selection against these criteria from the initially selected list of studies based on a reading of their introductions and conclusions; (iii) and finally, the studies were fully read and the ones related to outsourcing with distributed teams in large-scale settings were selected.

3.1.1. Inclusion/Exclusion criteria

The following criteria guided the selection of studies. We *included*: (IC1) complete, peer-reviewed, published studies; (IC2) studies directly related to the research question; (IC3)

the study is available via the university library services accessible to the authors during the time of the research; (IC4) keywords of the research string appear on abstract and/or author keywords; (IC5) studies directly related to outsourcing in large-scale distributed settings. We *excluded*: (EC1) texts not published in English; (EC2) technical content, e.g: editorials, tutorials, keynote speech, white studies, thesis, dissertations, technical reports, books; and (EC3) short studies (<=4 pages); (EC4) studies not related to outsourcing in large-scale setting; (EC5) studies related to education matters.

Each study was checked to ensure there was no duplication or replication. For example, if a given study was published in two different journals with a different order of primary authors, only one study would be included; this would usually be the most comprehensive or recent study. By doing this, it was possible to identify 90 duplicated studies and two replication. After excluding the duplicated ones, the selection phase was executed, and we identified 247 studies for phase 1. During phase 1, the introduction and conclusion of each study were read, and 91 studies were selected for the full read. In phase 2, the total read removed ten studies, and the final set consisted of 81. Of those 81 studies, the last phase focused on selecting the ones related to outsourcing with distributed teams on a large scale. Finally, 23 studies [Kommeren and Parviainen 2007, Cho 2007, Hole and Moe 2008, Paasivaara et al. 2008a, Paasivaara et al. 2008b, Hossain et al. 2009, Paasivaara et al. 2009, Batra 2009, Paasivaara and Lassenius 2010, Kussmaul 2010, Batra et al. 2011, Hossain et al. 2011, Noordeloos et al. 2012, Dorairaj et al. 2012, Daneva et al. 2013, Paasivaara et al. 2014, Koch et al. 2014, Lehtinen et al. 2015, Bass 2016, Matthiesen and Bjørn 2017, Vithana et al. 2018, Gupta et al. 2019] passed to the data extraction phase. The 23 studies were selected from IEEE (1), Wiley (1), ACM (4), Scopus (8) and Springer(9).

3.2. Study Evaluation

We based the studies evaluation process on the method of rigor and relevance evaluation assessment proposed by [Ivarsson and Gorschek 2011]. The rigor aspects are evaluated on a scale of 0 (“weak”), 0.5 (“medium”), and 1 (“high”), and it has three dimensions which are (i) context, (ii) study design, and (iii) threats to validity. The maximum value of rigor is 3.

On the other side, the industry relevance is concerned with the impact a study can have on industry and academia, considering relevant research topics and real industry settings. The relevance aspect has a binary score, 1 for present and 0 for not present. The aspects are (i) the subjects of the study that are described as the people involved in the case, e.g industry professionals; (ii) the context in which the study was conducted, e.g industrial settings; (iii) the scale of the applications used in the study, e.g realistic industrial applications; and (iv) the research method used. The maximum value of relevance is 4.

3.3. Data Extraction

Each selected study was fully read to extract information regarding practices of outsourcing distributed teams that work in large-scale environments. This step can be described as the process of gathering all relevant data from the studies to answer a research question [Wohlin et al. 2012]. In order to synthesize the data and ease the management, we decided to conduct some recommended steps by [Cruzes and Dyba 2011]. We also categorized the studies based on contribution facets, as suggested by Petersen et al. [Petersen et al. 2008]

and the research facets from Wieringa et al. [Wieringa et al. 2006]. The open-coding activity was executed in the MAXQDA Software [1].

4. Results

In this section, we present an overview of the studies from the SLR. Also, the tailored outsourcing practices are discussed and presented in the following subsections.

4.1. Overview of the Outsourcing Studies

From the 23 studies related to outsourcing, the majority of them, 21 studies, used a qualitative approach, while 3 used a quantitative approach. Based from Wieringa *et al.* [Wieringa et al. 2006] facets, most of the studies were categorized as evaluation studies (15 papers), followed by experience studies (5 papers), solution studies (2 papers) and finally philosophical studies (1 paper) (See figure 1).

The distribution of contribution type facets derived from Petersen *et al.* [Petersen et al. 2008], were classified as lessons learned (18 papers), followed by model contributions (2 papers), framework studies (2 papers), and finally advice studies (1 paper) (See figure 2).

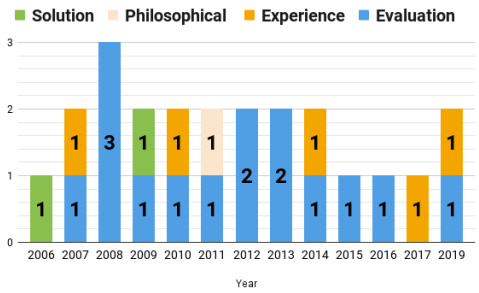

Figure 1. Research type facets over time.

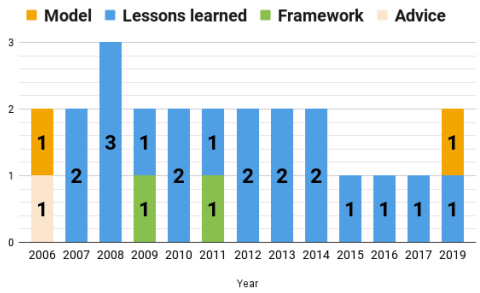

Figure 2. Contribution type facets over time.

Regarding the research methods, most of the studies used the case study methodology (12 papers), although only 2006 has no case studies regarding the theme. Multiple-case studies were the second most used method (5 papers), followed by experience reports (3 papers). Finally, the least used methodologies were ethnography, theory and grounded theory (2 papers each), and literature review with only one study.

In Figure 3, we can see the coefficient of rigor and relevance of each study. Only 2 out of 23 studies had the highest score on rigor and relevance, 3 and 4, respectively. None of the studies reached a score of 0 on rigor, but one study had a 0 on relevance, which can be considered disappointing from the industrial perspective. Finally, the rest of the studies had a rigor aspect value between 1 to 2,5.

Furthermore, the outsourcing theme has a high industry relevance score due to real industrial scenarios that those studies take place (Figure 4). However, since some do not give proper attention to aspects such as threats to validity and study design, the industrial scenario leads them to lower scores.

[1] www.maxqda.com

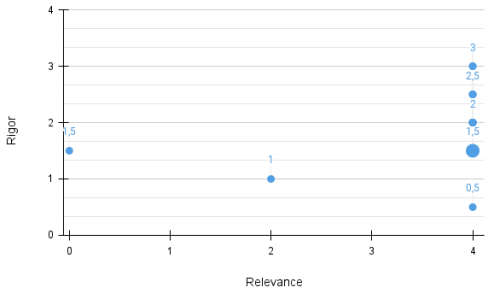

Figure 3. rigor and Relevance of outsourcing studies.

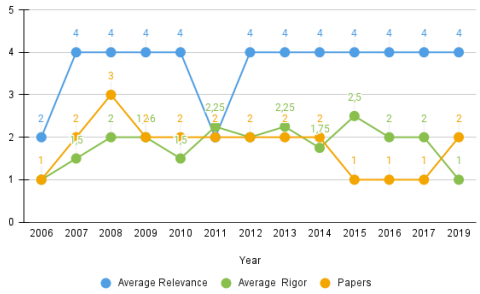

Figure 4. Average Rigor and Relevance over time.

4.2. How do large-scale agile projects with outsourced distributed teams handle their operation?

This section describes the six themes regarding how outsourcing distributed teams were working on large-scale distributed projects. The themes emerged through the steps of thematic synthesizes [Cruzes and Dyba 2011]. From six themes, we have a total of 15 patterns. Each theme and its patterns will be described in the following subsections. For better comprehension, the 15 patterns are described based on the structure suggested by Välimäki and Kääriäinen [Välimäki and Kääriäinen 2008].

4.2.1. Contracts in outsourcing large-scale distributed projects

Contracts in Outsourcing Environment (2): **Problem**: agile values and principles have put the challenge when companies are dealing with third parties. First, the agile value "customer collaborations vs contract negotiation" cannot be followed by the book since standard outsourcing practices work based on contract negotiations [Batra 2009]. Second, welcoming changes can be trouble for fixed-bid contracts or vendors with pre-defined resources [Batra 2009]. **Context**: a client that works with agile methodologies can expect the vendor to be open to customer interaction over contract negotiation. However, it can lead to severe problems regarding development and delivery if the vendor works with pre-defined plans, resources, and fixed-price contracts. **Solution**: Contracts cannot be avoided due to the agreements that protect both client and vendor [Batra et al. 2011]. Further, contracts without fixed prices can also be a way out, although the client must be aware that all changes and new work hours will be counted. **Consequences**: The fixed-price contract can stifle an agile process of a large client, although it can also protect the same client from spending a a lot of money.

Supplier Determine Price After Sprint (1): **Problem**: a study aims to solve the problem regarding price-fixed contracts and also accept late changes [Kussmaul 2010]. However, vendors and clients need to work with closed-range budgets to accomplish such a goal. **Context**: Both customer and supplier agree to work with dynamic contracts during the interactions [Kussmaul 2010]. **Solution**: to minimize the risks, after each sprint, the supplier is responsible for determining the final price based on the actual effort made, and the scope of work completed [Kussmaul 2010]. **Consequences**: the teams become more responsive to uncertainty and change requests during the planning. Also, as a result, both

parts could minimize costs since a price range was calculated to accommodate changes in the contracts. Finally, determining the price later avoided conflicts regarding the fixed-price contracts.

4.2.2. Agile ceremonies and artifacts

Planning meetings with onshore and offshore teams (4): **Problem**: synchronizing meeting hours with distributed offshore and onshore teams can be a challenging task in large-scale projects. The planning meetings need to accommodate typical hours of teams with different time zones. **Context**: Planning meetings involving both teams are held in different phases to involve everybody that is spread in global settings. In some cases, the companies can send the offshore team to the onshore site [Paasivaara et al. 2008a, Paasivaara et al. 2008b, Paasivaara et al. 2009]. **Solution**: a phased meeting model which has a distributed meeting, followed by two local meetings in offshore and onshore teams. The distributed meetings can be held during the typical hours of both sites through teleconferences tools, where the Product Owner is responsible for presenting the prioritized backlog items [Paasivaara et al. 2008a, Paasivaara and Lassenius 2010]. After that, the onshore team continues the planning meeting, and the offshore team continues the work done by the onshore team discussing on the draft plan received. **Consequences**: in a distributed environment, the phased planning meeting provided enough visibility for both teams regarding the work that needs to be done and also an open forum for discussion [Paasivaara and Lassenius 2010]. However, separating the meetings regularly in a distributed environment demands a large effort [Paasivaara et al. 2009, Paasivaara and Lassenius 2010].

The Main Supplier as the owner of agile ceremonies (1): **Problem**: the development of a product by two unaffiliated IT suppliers working in the same baseline [Vallon. et al. 2013]. **Context**: one leading supplier (MS) that holds the contract and provides most of the staff, while the additional supplier (AS) is a subcontractor with developers or testers without scrum roles defined [Vallon. et al. 2013]. The MS does not involve the AS in regular scrum events or the scrum board and burndown charts. Consequently, the absence of AS involvement leads to several issues regarding communication and coordination. **Solution**: establish long-lived single-site Scrum teams, enabling electronic tool support, and using BDD to make implicit requirements knowledge explicit and transparent to all parties [Vallon. et al. 2013]. **Consequences**: the refusal to use electronic tools and to seek a healthy relationship with unaffiliated suppliers can lead to several problems. However, if both suppliers are treated in the same way without specific restrictions due to different contracts, their relationship can improve into both seek project success.

4.2.3. Common outsourcing organization designs

Domain expertise stays onshore (8): **Problem**: during the outsourcing of a project, the domain expertise regularly belongs to the client, which is the owner of the business. Also, most of the time, the hired outsourced supplier cannot translate business needs to function requirements since the domain business does not belong to them. **Context**: The necessity to keep domain expertise and requirements elicitation in the onshore site

[Hossain et al. 2011, Bass 2016, Matthiesen and Bjørn 2017, Gupta et al. 2019]. **Solution**: If an offshore team has difficulty keeping up with tasks that need domain expertise, the managers can involve them in less cross-site coordination tasks [Hole and Moe 2008]. In this way, the onshore team focuses on requirements analysis, architecture decisions, and new features while the offshore team works with features implementation and system configuration [Kussmaul 2010]. Finally, the team with domain expertise should actively involve offshore developers and testers to participate in requirements decisions and gradually pass the domain knowledge to them [Noordeloos et al. 2012]. **Consequences**: This approach can cause issues regarding the relationship with the offshore teams, which can feel left out of the project just for not mastering the business. From another perspective, gradually involving the offshore teams can bring benefits, since the seeking for engagement can improve the relation of both teams and also put everybody in the same ruler of business knowledge [Hole and Moe 2008, Kussmaul 2010].

Technical Expertise stays offshore (3): **Problem**: most of the companies that are looking for an offshore team from an outsourcing approach are looking to access high-skilled talents, cheap labor, and a 24/7 development cycle through other regions. However, most of this expertise is used mainly for technical purposes focused only on software development, which can cause issues in the teams' relationships. **Context**: Offshore team with the unique responsibility of implementing the features previously specified by the onshore team [Noordeloos et al. 2012, Bass 2016]. **Solution**: carrying out development activities with adequate onshore support and integration practices between teams [Hossain et al. 2009]. **Consequences**: hire an offshore team to focus specifically on the technical aspects of the project can bring positive impacts as long as they get proper support during all the development phases [Hossain et al. 2009]. However, the lack of support for those team members spread around the globe can create the "us and them" feeling [Noordeloos et al. 2012].

Vendor Responsibilities (2): **Problem**: in an outsourced project, the vendor needs to be aware of its responsibilities, which involve participating in all phases of a development process. Further, the vendor also needs to be aware of the type and size of the project to evaluate how better the development process can be conducted. **Context**: In big projects with many teams, the vendor needs to be close to the customer from the beginning to capture, analyze and transform business requirements into technical stories [Daneva et al. 2013]. However, whether the project is small, the vendor needs to reflect on how the operation activities are worthy of keeping [Dorairaj et al. 2012]. **Solution**: the vendor can be close enough to the customer to capture the requirements in a user story manner with acceptance criteria, followed by other steps such as architecture design, code, and test [Daneva et al. 2013]. In small duration projects, all development activities can be held on the customer site itself [Dorairaj et al. 2012]. **Consequences**: when vendors are genuinely involved in the process of serving the customer by understanding their needs and, based on it building stories that reflect the system specifications, the project has many chances to succeed [Daneva et al. 2013].

Offshore Team relationship (4): **Problem**: working with teams in distributed software development settings can lead to many problems related to communication, coordination, and control of the development process [Marinho et al. 2018]. The vendor can deal with the same issues in outsourced projects while handling both onshore and off-

shore teams. The difficulty of communication between sites can create the "us and them" feelings, which hinders the development process and diminishes trust in the offshore team [Paasivaara et al. 2014]. Such feelings can be also caused when new developers are hired in one location and developers from other locations do not try to get to know the new ones [Cho 2007]. **Context**: the appearance of a "yes-sir" culture in the offshore team where the members do not try to understand the sprint goal may cause the implementation of failed increments [Koch et al. 2014]. **Solution**: adoption of scrum can increase the feeling of belonging in the offshore team, bringing the team closer to the onshore site as well as the client site with constant interaction with them [Noordeloos et al. 2012]. **Consequences**: due to the agile process of reducing the gap among the offshore team members, the team can increase their knowledge of the client's domain which makes their work more interesting. As another consequence, bringing interactions closer to the offshore team with the client can give them a chance to improve personal skills and also their motivation [Noordeloos et al. 2012].

Outsourcing Organization Models (2): **Problem**: in the process of outsourcing the product development, the customer should choose the suitable model that fits his activities. The model can be a multi-site project in which each site is responsible for a module of the project or a closer approach in which the vendor interacts more closely with the client site to understand the complexities of the project better. **Context**: Depending on the required project communication, different models can fit in the relation of customer and vendor [Batra 2009]. The company decides to work with multi-site projects allocating particular team groups with the required competence to develop specific components of the solution [Kommeren and Parviainen 2007]. **Solution**: customer-supplier should align how agile methods can be adopted in their environments [Kommeren and Parviainen 2007, Batra 2009]. **Consequences**: in a multi-site project approach, the hiring of different team groups for specific components can minimize the problems of concentrating the development in a unique supplier [Kommeren and Parviainen 2007], although it also can lead to an overhead of coordination and communication. On the other side, the regular interaction between customer and vendor through client representatives in both sites can make the development process more expensive when compared to a more direct agile model of interaction [Batra 2009].

4.2.4. Third party suppliers

Third parties suppliers involvement (1): **Problem**: many customers look for hiring third parties and additional suppliers to handle some activities of the development process that can cause overhead to them [Lehtinen et al. 2015]. However, third parties suppliers require much expertise in the customer domain to make those things work right, which sometimes does not happen. **Context**: the customer hires third parties suppliers for a specific activity such as data mass testing, requirement elicitation or design tasks [Lehtinen et al. 2015]. **Solution**: as a consequence of the lack of domain by the POs, the client needed to hire third parties to handle some of their activities to keep the project on the road [Lehtinen et al. 2015]. However, bringing those third-party suppliers may not solve the problem correctly since they might not have the domain expertise. Due to it, the best approach for the clients is to look for specialists suppliers or train their POs in the domain. **Consequences**: as a result of third parties eliciting requirements with-

out a proper domain, the stories can be inaccurate and lead the POs to have difficulty in communicating and prioritizing requirements that they do not understand.

Same backlog for different suppliers (1): **Problem**: when a vendor is dealing with an additional supplier (AS) through subcontracts, it is necessary to pay enough attention to the AS activities. Keeping the same backlog without sharing the whole story can cause problems in the relationship of both suppliers. **Context**: More than one supplier, with the main one and all using the same backlog, although only the main with complete control [Vallon. et al. 2013]. **Solution**: due to the lack of a PO and SM to handle those activities involving the backlog and planning, some developers from the AS site can become unofficial SM, and PO [Vallon. et al. 2013]. Despite the subcontracts, the best way to follow is to engage both suppliers in the development process with their functions clearly defined. **Consequences**: the use of the same backlog for both suppliers cannot cause adverse impacts for itself, although the lack of vital roles in the team can lead to exclusion of the AS from agile ceremonies.

4.2.5. Outsourcing roles and functions

Ambassador and Coordinator Role (6): **Problem**: despite the regular composition of a distributed agile team, an outsourcing project requires some additional roles. Those roles are coordinators, and ambassadors [Paasivaara et al. 2008a, Batra 2009]. **Context**: The ambassador is a client employee responsible for staying in the outsourcing vendor site, interacting with the team members daily, driving the future of the project, and providing business context to the offshore team [Paasivaara et al. 2008a, Paasivaara et al. 2008b, Daneva et al. 2013, Batra 2009, Batra et al. 2011]. The coordinator is a vendor employee responsible for staying with the outsourcing client, interacting with the client team daily, and also being a focal point to the client regarding the vendor's activities [Batra 2009]. **Solution**: the ambassador and coordinator roles can be seen as both ends of a bridge that supports the communication channel between the customer and the outsourcing vendor [Batra 2009, Dorairaj et al. 2012]. The role has some different ways to be implemented; in most cases, hiring a specialist to form the bridge with people that has higher soft skills in communication aspects [Paasivaara et al. 2008a, Paasivaara et al. 2008b, Batra 2009, Batra et al. 2011, Daneva et al. 2013].**Consequences**: establish two different roles can provide better communication among the teams involved in the project.

Outsourcing Team Composition (2): **Problem**: an outsourcing team may have some tailored roles to accommodate the nature of a distributed large-scale context. **Context**: the definition of main roles for an outsourced team with business analysts, tech lead, domain owner, delivery team head, and test scenario team lead which work collaboratively making the requirements decision during iterations [Batra 2009, Daneva et al. 2013]. **Solution**: the ambassador and coordinator roles work as a bridge for customer and vendor communication. The client's design team can help to understand the requirements, while the integration team works closely with the architecture to handle components integration [Batra 2009]. The business analysts must know the business process to be able to translate it into user stories; The teach leads must serve as a technical reference for the team; Domain owners must master the business domain of the client and share it with the others; Delivery Team Heads are responsible to ensure that the development incre-

ments will be delivered following all functional and non-functional requirements; Finally, Test Scenario Team Leads are responsible for managing the technology dependencies and constraints related to the project. **Consequences**: tailoring the roles to fit an outsourced environment better is not only a suggestion but sometimes a necessity to achieve project success. Such adaptations were made to the regular roles, and the insertion of new ones can improve the dynamics of the team [Batra 2009, Daneva et al. 2013].

4.2.6. Outsourcing environments characteristics

Regulation problems between Offshore and Onshore team (1): **Problem**: the development of software from a regulated area can bring some additional challenges. Some regulated areas, such as health, aviation or financial, do not allow offshore teams to work on solutions of the onshore site [Noordeloos et al. 2012]. **Context**: The onshore client team is responsible for the design and requirements decision, while the offshore team is mainly responsible for the development and test of the product. In some cases, the financial area is regulated by law or regulatory requirements. **Solution**: the onshore vendor team needs to work more closely on the technical tasks that require access to regulated areas [Noordeloos et al. 2012]. **Consequences**: Due to the regulated environment, some limitations cause delays and bottlenecks in the solution delivery [Noordeloos et al. 2012].

Careful Supplier Selection (2): **Problem**: The unreliable vendor selection can cost the client the whole project [Kommeren and Parviainen 2007]. **Context**: when a company is looking to hire an outsourced vendor to assist or even develop its product, it requires trust in the supplier [Kommeren and Parviainen 2007]. **Solution**: to include the supplier selection as an integral part of the product road-mapping process. The client must evaluate the vendors' capabilities by looking for past experiences or even making an audition [Kommeren and Parviainen 2007]. Build a relationship with the vendor, evolving it into an alliance [Kussmaul 2010].**Consequences**: both approaches to selecting a vendor carefully and also building a healthy relationship with them bring good benefits. However, auditing or conducting a proof-of-concept can become expensive for clients and vendors.

5. Discussion

Our study has shown 15 different patterns spread across six themes regarding the work of distributed outsourcing teams in large-scale contexts. Those patterns reinforce the way of work in outsourcing large-scale distributed environments and how both customer and vendor manage their day-to-day to accomplish project success. Further, it was possible to realize that managing customer-vendor relationships in large-scale projects requires much tailoring in agile methodologies. As presented in the studies of Batra and Daneva [Batra 2009, Daneva et al. 2013], following the agile principles and values by the book may harm the development process of outsourcing distributed teams. However, if these issues are overcome, it can lead to customer-vendor relationship well-being.

It was possible to observe some relationships among patterns from different themes, which can be combined to solve mutual problems. Patterns from theme *Contracts in outsourcing large-scale distributed projects* (See section 4.2.1) and *Common Outsourcing Organization Designs* (See Section 4.2.3) related to fixed-contracts, late changes and vendor responsibilities can be merged to solve the same issues at once. Since late changes,

fixed contracts and dynamic prices are not commonly feasible for vendors and customers. It is possible to engage those patterns into the vendor responsibilities pattern. According to Daneva *et al.*, [Daneva et al. 2013] the client PO must be aware of communicating any changes to the vendor early, and also, the vendor assumes that the client master the business value of the solution. Further, the vendor was responsible for allowing only two types of changes: (I) changes from a business perspective related to regulation issues; (ii) changes from a technical perspective that occurs when tech leaders from the vendor find stories too expensive for the architecture. Finally, the approach defined by the vendor of developing "delivery stories" also facilitates the process of avoiding late changes and fixed contracts by refining the user stories that came from the client [Daneva et al. 2013].

The patterns of *Careful Supplier Selection (2)* and *Technical Expertise stays offshore (3)* can be related to some of the success factors studied by Khan *et al.* [Khan et al. 2011]. The most common success factors for vendor selection, according to Khan *et al.* are appropriate infrastructure, cost-saving, and skilled human resources. Keeping the technical expertise offshore is the pattern that most look for cost-saving and access to talented individuals for a cheaper price [Hossain et al. 2009]. However, since the COVID-19 pandemic, remote work has taken hold, and looking for offshore teams became a matter of survival for many companies [NicCanna et al. 2021]. Based on this, to achieve the success factors getting technical expertise from offshore places, the pattern regarding the careful selection of a supplier must be considered. As we saw, clients must audit and evaluate the capabilities of their potential vendors [Kommeren and Parviainen 2007], and according to Khan's study, a large-scale client must also track the record of the successful projects of a vendor, ask for SPI Certifications, and also check the turnover rate. Finally, an efficient outsourcing relationship requires enough knowledge of the client management and culture [Khan et al. 2011], which can avoid problems such as the one presented by Noordeloos *et al.* [Noordeloos et al. 2012].

5.1. Limitations

Limitations are inevitable in any research study. However, in this study, several actions were taken to minimize the impacts that could harm the validity of the SLR. In this section, we discuss three types of threats to validity and how we manage it [Wohlin et al. 2012].

Construct Validity: it was ensured through a peer-review process during the SLR in which both the first and second authors were responsible to evaluate each of the studies to build the patterns. Also, whether any authors disagreed with a study, they would discuss it to reach a consensus. In addition, the third author was involved in an independent pattern assessment. *External Validity:* the SLR was conducted through a rigid research protocol validated and developed by the authors. Further, the studies were extracted from five of the most renowned bibliographic databases, as mentioned in section 3. Also, using a search string in those databases ensures a certain degree of generalism regarding the theme studied. *Internal Validity:* the selected studies were evaluated by the first and second authors, when one of them disagreed with a choice, all of them discussed reaching agreement. All authors were also involved in the analysis of the results and redundancy.

6. Conclusion

Outsourcing in distributed large-scale software development can be considered a way out for many companies that aim to access high-skilled suppliers to develop their complex

solutions. However, the nature of those environments requires attention. In this study, we carried out a Systematic Literature Review to better understand how outsourced distributed teams from large-scale contexts handle their operation. Through 23 selected studies from 2006 to 2019, six themes and 15 patterns regarding the operation have emerged. The patterns provide information on how customer and vendor can better coordinate their operation while developing a solution with onshore and offshore teams.

The results and findings of this study will enable researchers and practitioners to better understand the outsourcing nature by providing some solutions to common problems of those types of projects. We hope to evaluate those patterns in a case study of an actual industry project with outsourced teams as future work. Further, we aim to survey outsourced teams from large-scale companies to evaluate better the presence of those patterns and the emergence of new ones. By doing this, we expect to offer solutions to some of the problems encountered in the patterns.

References

Bass, J. M. (2016). Large-scale offshore agile tailoring: Exploring product and service organisations. In *Proceedings of the Scientific Workshop Proceedings of XP2016*, XP '16 Workshops, New York, NY, USA. Association for Computing Machinery.

Batra, D. (2009). Modified agile practices for outsourced software projects. *Commun. ACM*, 52(9):143–148.

Batra, D., VanderMeer, D., and Dutta, K. (2011). Extending agile principles to larger, dynamic software projects: A theoretical assessment. *J. Database Manage.*, 22(4):73–92.

Cho, J. (2007). Distributed scrum for large-scale and mission-critical projects. In *Reaching New Heights. 13th Americas Conference on Information Systems, AMCIS 2007, Colorado, USA, August 9-12, 2007*, page 235. Association for Information Systems.

Cruzes, D. S. and Dyba, T. (2011). Recommended steps for thematic synthesis in software engineering. In *2011 International Symposium on Empirical Software Engineering and Measurement*, pages 275–284.

Daneva, M., van der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., Ajmeri, N., Ramteerthkar, U., and Wieringa, R. (2013). Agile requirements prioritization in large-scale outsourced system projects: An empirical study. *Journal of Systems and Software*, 86(5):1333–1353.

Dingsøyr, T., Fægri, T. E., and Itkonen, J. (2014). What is large in large-scale? a taxonomy of scale for agile software development. In *PROFES*.

Dorairaj, S., Noble, J., and Malik, P. (2012). Understanding team dynamics in distributed agile software development. In *Agile Processes in Software Engineering and Extreme Programming*, pages 47–61, Berlin, Heidelberg. Springer.

Gupta, R. K., Venkatachalapathy, M., and Jeberla, F. K. (2019). Challenges in adopting continuous delivery and devops in a globally distributed product team: A case study of a healthcare organization. In *2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)*, pages 30–34.

Gupta, R. S. (2022). Knowledge management and quality control in software outsourcing projects. In *Research Anthology on Agile Software, Software Development, and Testing*, pages 1484–1510. IGI Global.

Hole, S. and Moe, N. B. (2008). A case study of coordination in distributed agile software development. In *Software Process Improvement*, pages 189–200, Berlin, Heidelberg. Springer.

Hossain, E., Ali Babar, M., and Verner, J. (2009). Towards a framework for using agile approaches in global software development. In *Product-Focused Software Process Improvement*, pages 126–140, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hossain, E., Bannerman, P. L., and Jeffery, R. (2011). Towards an understanding of tailoring scrum in global software development: A multi-case study. In *Proceedings of the 2011 International Conference on Software and Systems Process*, ICSSP '11, page 110–119, New York, NY, USA. Association for Computing Machinery.

Ivarsson, M. and Gorschek, T. (2011). A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering*, 16:365–395.

Khan, S. U., Niazi, M., and Ahmad, R. (2011). Factors influencing clients in the selection of offshore software outsourcing vendors: An exploratory study using a systematic literature review. *Journal of Systems and Software*, 84(4):686–699.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, EBSE Technical Report EBSE-2007-01.

Koch, C., Jørgensen, C., Olsen, M., and Tambo, T. (2014). We all know how, don't we? on the role of scrum in it-offshoring. In *Creating Value for All Through IT*, pages 96–112, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kommeren, R. and Parviainen, P. (2007). Philips experiences in global distributed software development. *Empirical Softw. Engg.*, 12(6):647–660.

Krancher, O. (2020). Agile software development practices and success in outsourced projects: The moderating role of requirements risk. In *Agile Processes in Software Engineering and Extreme Programming*, pages 56–72, Cham. Springer Publishing.

Kussmaul, C. (2010). Onshore and offshore outsourcing with agility: Lessons learned. In *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*, pages 91–106, Berlin, Heidelberg. Springer Berlin Heidelberg.

Lehtinen, T. O., Virtanen, R., Heikkilä, V. T., and Itkonen, J. (2015). Why the development outcome does not meet the product owners' expectations? In *Agile Processes in Software Engineering and Extreme Programming*, pages 93–104, Cham. Springer.

Marinho, M., Noll, J., and Beecham, S. (2018). Uncertainty management for global software development teams. In *11th International Conference on the Quality of Information and Communications Technology*, pages 238–246, Coimbra, Portugal. IEEE.

Marinho, M., Noll, J., Richardson, I., and Beecham, S. (2019). Plan-driven approaches are alive and kicking in agile global software development. In *International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–11, Porto de Galinhas, Brazil. IEEE.

Matthiesen, S. and Bjørn, P. (2017). When distribution of tasks and skills are fundamentally problematic: A failure story from global software outsourcing. *Proc. ACM Hum.-Comput. Interact.*, 1(CSCW).

NicCanna, C., Razzak, M. A., Noll, J., and Beecham, S. (2021). Globally distributed development during covid-19. In *2021 IEEE/ACM 8th International Workshop on Software Engineering Research and Industrial Practice (SER IP)*, pages 18–25.

Noordeloos, R., Manteli, C., and Vliet, H. V. (2012). From rup to scrum in global software development: A case study. In *2012 IEEE Seventh International Conference on Global Software Engineering*, pages 31–40.

Paasivaara, M., Durasiewicz, S., and Lassenius, C. (2008a). Distributed agile development: Using scrum in a large project. In *2008 IEEE International Conference on Global Software Engineering*, pages 87–95.

Paasivaara, M., Durasiewicz, S., and Lassenius, C. (2008b). Using scrum in a globally distributed project: a case study. *Softw. Process. Improv. Pract.*, 13:527–544.

Paasivaara, M., Durasiewicz, S., and Lassenius, C. (2009). Using scrum in distributed agile development: A multiple case study. In *2009 Fourth IEEE International Conference on Global Software Engineering*, pages 195–204.

Paasivaara, M. and Lassenius, C. (2010). Using scrum practices in gsd projects. In *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*, pages 259–278, Berlin, Heidelberg. Springer Berlin Heidelberg.

Paasivaara, M., Väättänen, O., Hallikainen, M., and Lassenius, C. (2014). Supporting a large-scale lean and agile transformation by defining common values. In *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*, pages 73–82, Cham. Springer International Publishing.

Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, page 68–77, Swindon, GBR. BCS Learning & Development Ltd.

Välimäki, A. and Kääriäinen, J. (2008). Patterns for distributed scrum — a case study. In *Enterprise Interoperability III*, pages 85–97, London. Springer London.

Vallon., R., Bayrhammer., K., Strobl., S., Bernhart., M., and Grechenig., T. (2013). Identifying critical areas for improvement in agile multi-site co-development. In *Proceedings of the 8th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE,*, pages 165–172. INSTICC, SciTePress.

Vithana, V. N., Asirvatham, D., and Johar, M. (2018). An empirical study on using agile methods in global software development. In *2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 150–156.

Wieringa, R., Maiden, N., Mead, N., and Rolland, C. (2006). Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. *Requir. Eng.*, 11:102–107.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., and Wesslén, A. (2012). *Experimentation in Software Engineering*, pages 123–151. Springer.