A Catalog of Process Patterns for Academic Software Projects

Caroline Guterres Silva and Lisandra Manzoni Fontoura
Programa de Pós-Graduação em Ciência da Computação, Federal University of Santa Maria (UFSM), Brazil

Keywords: Software Process, Process Patterns, Research-based Software Project, University Partnership.

Abstract: Universities have established partnerships with industry or government through technological innovation projects to develop solutions based on problems presented by institutions. Based on a systematic literature review, we identified a lack of software processes suitable for projects developed in academia. This article proposes a catalogue of process patterns documenting practices that have been successfully adopted in academic projects involving external partnerships. Process patterns describe solutions to problems and challenges commonly found in projects developed in the university environment. We conduct a systematic literature review to identify problems commonly encountered in academic projects and the software practices applied to solve them. Later, with the help of the literature, we deepened the understanding of how the software practices can be used in software projects and documented them as process patterns. As a result, we have identified thirteen problems and documented ten process patterns describing possible solutions related to the problem. Eight researchers with experience in software projects in partnership with academia participated in the validation. The validation showed that the proposed process pattern catalogue describes relevant solutions to the problems and is applicable to the academic context.

1 INTRODUCTION

Due to the growing relevance of knowledge and research for economic development, the role of the university needs to be reviewed, as well as focusing on teaching, research, and extension activities; it has the mission to assist in economic development. For this, it is necessary to bring the university closer to other sectors of the economy, establish partnerships with industries and the government, aiming to propose innovative solutions to the problems found in these organizations (Damoc, 2017).

However, it should point out that there are differences between these institutions in terms of culture and goals. In a simplistic view, the government is oriented towards economic development, universities towards knowledge, and industries oriented profit, representing three different environments (Mineiro et al. 2019).

In academic projects in collaboration with other institutions, the chosen practices also need to be suited to the project environment. It appears that there is a challenge, considering the environment of collaboration between institutions, which makes it necessary to identify a software development approach that benefits both parties involved.

According to a systematic literature review conducted in 2020 (Silva et al., 2020), we see a lack of software processes suitable for research projects developed in academia in partnership with industry or government. The research showed that there are cultural and organizational differences between institutions. These differences must be considered when choosing appropriate practices for joint software development. Several articles report experiences or case studies in this context, but the knowledge is not systematized (Brondani et al., 2019), (Dias et al., 2013), (Cereci and Karakaya, 2018), and (Andrade et al. 2017).

This work proposes a catalog with the recommendation of software development practices, described as process patterns, for use in projects developed in academia in partnership with industry or government. In this way, an analysis was carried out in the literature seeking to identify the problems and challenges identified in the university environment in the context of software development in partnership with other institutions, and solutions were linked to these problems, allowing, later, the documentation of process patterns.

The remainder of this paper is structured as follows. Section 2 describes the method used to

Silva, C. and Fontoura, L.
A Catalog of Process Patterns for Academic Software Projects.
DOI: 10.5220/0011090700003179
In *Proceedings of the 24th International Conference on Enterprise Information Systems (ICEIS 2022) - Volume 2*, pages 175-182
ISBN: 978-989-758-569-2; ISSN: 2184-4992

build and validate the software process catalog. Section 3 explains how we document the process patterns. In Section 4, we describe the validation of the work. Section 5 summarizes our approach, its results, our contributions, and future work.

2 RESEARCH METHOD

The software process patterns aim to assist the development team in making decisions regarding the choice of best software practices, allowing better management of resources, activities, and artifacts that involve software projects developed at the academy in partnership with other institutions. For the elaboration of the catalogue of process patterns, three activities were followed (see Figure 1), which are:

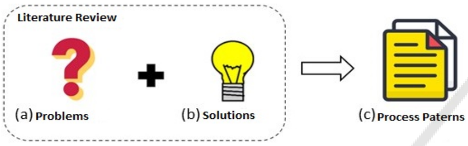

Figure 1: Phases of the research method.

a) Identification of problems in academic projects: identification of problems related to software projects developed in universities in partnership with other institutions from the literature;

b) Association of software practices successfully adopted in real projects to address the problems identified previously;

c) Documentation of the process pattern associating the problems found with the solutions described in the literature as software practices;

d) Validation of the work through interviews with researchers and professionals who develop software in universities in partnership with other institutions.

3 CATALOG OF PROCESS PATTERNS

This section details the activities we follow to document process patterns.

3.1 Identification of Problems in Academic Projects

Based on the specialized literature, we identified problems faced in software projects developed in universities with partnerships with other institutions. We searched the following databases: IEEE Xplore, ACM, Scopus, and academic Google. The inclusion criteria for the works were: having been published between 2011 and 2020 and describing real experiences.

From the analysis of the works, we list thirteen problems that were cited by the works that are described below.

High Turnover (p1): Brondani et al. (2019) report that because the software teams are formed by undergraduate or graduate students, they remain in the project for an average of two years, resulting in a high turnover in the team. Cereci and Karakaya (2018) point out the turnover among undergraduate students as a critical problem that causes the loss of knowledge and some divergences from the project plan.

Part-time Availability (p2): several works report that academic teams are formed by undergraduate and graduate students (Brondani et al., 2019; Dias et al., 2014; Cereci and Karakaya, 2018; Andrade et al., 2017). These students need to split time between the course and their research, participating in the projects part-time that can negatively affect the progress of the projects.

Communication Problems (p3): the availability of part-time impacts the reduction in the frequency of meetings. Cereci and Karakaya (2018) comment that the less frequent the project meetings, the less information researchers have about the status of the project and the progress of other researchers. The hierarchical communication system that exists in some industries or government environments makes communication between the team and stakeholders difficult, causing the absence or delay in responding to questions raised by the project team (Brondani et al., 2019; Dias et al., 2014; Andrade et al., 2017).

Unclear Division of Responsibility (p4): Cereci and Karakaya (2018) highlight that university teams are dynamic, so the roles responsible for performing each activity are not well established. Another issue is that the experience level of the university team is lower compared to industry teams.

Lack of Familiarity with Project Domain (p5): Brondani et al. (2019) and Andrade et al. (2017) describe difficulties related to understanding the business domain and the activities carried out by partner industries. This difficulty results in a demand for additional time to understand organizational processes before defining requirements.

Workers with Different Skills (p6): an academic team consists of individuals with different capacities. According to Cereci and Karakaya

(2018), undergraduate students have less experience and know-how than industry professionals.

Complexity of Solutions (p7): partner institutions look to universities to develop complex and innovative solutions that require research to solve problems (Brondani et al., 2019). Crawrford (2002) point out that as complexity increases, effective communication becomes critical to project success.

Unstable Requirements (p8): innovative software development makes it difficult to define a stable set of software requirements. The trend is that the requirements evolve throughout the project as the results are obtained in the research developed. This iterative development implies changes in the specifications (Brondani et al., 2019; Dias et al., 2014; Cereci and Karakaya, 2018; Andrade et al., 2017).

Little Contact with Customers/users (p9): Cereci (2018) describes that, in some academic projects, the end-users do not participate in the project, not interacting with the team. In other projects, end-users only engage in requirements elicitation activities and provide feedback on progress. Some authors cite the difficulty in finding users to perform acceptance tests of the developed software (Andrade et al., 2017; Dias et al., 2014).

Difficulty in Marketing Products (p10): according to Dias et al. (2014), product marketing is an issue for academic projects. This problem is justified by the fact that the partner institution is interested in obtaining the exclusive commercial right of the product (Dias et al., 2014; Cereci and Karakaya, 2018; Andrade et al., 2017).

Difficulty in Publishing Research Results Due to non-Disclosure Agreements (p11): Andrade et al. point out that in projects with other institutions, non-disclosure agreements (NDAs) are most often signed to protect the knowledge of hardware and software made available during the execution of the project, as well as information obtained in visits or meetings. These agreements may limit the publication of research results, which are needed success indicators for university projects.

Divergent Visions/Goals (p12): Dias et al. (2014), Cereci and Karakaya (2018), and Andrade et al. (2017) report that the university and the industry have divergent objectives with the project being developed. Andrade et al. (2017) report that, in the eyes of the industry, the academy has only theoretical knowledge, while, for the academy, the industry has only practical knowledge. These different views can lead to conflicts that need to be resolved for projects to achieve their goals.

Feedback Delay by Delivery Stakeholders (p13): Andrade et al. (2017) describe that many projects lack adequate and timely feedback from stakeholders, thus causing a delay in software development.

3.2 Documentation of the Process Pattern

After mapping the problems encountered in software projects in the context of the university, we identified software development practices that have been used successfully to reduce the impact of each issue. We rely on the literature and consider only works that describe experiences of software projects developed in academia. For the description of each software process pattern (PP), we use the following properties: purpose, problem description, and solution.

a) PP1: Develop Business Process Diagrams

Purpose: helps the team understand the business domain related to the software that will be developed. Creating a business process diagram makes it easier to understand an organization's business processes and helps the team understand, specify, and prioritize software requirements.

Problems (p1,p8): the software team needs to understand the business domain, as this domain influences the understanding and specification of software requirements. Understanding the business domain is a complex activity for developers, as they are unfamiliar with it. In many situations, some manuals provide domain information but are easy to understand, which complicates the understanding and learning of organizational processes for the team. The lack of knowledge of terms related to the business process impacts the communication between the development team and the client. Brondani et al. (2019) cite in their work the difficulties encountered in understanding military doctrines for the development of a tactical virtual simulator for military training.

Solution: for the elaboration of business process diagrams, the team collects information about this domain and documents specific terms to facilitate the tasks related to requirements management (Prieto-Díaz, 1990). Brondani et al. (2019) describe that the adoption of business diagrams, in addition to helping the team to understand the project domain, improved the communication flow between the team and the stakeholders.

b) PP2: Iterative and Incremental Process

Purpose: a life cycle for the software process that uses short development cycles called iterations. At the beginning of the iteration, the requirements that add the most value to the customer's business are prioritized. At the end of the iteration, an operational version is delivered to the client.

Problem (p12): it is unrealistic to define a set of requirements at the outset of the project and to expect these requirements to remain unchanged during development. In academic projects, there is more uncertainty than in industrial projects, which causes difficulty in defining stable requirements. Many requirements depend on the results of research carried out during the project.

Solution: the iterative and incremental approach allows improving scenarios in which changes are inevitable and, thus, controlling the resulting risks (Schwaber and Sutherland, 2020). Beck (2000) and Fowler (2004) describe that development should be carried out in short, iterative cycles (1 - 4 weeks), in which the result of the next iteration is an increment of improved work. In this way, advances obtained in the research carried out in the project can give rise to requirements that will be implemented in subsequent iterations.

c) PP3: Definition of Individual Research Goals

Purpose: individual research must be defined from the project objectives, aiming to delimit the research context of each member of the project. Personal projects may lead to end-of-course work, master's dissertations, and doctoral theses.

Problem (p2): the more innovative the project, the more complex the solutions are, and in many cases, they are required to test various alternatives. Therefore, several personal projects can be defined from an innovative academic project.

Solution: due to the complexity and the different possible alternatives for solving the problem, research subjects with clear objectives can be isolated so that they can generate research projects to be developed by researchers or undergraduate and graduate students.

Brondani et al. (2019) describe in their study as a solution to this problem the identification of research problems that may be explored in final papers or master's dissertations carried out under the supervision of a researcher in the area. The authors suggest that if the research results are successful and achieve the desired objectives be incorporated into the software developed in the research project. It should be noted that more than one research project can be conducted simultaneously to analyze alternatives to achieve a goal.

d) PP4: Refactoring

Purpose: elimination of unreadable or unnecessary code.

Problem (p13): as identified in the systematic literature review and related work, academic teams are made up of people with different skill levels who stay on the project for a while (high turnover). Therefore, the lack of experience of some team members impacts the production of understandable and easy-to-maintain software.

Solution: Brondani et al. (2019), describes that before milestones, the team can plan tasks for code refactoring, aiming to improve readability and documentation, in addition to removing unnecessary lines of code.

e) PP5: Define Persons Responsible for Communication between Teams

Purpose: define one person on the academic team and one on the customer team as a point of contact for clarifying questions or providing information.

Problems (p1, p9): Brondani et al. (2019) describe that hierarchical communication can cause a delay in software development since a question and its answer need to go through several levels in this communication structure.

Other works cite the lack of involvement of the user or the client in academic projects (Dias et al., 2014; Cereci and Karakaya, 2018; Andrade et al., 2017). Dias et al. (2014) describe that, in most cases, the client's participation occurs only at the end of the project.

Solution: to simplify communication between teams, this pattern suggests defining one person on the academic team and another on the partner team as a point of contact for resolving questions.

They are responsible for receiving the information and making it available to team members interested in that information. They also need to meet the needs of the partner team to provide information and answer questions.

f) PP6: Full-time Contract Workers

Purpose: hire full-time professionals for the development team. These professionals are responsible for maintaining project history and passing on knowledge to team members, minimizing the effects of turnover among team members.

Problem: academic teams have many members who are university students and stay on the project for the duration of a course. As such, these students balance the teaching activities with the project activities.

Furthermore, according to Andrade et al. (2017), project members often participate in research to produce scientific publications, participating in conferences looking for solutions to proposed challenges. Other works explore the high turnover of the team and the participation of a large part of the team on a part-time basis.

Solution: to minimize the risks of high turnover and a large number of part-time and inexperienced workers, Brondani et al. (2019) and Andrade et al. (2017) suggest hiring experienced full-time professionals to manage team members and retain the knowledge of the project.

g) PP7: Collaborative Work

Purpose: allow a collaborative software development environment to complement individual skills and knowledge, minimizing failures and, consequently, producing better results concerning the development process of a software product.

Problems (p7, p8, p13): the high turnover of team members can lead to a loss of knowledge regarding advances made in research challenges and on knowledge of the business domain and software requirements.

The low frequency of meetings and the part-time dedication of a large part of the team can generate communication problems between project members and between them and the client.

The university teams consist of many undergraduate and graduate students, which results in divergent skill levels among team members and a high turnover rate as they stay on the project while they take their courses.

Solution: according to Brondani et al. (2019), team members can learn from each other, sharing information about the business domain and requirements and, in this way, disseminating knowledge about the system being developed. Knowledge sharing between team members can be encouraged by using practices such as pair programming and code review.

h) PP8: Review Material before Publication

Purpose: establish agreements regarding the objectives and visions of the different institutions involved in the project, and define a process for reviewing publications.

Problems (p4, p5): considering the project development environment in partnership with academia, Andrade et al. (2017) indicate that universities and industry have differing interests and attitudes regarding the publication of project results. The industry does not show interest in publishing scientific papers due to the disclosure of strategic information. Academia needs publications to improve its productivity indicators.

Solution: Andrade et al. (2017) propose to solve this problem the definition of an agreement in which the academy must previously send the works and articles produced under the project for review by the partner. Only after approval, the academy may make project-related content available to third parties. This mechanism guarantees the partner that only information authorized by it will be published.

i) PP9: Regular Meetings

Purpose: organizing regular meetings improves communication among team members, helps in establishing and tracking goals and objectives.

Problems (p1, p5): it appears that as the team meets less frequently, employees have less information about the status of the project and the progress of activities, affecting communication between those involved in the project and control (Cereci and Karakaya, 2018).

Solution: the adoption of regular meetings between project team members allows for an overview of the project's progress, continuous feedback, and the reporting of impediments that affect the performance of activities by team members (Andrade et al., 2017). The frequency will depend on the team availability, but the idea is that they are in very short intervals.

j) PP10: Project Plan Definition

Purpose: the project plan outlines the project goals, sets out the responsibilities of each partner, deadlines, and budgets for the development of the project. The project plan establishes warranty terms concerning the software product.

Problem (p1, p2, p3, p5, p11): Changes can result in software development delays, causing additional costs (Andrade et al., 2017).

Solution: The work plan is a formal document that outlines the project's goals, specifies the tasks, and indicators for monitoring the goals. Once drawn up, it must be approved by all parties (Andrade et al., 2017). It is verified that, through a work plan, it is possible to establish the project planning, avoiding possible misunderstandings and assisting in the communication of those involved in the project.

The plan should describe the rules that govern the partnership agreement, such as clauses regarding confidentiality, publication of results, percentage of royalties, rights, and duties of each institution.

4 STUDY VALIDATION

A case study is carried out to study a single entity or phenomenon in a given time frame. Case studies help to assess the benefits of process and tools and provide a cost-effective way to ensure that process changes predict desired outcomes (Kitchenham and Pfleeger, 2002).

According to Wohlin et al. (2003), if the effect of a process change is widespread, a case study is more suitable. The effect of change can only be assessed at a high level of abstraction because process change includes smaller and more detailed changes throughout the development process. Based on the literature (Wohlin et al., 2003; Runeson and Höst, 2009), we define the following steps:

1) Conception and Project

In this step, we define the objectives, the research hypotheses, how such hypotheses would be evaluated and the results obtained.

This case study aims to verify the applicability of the proposed process patterns in software projects developed in academia with industries and/or government. Experienced developers were invited to assess the applicability of the process patterns considering academic projects with external partnerships already developed by them.

We defined a set of hypotheses based on the documentation of process patterns to guide the validation of this study. For each problem (p) associated with the proposed process pattern (PP), we define a hypothesis (h), as shown in Table 1. The hypotheses are:

[h1] Adopting business process diagrams facilitates team communication with stakeholders;

[h2] The elaboration of business diagrams helps in understanding the application domain;

[h3] Using an iterative and incremental process can minimize the number of requirements change requests;

[h4] Individual research projects can contribute solutions to complex systems;

[h5] Constant code review can improve code readability and documentation and thus minimize rework;

[h6] The appointment of a person responsible for the flow of information can solve communication problems;

[h7] Those responsible for the communication flow help to resolve doubts whenever necessary, increasing the client's contact with the team;

[h8] Full-time workers assist in managing team members and managing the history and sharing of information, allowing for continuity in the project;

[h9] Full-time worker management allows more experienced collaborators to share knowledge about the project domain;

[h10] Collaborative work allows everyone to have information regarding the software project;

[h11] From collaborative work, it is possible to share information from the application domain, keeping the team with the same level of knowledge regarding the development of the project;

[h12] Collaborative work allows more experienced professionals to share their knowledge in a way that empowers employees and improves their skill level;

[h13] Reviewing reports and articles before publication allows for analysis and authorization of published information;

[h14] The agreement regarding the publication of information allows establishing guidelines on the results of the software product, considering each institution;

[h15] More frequent meetings help communication in the project;

[h16] Periodic meetings facilitate contractual policy issues;

[h17] Performing the software work process definition allows for effective communication in the project;

[h18] The problem regarding the division of responsibilities can be solved by defining a work plan;

[h19] Workplan helps in the context of the complexity of software solutions;

[h20] In the work plan, it is possible to define questions regarding the commercialization of products;

[h21] Contractual guidelines can be defined through a work plan.

2) Preparation for data collection:

We used a questionnaire, prepared in Google Forms, as a data collection method, thus allowing a better understanding of the problem situation based on the participant's experience. The questionnaire shows information about the description of each pattern, explaining the related problem, the suggested solution, and, when available, the dynamics of the process pattern's execution. Then the assumptions are shown, and the participant must assess the extent to which the hypothesis is satisfied by the proposed process pattern.

Table 1: Association of hypotheses (h) to process patterns (PP) and problems (p).

PP1		**PP2**	**PP3**	**PP4**	**PP5**		**PP6**		**PP7**			**PP8**		**PP9**		**PP10**				
p1	p8	p12	p2	p13	p1	p9	p7	p10	p7	p8	p13	p4	p5	p1	p5	p1	p11	p2	p3	p5
h1	h2	h3	h4	h5	h6	h7	h8	h9	h10	h11	h12	h13	h14	h15	h16	h17	h18	h19	h20	h21

The target audience is classified as professionals in the software development area who participate or have participated in a project that involves collaboration between academia and industry and/or government and has experience in this context.

We selected participants through convenience sampling, looking for projects related to the research topic on the web and in projects known to the authors of the work. Based on these selection criteria, we invited ten professionals and, of these, eight responded.

3) Collection:

The questionnaire describes twenty-one closed questions and a descriptive open question. We used Likert Scale ranging from strongly agree (5) to strongly disagree (1) to evaluate each closed question. We selected participants through convenience sampling, looking for projects related to the research topic on the web and in projects known to the authors of the work. Based on these selection criteria, we invited ten professionals and, of these, eight responded. The questionnaire describes twenty-one closed questions and an open question.

4) Analysis:

In this step, the information collected was analyzed in order to determine the applicability of the process pattern to the described problem. Section 4.1 highlights the results of this analysis.

4.1 Results and Discussions

For each hypothesis defined in Section 5, participants indicated whether or not they agree with the hypothesis. We defined the following options: 5 - strongly agree, 4 - partially agree, 3 - neither agree nor disagree, 2 - partially disagree, and 1 - strongly disagree. Figure 2 shows the frequency distribution of participants' responses.

Regarding the hypotheses (h4, h6, h7, h8, h9, h10, h11, h12, h13, h14, h17, and h18), the participants agree with the solution defined through the process pattern for the associated problem. For hypotheses h3, h5, h15, h16, h19, h21, a minority of participants remained neutral in the evaluation of the process. It is noteworthy that the others strongly or partially agreed with the proposed pattern.

For h20, one of the participants strongly disagreed that the Project Plan Definition (PP10) can be a solution to the problem "Difficulty in marketing products" (p3). In this specific case, we complement the process pattern by defining the adoption of a contract with the rights of each institution, specifying the authors involved in the development of the software, and defining issues regarding intellectual property and commercial use rights for the partner institution. After collecting and analyzing the participants' responses, we concluded that the proposed process pattern describe good solutions to the main problems encountered in software development projects carried out at universities with external partnerships.

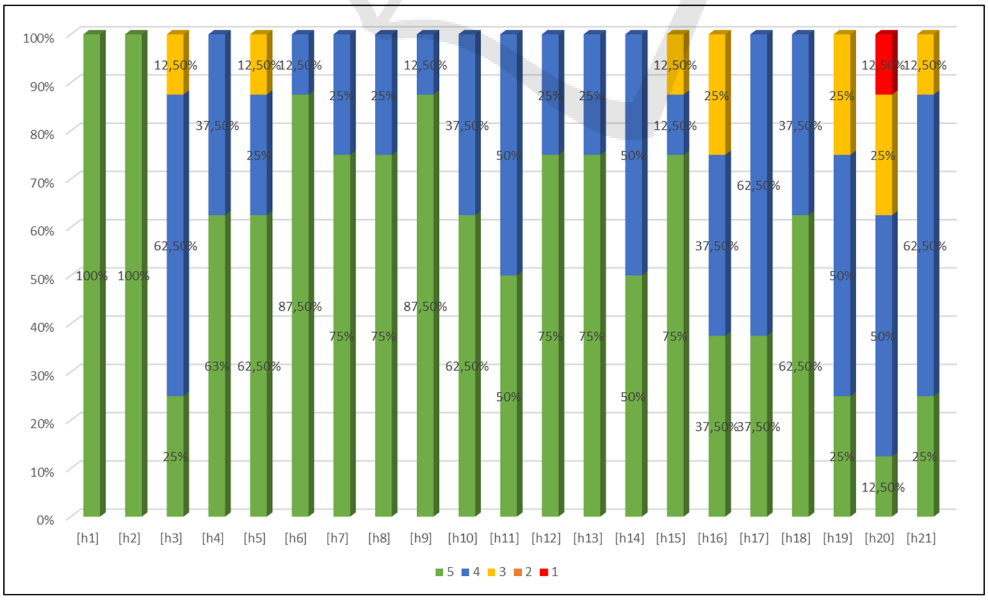

Figure 2: Frequency analysis of participant's responses.

5 CONCLUSIONS

In this work, we searched the literature for studies that identify problems present in software projects developed in universities with external partnerships. After identifying the problems, we look for solutions successfully applied to real projects. Then, we document these solutions as process patterns.

For validation, we define hypotheses that aim to assess the applicability of the process pattern to solve each associated problem. These hypotheses were evaluated by experienced professionals who have participated in academic projects with external partnerships. We had participants from the three institutions involved. The results obtained in the evaluation were positive considering that the participants agreed with the proposed process patterns. There is a deficiency in the software process literature that considers the characteristics of software development projects in universities with external partnerships. These projects have different features from projects developed by the industry, requiring processes tailored to this reality.

We cite as threats to the validity of the study, the fact that the participants were invited by the researchers to participate in the experiment. In addition, the number of participants could have been higher. However, we highlight the difficulty of finding participants with the desired profile.

We requested that the hypotheses be evaluated based on experiences in a real software development project developed at universities with external partnerships but we do not have information about the projects considered in the evaluation.

As future work, we will experiment with the proposed process patterns in a software project developed at our university.

ACKNOWLEDGEMENTS

We thank the Brazilian Army and its Army Strategic Program ASTROS for the financial support through the SIS-ASTROS GMF project (898347/2020).

REFERENCES

Andrade, R. M. C., Lelli V., Castro, R. N. S. and Santos, I. S. (2017). Fifteen years of industry and academia partnership: lessons learned from a brazilian research group. IEEE/ACM 4th *International Workshop on Software Engineering Research and Industrial Practice* (SER&IP), pp. 10-16.

Beck, K. (2000). *Extreme Programming Explained.—Embrace Change*. Addison-Wesley Professional.

Brondani, C., Mello, O. and Fontoura, L. (2019). A case study of a software development process model for SIS-ASTROS". In *The 31st International Conference on Software Engineering and Knowledge Engineering*, Lisboa, pp 600-605.

Cereci, I. and Karakaya, Z. (2018). Need for a software development methodology for research-based software projects. In 3rd *International Conference on Computer Science and Engineering*, Sarajevo: pp. 648-651.

Crawrford, L. (2002). *Profiling the competent project manager*. In: Slevine, Cleland and Pinto (Edts), the frontiers of project management research (pp. 151-176). Newton Square, PA: Project management institute.

Damoc, A. I. (2017). The strategic role of partnerships between universities and private corporations as a driver for increasing workforce competitiveness in a global economy. *Proceedings of the International Conference on Business Excellence*.

Dias, D. M. P., Kodikara, N. D. and Jayawardena, M. (2013). The need for novel development methodologies for software projects in universities: a Sri Lanka case study. In: *International Journal of Future Computer and Communication* v.1: 494–98.

Fowler, M. (2004). Using agile software process com offshore development. https://martinfowler.com /articles/agileOffshore.html.Acesso em: Julho, 2021

Kitchenham, B. and Pfleeger, S. (2002). Principles of survey research part 2: designing a survey. *SIGSOFT Softw. Eng. Notes,* pp. 18–20.

Marijan, D. and Gotlieb, A. (2020). Industry-Academia research collaboration in software engineering: The Certus model. Information and Software Technology. 132. 106473. 10.1016/j.infsof.2020.106473.

Mineiro, A., Souza, D. L., Vieira, K. C., Castro, C. and Brito, M. J. (2019). Da Hélice Tríplice A Quíntupla: Uma Revisão Sistemática. Revista Economia & Gestão. 18. 77-93.

Prieto-Díaz, R. (1990). Análise de domínio: uma introdução. *SIGSOFT Softw. Eng*. Notes 15, 2 (abril de 1990), 47–54.

Runeson, P. and Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering, *Empir. Softw. Eng.,* vol. 14, no. 2, pp. 131– 164, 2009

Schwaber, K. and Sutherland, J. (2020). *O guia do scrum: o guia definitivo para o Scrum as regras do jogo*.

Silva, C. G., Filho, E. L. F, Fontoura, L. M. (2020) Software Processes Used in University, Government, and Industry: A Systematic Review. In *The 22nd International Conference on Enterprise Information Systems*. pp. 314-321

Wohlin C., Höst M. and Henningsson K. (2003). *Empirical research methods in software engineering*. Springer.

Yin. R. K. (2005). Estudo de caso: planejamento e métodos. 3 ed., Porto Alegre: Bookman.