Association for Information Systems
AIS Electronic Library (AISeL)

Research Papers | ECIS 2017 Proceedings

Spring 6-10-2017

A SURVEY OF FAILURES IN THE SOFTWARE DEVELOPMENT PROCESS

Rita Marques
IST/INOV, Avenida Rovisco Pais 1, 1049 Lisboa, Portugal, rita.marques@tecnico.ulisboa.pt

Gonçalo Costa
IST/INOV, Avenida Rovisco Pais 1, 1049 Lisboa, Portugal, goncalo.m.da.costa@tecnico.ulisboa.pt

Miguel Silva
IST/INOV, Avenida Rovisco Pais 1, 1049 Lisboa, Portugal, mms@tecnico.ulisboa.pt

Pedro Gonçalves
Xpand IT, Rua de Fornelos 77, 4900-709 Viana do Castelo, Portugal, pedro.goncalves@xpand-it.com

A SURVEY OF FAILURES IN THE SOFTWARE DEVELOPMENT PROCESS

Research paper

Marques, Rita, IST/INOV, Avenida Rovisco Pais 1, 1049 Lisboa, Portugal, rita.marques@tecnico.ulisboa.pt

Costa, Gonçalo, IST/INOV, Avenida Rovisco Pais 1, 1049 Lisboa, Portugal, goncalo.m.da.costa@tecnico.ulisboa.pt

Mira da Silva, Miguel, IST/INOV, Avenida Rovisco Pais 1, 1049 Lisboa, Portugal, mms@ tecnico.ulisboa.pt

Gonçalves, Pedro, Xpand IT, Rua de Fornelos 77, 4900-709 Viana do Castelo, Portugal, pedro.goncalves@xpand-it.com

Abstract

Software development is one of the most important worldwide industries and continues to grow. To deal with this challenge, organizations are adopting ever more tools and methodologies. However, software development projects are still failing in meeting time, budget and functional requirements.

This study provides insights on the failures faced by software development organizations regarding their processes, the reasons leading to these failures, and initiatives taken to cope with them. A re-search methodology was used to gather and compare results from a literature review and semi-structured interviews. We learnt that there are more failures in Management activities, although they were not often report-ed, while failures in Requirements Engineering and Software Testing are less in number but more fre-quently reported. Lack of communication, lack of time for improvements and appropriate testing, and poor requirements and functionalities specification were the mostly reported failures. Furthermore, we learnt that organizations are not implementing any initiative to address these failures, although they suggested solutions.

Keywords: Software development process, failures, literature review, empirical study.

1 Introduction

Software development is one of the most important worldwide industries (Xu & Brinkkemper 2007) which is rapidly evolving due to the increasingly dynamic business environments, fast technologies development and increase of end user demands (Marchenko & Abrahamsson 2008). To cope with this, organizations are adopting new tools and methodologies towards a more cost-efficient, flexible and faster process that can produce higher quality products (Overhage et al. 2011). Some recent and popu-lar approaches focus on agile methodologies, such as Scrum (Schwaber 1995) and Extreme Program-ming (Beck 2000), which are combined with software tools.

Nevertheless, too often software development projects fail in meeting those goals, generating a low quality artifact or exceeding budgets (Dubois & Tamburrelli 2013). Several reasons can be leading to this, some of them related to the correct implementation of the software development process adopted in the organization.

We reviewed the literature and conducted interviews to answer some relevant research questions:

- **RQ1:** Which activities are being implemented by software development organizations?
- **RQ2:** Which are the reasons leading to failures in the software development process?
- **RQ3:** How are organizations addressing these failures?

The remainder of this paper is structured as follows. In the next section the research background is presented. After explaining the research methodology used to conduct this study, data collected during the interviews is analysed and discussed. Then, the authors discuss these results in comparison with evidence found in prior literature and provide answers to the established research questions. Limitations of this study are presented, and the paper closes with a conclusion and guidance for future work.

2 Background

A software process can be defined as "a set of activities, methods, practices and transformations that people use to develop and maintain software and the associated products" (Zahran 1998). Organizations related to the software development should follow some kind of process, which will allow the product to be delivered on time and with sufficient quality to satisfy both customer and end users (Sánchez-Gordón & O'Connor 2016).

To improve their understanding, software development processes are usually represented in a simplified form using software process models. Over the years, a wide range of models have been proposed, each with variable strengths, weaknesses and degree of success. More recently, the limitations of traditional models such as Waterfall (Royce 1970) are being addressed by more flexible agile models such as Scrum (Schwaber 1995). Some organizations developed their own software processes in-house, since no other proposal aligns with their needs (Sommerville 2015).

Nevertheless, and despite this diversity of models, organizations are still not following their software processes rigorously nor uniformly (Fitzgerald 1998), which is a potential root for software project's failure. Despite not being the only industry facing project failure, software development projects are more likely to miss deadlines and not meeting scope (Varajão et al. 2014). According to Standish Group's CHAOS report, most projects run during 2015 were either failed (19%) or challenged (52%) (Standish Group 2015). In fact, the major issues conducting to project failure are human, not technological (DeMarco & Lister 2013).

Process standards such as ISO 9001 (International Organizational for Standardization 2015) and the Capability Maturity Model Integration (CMMI) (Software Engineering Institute 2010) are an option for organizations to improve their processes concerning, for example, communication and quality. These encapsulate a set of best practices, which can include definitions for specification, process support tools or documentation. However, having a certification for a standard does not mean having a software product with quality; it only ensures that organizations follow formalized processes (Sommerville 2015).

Typically, small and very small enterprises, which constitute many of the software companies (Laporte et al. 2008), do not adopt these standards since they perceive them as being oriented towards large organizations, and even these only seek for certification when they intend to work with bodies that establish such requirement. Other reported reasons include high implementation costs and a process that is very complicated, with several bureaucracy and tedious documentation (Basri & O'Connor 2010) (Staples et al. 2007).

Practitioners do seem to recognize these failures' existence, a problem that is widely discussed by recognized organizations and published in relevant sources (Florentine 2016; Cramm 2001; Bucero 2007; Shaker 2010). Notwithstanding this acknowledgements' importance, such articles were not scientifically validated, thus were not considered for the study reported in this paper.

The objective of this study was to provide insights on the failures faced by software development organizations regarding the implementation of their processes, the reasons leading to poor implementation, and initiatives developed by organizations to cope with these failures. To achieve this objective, the opinions and perspectives of people working in Portuguese organizations were mapped against the existent literature.

3 Research Method

A qualitative research methodology was adopted to identify which software development activities are not being implemented correctly, which are the reasons leading to this poor implementation, and what are organizations trying to do to solve these failures. This decision was based on the fact that this methodology allows to understand and explain social phenomena (such as behaviours and interactions) based on qualitative data such as interviews (Myers 1997).

More specifically, this study follows a qualitative exploratory research methodology (Janesick 2000) that adapts well to our objectives: learn the failures in the software development process using the research literature and results collected from interviews.

This methodology, which is illustrated in *Figure 1*, was developed based on three stages: research design, data collection and data analysis. The remainder of this section is devoted to the description of these phases and discussion on how they were applied in practice.

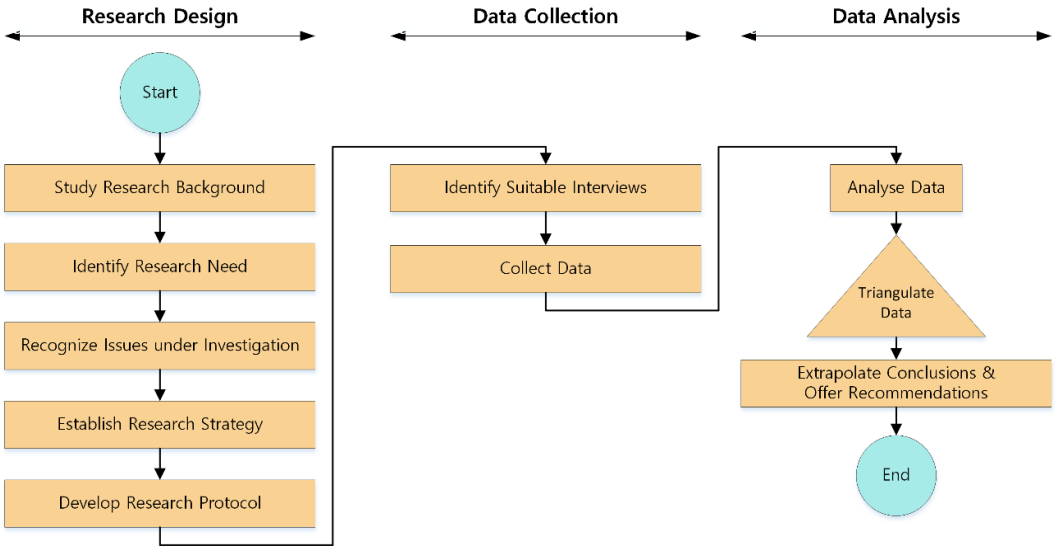

Figure 1. Research Methodology

3.1 Research Design

During the research design, the authors studied the background of the research area and identified the research need. Following this, decisions were taken regarding the research questions (already presented, see section 1) and relevant data to collect and analyse. In the end, a research strategy was established and the research protocol developed.

3.2 Data Collection

In the second stage, data collection, the researchers performed the work necessary to collect the data, following two of the main principle methods of conducting exploratory research: by interviewing "experts" in the subject and searching literature (Saunders et al. 2008).

The first step was to identify suitable interviewees. Five practitioners, whose characterization is depicted in *Table 1*, involved or related to software development processes were identified. Criteria for selection included the usage of a specific software management tool working for a relatively large organization, and the availability to schedule the interviews. From now on, each of the interviewees will be referenced in this document by his/her ID.

ID	Position/Role	Education and Training	Experience	Organization Sector	Processes
I1	Delivery Manager Responsible for Process Implementation	Informatics and Information Systems Management Scrum Certifications (including Scrum Master)	8 years in project management 11 years in software development	IT Telecommunications	Scrum RAD
I2	IT Director Project Manager	Computer Science Engineering	15 years in project management	Telecommunications	Waterfall
I3	Responsible for Process Implementation	Computer Science Engineering	8 years in project management 23 as software developer	IT Telecommunications	Proprietary Scrum
I4	Quality Director	Enterprises organization and management Systems Integrated Management	24 years as quality manager and process certification	IT	Proprietary CMMI
I5	Team Leader Scrum Master Project Manager	Background on Computer Science Engineering	3 years in project management 11 years in software development 8 years in gaming industry	Game Industry	Scrum Kanban

Table 1. Interviewees' characterization

Quantitative interviews are "the most common and one of the most important data gathering tools in qualitative research" (Myers & Newman 2007). In this study, semi-structured interviews were used because, despite the existence of a base guideline, they allowed for flexibility and improvisation, clearing the path for the interviewer to explore emerging lines of research. The questions used in the guideline can be found in *Figure 2*.

Warm up

1. Briefly tell us about your position/role in the company.
2. Your job is, certainly, guided by some work processes related to development, operations and support of software. Can you give us some examples?

Work Processes

3. Choose one of the previously mentioned processes. Draw and briefly explain it.
4. Is that process flawlessly followed by your team, complying with all the steps identified in the previous scheme?
    a. If not, which are the main flaws?
5. Are your team members rewarded for their productivity?
    a. If yes, do you think that motivates them? How?
    b. If not, do you think it would motivate them? How?
6. Imagine your team started failing deadlines and decreasing performance.
    a. Which problems in the process could lead to this?
    b. How would you try to address those issues?

Figure 2. Interview questions

A mapping between research questions and interview questions is presented in *Table 2*. The introductory questions 1 and 2 were not considered in this mapping since they were used to perform the interviewee characterization and were not directly used to answer any research question.

Research Questions	Interview Questions
RQ1	3, 4
RQ2	4a), 6a)
RQ3	5a), 5b), 6b)

Table 2. Mapping between research and interview questions

All interviews were performed by two researchers with different responsibilities. While one interviewer conducted the conversation with the interviewee, the other one took notes on the computer, using the interview guide as the data collection template. Immediately afterwards, those notes were revised by both researchers to create a final script.

Interviews were held between June 2016 and October 2016 and took between 15 to 30 minutes, due to their partial open form. Two interviews were remote (one by Skype and another by phone) and the remaining were performed in the interviewees organization's office.

Literature related to this study was searched using Google Scholar. Keywords were selected as follows: ("software development" AND ("process" OR "methodologies" OR "methods" OR "activities") AND ("failure" OR "poor adoption")). The authors of the papers that were not available online were contacted to request the documents, but few replied to the research team. The content of the available papers was examined and only those relevant for this study were maintained. References of these papers were analysed to find other relevant papers.

Initial criteria for inclusion was the presence of content that supported answers to the research questions, namely empirical studies that addressed the failures in the software development processes, presented reasons for the poor implementation of some activities, and/or discussed initiatives taken by organizations to overcome implementation challenges.

Nevertheless, these criteria returned few results. So we extended the search to papers that discussed reasons leading to software project failure, i.e. when projects do not meet the established criteria for success (Kaur & Sengupta 2011). But the focus was still on failures related to practitioners' characteristics and relationships and the development process itself, which are less entrenched in the project, thus less hard to change. Papers addressing the general adoption of specific methodologies and standards as well as the implementation of software process improvements were discarded.

3.3 Data Analysis

Data from interviews and literature search were interpreted and categorized when needed, for example, to create the list of the main reasons leading to poor implementation of software development processes for RQ2. Afterwards, data from both sources were triangulated for validity of the results. Triangulation comprises the use of multiple data sources in one study to ensure that data is providing valid and consistent information (Saunders et al. 2008).

A classification scheme based on software development process areas (Lehtinen & Mantyla 2011) traditionally used in the software engineering field was adopted to organize the output of this phase. These process areas are presented in *Table 3*.

The output of the analysis was formalized in a graphic diagram that summarizes the process areas of the classification scheme; the reasons for software development processes poor implementation; the failures previous source causes might lead to; and the solutions the organizations are attempting. Finally, conclusions are drawn and some recommendations are proposed.

Process area	Description
Requirements Engineering	Causes are focused on the requirements engineering and input from customers.
Management	Causes are focused on the company support and the way the project stakeholders are managed and allocated to tasks.
Development Work	Causes are focused on implementation of features and its output.
Software Testing	Causes are focused on software testing and its output.
Change Management	Causes are focused on implementation of change requests.
Product Release and Deployment	Causes are focused on installing and releasing the product.
Unknown	Causes that cannot be focused on any specific process area.

Table 3. Classification scheme for organizing the output of this study (taken from (Lehtinen & Mantyla 2011))

4 Results

The five interviewees were responsible for one or more of their IT processes (three of them being responsible for the process implementation itself). Their organizations are in the fields of IT and/or telecommunications, except for one that belongs to the gaming industry.

All interviewees demonstrated that they know well their processes. From the three organizations implementing Scrum, only one (I1) of them complied with all recommendations, while the other ones adapted Scrum to their workflows.

Even though claiming that they try to strictly follow these processes, interviewees admit the existence of several failings that lead, among other consequences, to significant delays. The main reasons cited as sources for software development process poor implementation are listed in *Table 4*, along with the information regarding the interviewee. Below the table we discuss the main reasons, starting with those cited more often.

Lack of communication was the mostly mentioned source for failures (only one organization did not reference it), which might lead to misconceptions that can negatively impact the final product. A situation susceptible to such effect is, for example, is the case where an analysis team performing requirement elicitation and the development team implementing them do not properly communicate.

Interviewees reported **testing** as a neglected phase in their processes. I3 stated that there was **no time allocated for appropriate testing**, while I5 stated that 10% to 20% of bugs are **not properly corrected** leading to bug reopening. I4 mentioned that there are other neglected phases, whose **steps are sometimes skipped** due to lack of time.

Another factor affecting organizations' work processes is **lack of cooperation**. Two interviewees reported that, by having a **poor distribution of responsibilities among employees and departments,** which can also work in isolation, teams are not cooperatively working to solve their clients' problems. I4 believes that employees' **focus is somehow deviated from the client's needs**, and that employees focused just on doing their job or complying with their departments' needs are not working towards the right direction.

Three interviewees stated that their organizations **do not have time allocated for improvements** in the product. This can result in a product with a bad architecture, which can in turn end up in delays in the process, thus impacting other areas of concerns (such as testing).

	I1	I2	I3	I4	I5
Requirements Engineering					
Lack of detail in features/requirements specification		✓			✓
Management					
No time allocated for improvements	✓	✓	✓		
Poor distribution of responsibilities amongst practitioners/departments		✓		✓	
Product owner not always available	✓				
External teams lack the time to assist on the project	✓				
External teams following different methodologies	✓				✓
Pressure to deliver the product to the client			✓		
Development Work					
Lack of process knowledge by practitioners			✓		
Focus deviated from the client's needs				✓	
Software Testing					
No time allocated for appropriated testing			✓	✓	✓
Tests are not properly performed					✓
Change Management					
Lack of agility of the process		✓			
Product Release and Deployment					
Unknown					
Lack of/Poor communication amongst practitioners/departments		✓	✓	✓	✓
Lack of cooperation amongst practitioners/departments		✓	✓	✓	
Lack of practitioners' commitment				✓	
Skipping important steps of the process				✓	

Table 4. Results from the interviews: reasons for software development process poor implementation.

Other reasons include employees' **lack of commitment** in performing their work with correctness, which can reflect on the **lack of detail in requirements and features specification** (usually detected only during implementation time, which can end up in delayed releases) and **external teams following different methodologies** and **lacking the time to work on the project**.

No reasons belonging to the process area of Product Release and Deployment were pointed out, eventually because the interviewees were focused on the software development.

An interesting conclusion is that I2, the interviewee from the organization that uses Waterfall, was the one that reported more failures, most of them linked to the process itself. When asked for the reasons not to adopt a more light-weight process, the interviewee informed that, in the past, a project was conducted using an Agile methodology, which ended up in costs that were twice the budget. Because of that, management maintained the Waterfall methodology.

Despite all these reported failures, none of the organizations seemed to be running initiatives to solve them. However, interviewees identified some potential solutions for tackling these failures:

- Allocate time in the project for improvements (such as refactoring) which is particularly appropriate for Agile methodologies;
- Define more realistic deadlines, which should consider not only the client's needs but also the necessary time to finish tasks with rigour;
- Schedule meetings with external teams earlier in time;
- Increase the motivation to perform all necessary meetings to discuss requirements and features;
- Distribute responsibilities equally amongst employees and departments;
- Automate testing phase with an appropriate tool;
- Adopt adequate software tools to assist on the work process;
- Ask the Product Owner to assign a representative that could act as a "proxy" to the development team (specific to Scrum methodology).

Because there is evidence that performance rewards are related to project success (Mahaney & Lederer 2006), we investigated if interviewees' organizations had performance rewarding mechanisms implemented and their personal opinions regarding the subject.

Three out of the five organizations rewarded employees based on performance evaluations, all in the form of monetary rewards. All interviewees agreed that it can, somehow, motivate employees. They believed that, more than the money, it is the **public recognition** that drive employees to work more and better, thus it is important that such rewards are **not anonymous**.

They also mentioned that **individual rewards** could potentially **lead to conflicts inside teams**, especially those following Agile methodologies, which are very team-oriented. Regarding collective rewards, it was suggested that teams should not be compared between them, since their goals and members can be different. An idea is to split a reward equally among team members if the team improved their results; i.e., the **team is evaluated against itself in the past**.

Finally, they suggested that this performance evaluation could be based on **relevant metrics** to provide meaningful **feedback**.

5 Analysis

Data collected from the interviews were then crossed against the research literature to provide means to answer the research questions.

As already mentioned, there are only a few research papers about the extent to which software development processes are implemented and the reasons underlying implementation failures. In fact, only two research papers were found to be directly relevant to this study.

Three other research papers addressing software development projects failure in general were also examined, though failures related to projects' characteristics and environment (economical or organizational) were discarded because, as already stated, the focus was on failures related to practitioners' characteristics and relationships as well as the development process itself.

A summary of the comparison between the interviews and the research literature can be found in *Table 5*, which was extended from *Table 4*.

Paper P1 describes the practical usage of software development processes by developers in very small organizations in Ecuador (Sánchez-Gordón & O'Connor 2016). Authors state that some reasons for failure in the software development processes in very small companies include the lack of focus on the customer's needs; lack of understanding of requirements (which affects the features specification); lack of time for proper testing; and the absence of progress tracking mechanisms for providing feedback. All these failures were also identified in the interviews, except the last issue.

	I1	I2	I3	I4	I5	P1	P2	P3	P4	P5
Requirements Engineering										
Lack of detail in features/requirements specification		✓			✓	✓		✓		✓
Management										
No time allocated for improvements	✓	✓	✓					✓		
Inadequate planning								✓	✓	✓
Poor distribution of responsibilities		✓		✓						
Poor definition of roles and responsibilities									✓	
Product owner not always available	✓									
Insufficient/inefficient project management										✓
Incompetent project manager									✓	
External teams lack the time to assist on the project	✓									
External teams following different methodologies	✓				✓					
Lack of control/communication with external teams									✓	
Pressure to deliver the product to the client			✓				✓			✓
Lack of progress tracking and feedback mechanisms						✓	✓			✓
Development Work										
Lack of process knowledge by practitioners			✓							
Lack of process experience by practitioners							✓			
Lack of technical expertise/experience									✓	
Focus directed to technical problems only									✓	
Focus deviated from the client's needs				✓		✓				
Software Testing										
No time allocated for appropriated testing			✓	✓	✓	✓		✓		
Tests are not properly performed					✓					
Change Management										
Lack of agility of the process		✓					✓			✓
Product Release and Deployment										
Unknown										
Lack of/Poor communication		✓	✓	✓	✓			✓	✓	✓
Lack of cooperation		✓	✓	✓						
Lack of commitment				✓					✓	
Lack of motivation and involvement							✓		✓	
Skipping important steps of the process				✓						

Table 5. Results from crossing interviews against literature: reasons for software development processes poor implementation

Paper P2 presents the results of twelve semi-structured interviews in a Danish software development company to understand how developers use information systems development methodologies in practice (Kautz et al. 2004). Failures such as the lack of agility of sequential processes (which can make it harder to accommodate changes) and the pressure to deliver the product on time are compliant with results achieved with the interviews. Other suggested failures linked with the poor implementation of software development processes include the lack of feedback on developers' progression; lack of experience using the adopted methodology and lack of developers' motivation and involvement.

An analysis on failures of software development is performed in paper P3 (Kaur & Sengupta 2011). The poor definition of requirements, which are often high level and vague; the improper communication inside teams; lack of time to perform testing and lack of time allocated for quality improvements are reasons highlighted both in this study and in the interviews. The inadequate planning was also an issue highlighted in this study.

Paper P4 analysis a survey conducted on research literature from 1996 to 2006, addressing the factors that influence the outcomes of software development (McLeod & MacDonell 2011). A total of 177 empirical studies were analysed, meaning that this paper provided insights based on a wide range of works. As suggested in the interviews, the lack of commitment and communication in developers are reasons leading to software development process poor implementation. Additionally, this study points out other reasons for failure: lack of developers' expertise/experience; lack of developers' motivation; focus on technical problems (thus, neglecting human and organizational issues), lack of control and communication with external teams; poor definition of roles and responsibilities, conflicts arising between developers; inadequate/insufficient planning; and the existence of an incompetent project manager.

Finally, paper P5 identifies some root causes for software project failures based on different sources (Mandal & Pal 2015). Confirming the interviews results, this study indicates lack of communication; non-adaptability of a process to changes (that means, lack of agility of the process); pressures to deliver the product on time; and the poor definition of system requirements as reasons for not implementing all activities within the software development process. Other reasons include insufficient/inefficient project management; poor project tracking and feedback; and inadequate planning.

The research literature confirms that "Lack of communication", "No time allocated for appropriated testing" and "No time allocated for improvements" are three failures affecting organizations' software development processes. "Lack of detail in features/requirements specification" was found to be one of the most relevant failures after data triangulation.

The literature also introduced 10 additional reasons for software development processes poor implementation, six of them in the Management process area. "Inadequate planning" and "Lack of progress tracking and feedback mechanisms" were reported in three papers and "Lack of motivation and involvement" in two papers; the remaining reasons were mentioned only once.

On the other side, there were eight reasons highlighted in the interviews but not in the literature analysis, four of them in the Management process area. "Poor distribution of responsibilities amongst practitioners and departments" and "External teams following different methodologies" were reported twice, while the remaining were reported only once.

The Management area was the one with greater variation between interviews and literature analysis. Similarly, no reasons were reported from the Product Release and Deployment area. Also, no evidence was found regarding initiatives conducted by organizations to address the mentioned failures.

Results from this comparison are presented in *Figure 3* that represents only the most relevant implementation failures. Process areas with no failures (Development Work and Product Release & Deployment) were omitted from the diagram. Because there were no organization initiatives mentioned

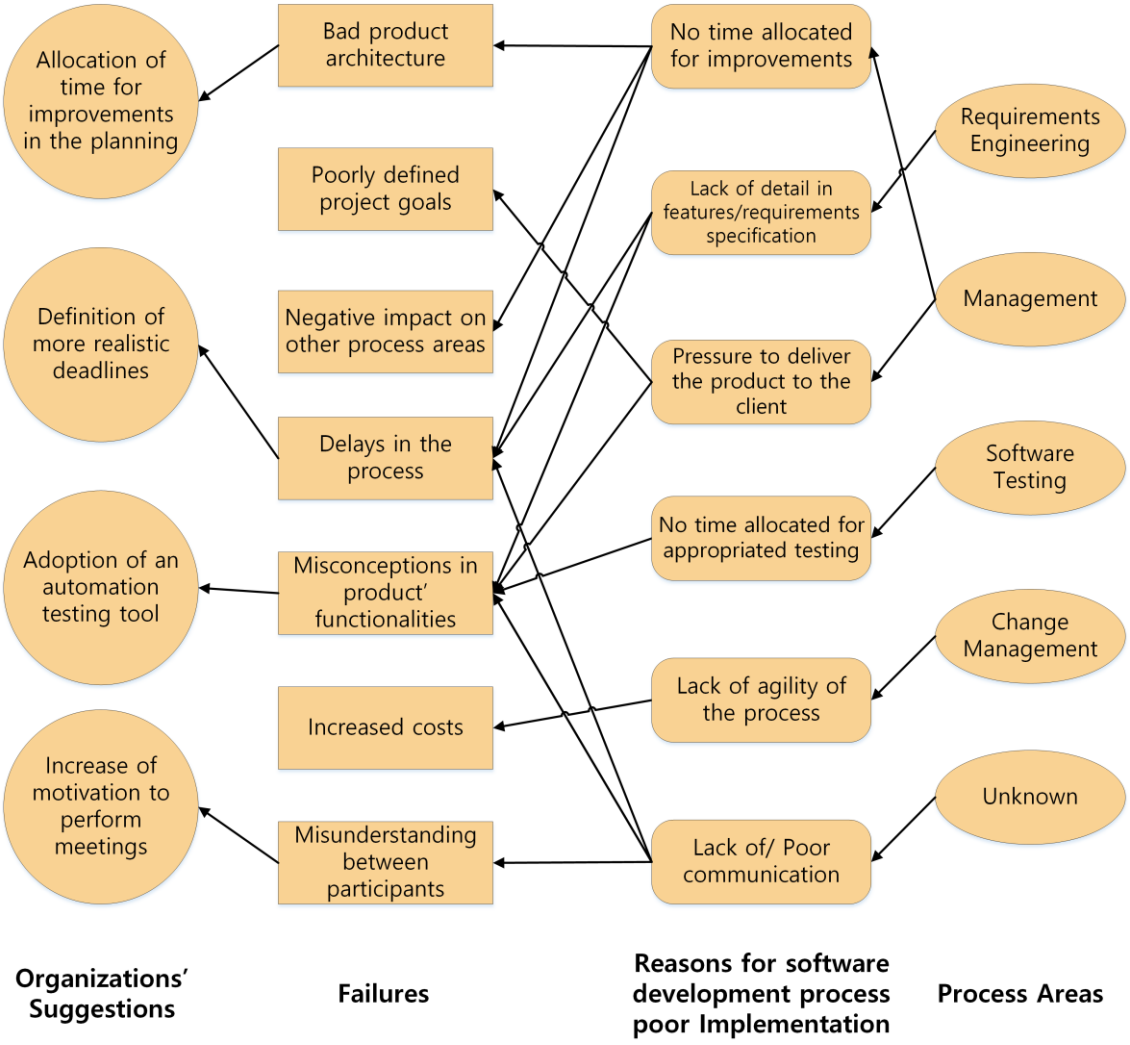

Figure 3. Summary of the study's outputs.

to cope with software development process failures, some of the suggestions provided by the interviewees were added to the diagram.

6 Lessons Learnt

RQ1: Which activities are being implemented by software development organizations?

All activities seemed to be implemented in the software development organizations, but this is not done flawlessly. A small number of reasons for failure were found in the Requirements Engineering and Software Testing areas. However, these belong to the most mentioned reasons in the whole list.

Management was the area where more failures' reasons were found, however they were not consistent between interviews and literature and the majority were reported only once or twice. In the Development Work area there were some failures pointed out, but none of them were frequent. Only one problem was reported in the Change Management area, and no evidence was found regarding failures affecting the Product Release and Development area. Despite the existence of only a few reasons in the Unknown area, it was the area that hosted the mostly reported problem.

RQ2: Which are the reasons leading to failures in the software development process?

The lack of communication between practitioners and departments was the problem affecting software development organizations the most. The lack of time for appropriate testing and to perform improvements on the product were found to negatively impact the development process, which furthermore suffers from the poor specification of requirements and features. Other reasons relevant and consistent among data sources are the pressure to deliver the product to the client and the lack of agility of the processes.

RQ3: How are organizations addressing these failures?

No evidence was found regarding the promotion of initiatives to cope with failures affecting the development process. Only in the interviews some suggestions were made, but none of them was applied in practice. However, some of these suggestions were introduced in the diagram as possible solutions for some of the failures affecting the development process.

7 Conclusion

This study addressed the problem of the poor implementation of software development processes in organizations. In order to understand which are the affected activities, the reasons leading to those activities' poor implementation and what are organizations doing to address these failures, the authors conducted five semi-structured interviews with professionals responsible for the implementation of their organization' processes. Insights of these software development workers were triangulated with information collected from relevant literature, which provided means to answer the research questions.

7.1 Contribution

The main contribution of this study is the better understanding of the failures affecting the software development process, particularly the validation that failures occurring in Portuguese organizations are referenced in the literature. All software process activities are implemented but not without some slip. Management activities have several reasons for failure associated, but each of them were not often reported. In contrast, Requirements Engineering and Software Testing present fewer reasons for failure, but they are more frequently reported. Lack of communication, lack of time for improvements and appropriate testing and the poor specification of requirements and features were the main reasons found that lead to poor implementation in software development processes. Yet, organizations do not seem to be applying initiatives to overcome these failures.

7.2 Limitations

To begin with, a small number of interviews were conducted. Despite the inexistence of a perfect sample size value (Mason 2010), the number of interviews should be sufficient to reach data saturation; that means, until the point where new data collected provides few or no insights at all (Saunders et al. 2008). However, for research questions directed to a homogeneous group, 12 interviews should be sufficient (Guest et al. 2006), while in this study only five were interviewed.

The homogeneity of the sample is itself a limitation: all interviewees were responsible for their development process implementation (thus, the perspectives of developers was left out of this study) and belonged to Portuguese organizations, mostly in the IT and/or telecommunications sector. This bias might explain why some reasons for failure were not reported during the interviews. Project managers unlikely would say that project management was inefficient or that they were incompetent.

The set of papers used to perform the literature review, besides being of small number, have other restraining factors. The two research papers directly relevant to the study were conducted in either a

specific type of organizations, in a specific country or in a single organization. The timeframe of the works presented in these papers can also be considered a drawback. Even though three papers were published recently (2015, 2016 and 2011), another paper is from 2004. The systematic survey paper gathering data from 177 empirical studies, despite being published in 2011, is based on work published between 1996 and 2006.

The analysis of the reasons leading to a poor implementation of software development processes was conducted as if these failures were independent and equally relevant in all contexts, which is not true. In the literature analysis, there is some evidence for the fact that some failures might be leading to others, and these relationships are important. Furthermore, only the frequency with which each issue is mentioned was studied, leaving the level of impact out of the scope.

7.3 Future Work

The set of research participants can be expanded, both in terms of number and diversity, to produce more consistent results. At this moment, a survey is being prepared to be distributed among software development practitioners all over the world. This will embrace the developers' perspectives on the failures affecting the implementation of software development processes, which will then be triangulated with this paper's results to validate findings.

By introducing developers in the study, the impact of failures related to motivation, feedback and performance rewarding on the software development process can be evaluated. This study suggests that the lack of feedback mechanisms is an issue leading to poor implementation of software development processes. In turn, positive feedback can influence the motivation of a developer to perform his/her work (McLeod & MacDonell 2011; Baddoo & Hall 2002). So, it seems relevant to test the hypothesis that feedback mechanisms combined with performance rewarding can increase developers' motivation and have a positive impact on the software development process. This discussion can point towards the direction of gamification (Deterding et al. 2011).

Further research is needed to understand why organizations are not conducting initiatives to address the reported failures, once practitioners seem to know what could possible avoid them.

Finally, the causal relations between these failures should be studied, as already done in previous works (Lehtinen et al. 2014). Failures outside of this study's focus (i.e., those related to project's characteristics and environment) that emerge as cause or effect of other failures should be considered under this context, as they can influence the interpretation of the failures they are related with. Nevertheless, failures related to practitioners and the development process must remain the focus.

Because the frequency with which an issue happens does not translate on the impact with which it affects the software development process, this study is also left for future work.

References

Baddoo, N. & Hall, T., 2002. Motivators of Software Process Improvement: An analysis of practitioners' views. *Journal of Systems and Software*, 62(2), pp.85–96.

Basri, S. & O'Connor, R. V., 2010. Understanding the perception of very small software companies towards the adoption of process standards. In *European Conference on Software Process Improvement*. Springer, pp. 153–164.

Beck, K., 2000. *Extreme programming explained: embrace change*, Addison-Wesley Professional.

Bucero, A., 2007. Failing to succeed. *PM Network*, 21(11), p.21. Available at: http://www.pmi.org/learning/library/negative-positive-project-failure-4577 [Accessed April 20, 2017].

Cramm, S., 2001. Managing Software Development Efficiently. *CIO*. Available at: http://www.cio.com/article/2441202/project-management/managing-software-development-efficiently.html [Accessed April 20, 2017].

DeMarco, T. & Lister, T., 2013. *Peopleware: productive projects and teams*, Addison-Wesley.

Deterding, S. et al., 2011. From game design elements to gamefulness. *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11*, pp.9–11.

Dubois, D.J. & Tamburrelli, G., 2013. Understanding gamification mechanisms for software development. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*. New York: ACM, p. 659.

Fitzgerald, B., 1998. An Empirical Investigation into the Adoption of System Development Methodologies. *Information & Management*, 34(6), pp.317–328.

Florentine, S., 2016. More than half of IT projects still failing. *CIO*. Available at: http://www.cio.com/article/3068502/project-management/more-than-half-of-it-projects-still-failing.html [Accessed April 20, 2017].

Guest, G., Bunce, A. & Johnson, L., 2006. How Many Interviews Are Enough ? An Experiment with Data Saturation and Variability. *Family Health International*, 18(1), pp.59–82.

International Organizational for Standardization, 2015. *ISO 9000 - Quality management*, Genève, Switzerland.

Janesick, V.J., 2000. The choreography of qualitative research design. *Handbook of Qualitative Research*, pp.379–399.

Kaur, R. & Sengupta, J., 2011. Software Process Models and Analysis on Failure of Software Development Projects. *International Journal of Scientific & Engineering Research*, 2(2), pp.1–4.

Kautz, K., Hansen, B. & Jacobsen, D., 2004. The Utilization of Information Systems Development Methodologies in Practice. *Journal of Information Technology Case and Application Research*, 6(4), pp.1–20.

Laporte, C.Y., Alexandre, S. & O'Connor, R. V, 2008. A software engineering lifecycle standard for very small enterprises. In *European Conference on Software Process Improvement*. Springer, pp. 129–141.

Lehtinen, T.O.A. et al., 2014. Perceived causes of software project failures - An analysis of their relationships. *Information and Software Technology*, 56(6), pp.623–643.

Lehtinen, T.O.A. & Mantyla, M. V, 2011. What Are Problem Causes of Software Projects ? Data of Root Cause Analysis at Four Software Companies. In *International Symposium on Empirical Software Engineering and Measurement*. IEEE, pp. 388–391.

Mahaney, R.C. & Lederer, A.L., 2006. The effect of intrinsic and extrinsic rewards for developers on information systems project success. *Project Management Journal*, 37(4), p.42.

Mandal, A. & Pal, S.C., 2015. Identifying the reasons for software project failure and some of their proposed remedial through BRIDGE process models. *International Journal of Computer Sciences and Engineering*, 3(1), pp.118–126.

Marchenko, A. & Abrahamsson, P., 2008. Scrum in a multiproject environment: An ethnographically-inspired case study on the adoption challenges. In *Agile 2008 Conference*. IEEE, pp. 15–26.

Mason, M., 2010. Sample size and saturation in PhD studies using qualitative interviews. In *Forum qualitative Sozialforschung/Forum: qualitative social research*.

McLeod, L. & MacDonell, S.G., 2011. Factors that affect software systems development project outcomes. *ACM Computing Surveys*, 43(4), pp.1–56.

Myers, M.D., 1997. Qualitative research in information systems. *Management Information Systems Quarterly*, 21(June), pp.241–242.

Myers, M.D. & Newman, M., 2007. The qualitative interview in IS research: Examining the craft. *Information and Organization*, 17(1), pp.2–26.

Overhage, S. et al., 2011. What Makes IT personnel adopt scrum? A framework of drivers and inhibitors to developer acceptance. In *Proceedings of the 44th Annual Hawaii International Conference on System Sciences*. IEEE, pp. 1–10.

Royce, W.W., 1970. Managing the development of large software systems. In *Proceedings of IEEE WESCON*. pp. 328–338.

Sánchez-Gordón, M.L. & O'Connor, R. V., 2016. Understanding the gap between software process practices and actual practice in very small companies. *Software Quality Journal*, 24(3), pp.549–570.

Saunders, M., Lewis, P. & Thornhill, A., 2008. *Research Methods for Business Students*, Harlow: Prentice Hall.

Schwaber, K., 1995. Scrum development process. In *Workshop on Business Object Design and Implementation*.

Shaker, K., 2010. Why do projects really fail? *PM Network*, 24(7), pp.20–21. Available at: http://www.pmi.org/learning/library/identify-factors-cause-project-failure-2442 [Accessed April 20, 2017].

Software Engineering Institute, 2010. *CMMI for Development, Version 1.3*,

Sommerville, I., 2015. *Software Engineering* 10th Editi., Pearson.

Standish Group, 2015. *CHAOS Report*,

Staples, M. et al., 2007. An exploratory study of why organizations do not adopt CMMI. *Journal of Systems and Software*, 80(6), pp.883–895.

Varajão, J. et al., 2014. Failures in software project management - are we alone? A comparison with construction industry. *International Journal of Modern Project Management*, 2(1), pp.22–27.

Xu, L. & Brinkkemper, S., 2007. Concepts of product software. *European Journal of Information Systems*, 16(5), pp.531–541.

Zahran, S., 1998. *Software process improvement—Practical guidelines for business success*, Boston, MA: Addison-Wesley.