PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TOWARDS AN SDLC FOR SOFTWARE DEVELOPMENT PROJECTS INVOLVING DISTRIBUTED SYSTEMS

RODRIGO AUGUSTO DOS SANTOS

Dissertation presented as a partial requirement to obtain a Computer Science Master´s degree at Pontifícia Universidade Católica do Rio Grande do Sul.

Supervisor: Prof. Avelino Francisco Zorzo
Co-Supervisor: Prof. Sabrina Marczak

Porto Alegre
2016

Dados Internacionais de Catalogação na Publicação (CIP)

S237t Santos, Rodrigo Augusto dos

Towards an SDLC for software development projects involving distributed systems / Rodrigo Augusto dos Santos. – 2016.
88 f.

Diss. (Mestrado) – Faculdade de Informática, PUCRS.
Orientador: Prof. Dr. Avelino Francisco Zorzo.
Co-orientador: Profª. Drª. Sabrina Marczak.

1. Sistemas distribuídos (Informática). 2. Engenharia de software. 3. Gestão de projetos. 4. Informática. I. Zorzo, Avelino Francisco. II. Marczak, Sabrina. III. Título.

CDD 23 ed. 005.276

Marcelo Votto Texeira CRB 10/1974
Setor de Tratamento da Informação da BC-PUCRS

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "*Towards an SDLC for Software Development Projects Involving Distributed Systems*" apresentada por Rodrigo Augusto Silva dos Santos como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, aprovada em 10 de março de 2016 pela Comissão Examinadora:

Prof. Dr. Avelino Francisco Zorzo– Orientador — PPGCC/PUCRS

Profa. Dra. Sabrina dos Santos Marczak – Coorientadora — PPGCC/PUCRS

Prof. Dr. Rafael Prikladnicki – — PPGCC/PUCRS

Prof. Dr. Cleidson Ronald Botelho de Souza - — UFPA

Homologada em 02/06/2016, conforme Ata No. 011 pela Comissão Coordenadora.

Prof. Dr. Luiz Gustavo Leão Fernandes
Coordenador.

RUMO A UM SDLC PARA PROJETOS DE DESENVOLVIMENTO DE SOFTWARE QUE ENVOLVAM SISTEMAS DISTRIBUÍDOS

RESUMO

[Contexto] Desde os anos 1970, Sistemas Distribuídos vêm se tornando em uma opção cada vez mais viável e confiável para a implementação de sistemas de informação. Desde então, a evolução destes sistemas continuou em um ritmo acelerado. Eles atualmente são aplicáveis a uma variedade de propósitos, tais como jogos online, sistemas financeiros, soluções computacionais em nuvem, etc. É possível então assumir que nos dias de hoje, Sistemas Distribuídos estão em todos os lugares, e que há uma grande probabilidade de que qualquer projeto de desenvolvimento de software em andamento esteja usando este paradigma como parte da sua proposta de entrega. Dessa forma, é relevante o estudo dos impactos que Sistemas Distribuídos trazem à disciplina de Gestão de Projetos. **[Objetivos]** Neste trabalho, nós discutimos estes impactos e desafios, assim como propomos um Ciclo de Vida de Desenvolvimento de Software (SDLC) e suas práticas associadas, ambos sendo adaptados para o uso em projetos de desenvolvimento de software que envolvam Sistemas Distribuídos. As práticas propostas foram otimizadas para implementação em um regime Cascata, sendo contudo também adaptáveis ao uso sob o *framework* Scrum. **[Método]** Em um primeiro momento, um Estudo de Mapeamento Sistemático foi conduzido para entendimento do Estado-da-arte com relação aos estudos acadêmicos localizados na intersecção entre Gestão de Projetos e Sistemas Distribuídos. A seguir, entrevistas qualitativas foram executadas com membros da indústria da Tecnologia da Informação, objetivando confirmar os resultados encontrados no Estudo de Mapeamento Sistemático, além de obter *feedback* relacionado aos desafios que Projetos atuais de Sistemas Distribuídos trazem, buscando-se ainda identificar contramedidas desejáveis para mitigar ou mesmo anular tais desafios. **[Resultados]** Como terceiro e final passo, um *SDLC* genérico, assim como práticas associadas à ele, ambos tailorizados para projetos envolvendo sistemas distribuídos, foram propostos como resposta direta aos resultados obtidos das entrevistas qualitativas. As práticas tailorizadas mencionadas se constituem na espinha dorsal de nossas contribuições. As propostas apresentadas passaram pelo processo de *member-checking* para validação e refinamento, o que resultou na versão final apresentada nesta dissertação.

Palavras-chave: Sistemas Distribuídos, Times Distribuídos, Engenharia de Software Global, Gestão de Projetos, Ciclo de Vida de Desenvolvimento de Software, Ciclo de Vida do Projeto.

TOWARDS AN SDLC FOR SOFTWARE DEVELOPMENT PROJECTS INVOLVING DISTRIBUTED SYSTEMS

ABSTRACT

[Context] Since the 1970's, Distributed Systems have been turning into a more viable and reliable option for the implementation of information systems. Since then, their evolution continued in an accelerated pace. They now are applicable to a variety of purposes, such as online games, financial systems, cloud computational solutions, etc. It is possible then to assume that today, Distributed Systems are found everywhere, and that there is a great probability for any given in-progress software development project to be using this paradigm as part of its delivery proposal. Thus, it is relevant to study the impacts that Distributed Systems bring to the Project Management discipline. **[Objectives]** In this dissertation we discuss those impacts and challenges, as well as propose a Software Development Lifecycle (SDLC) and associated practices, both adapted for use within software development projects involving Distributed Systems. These practices are optimized for implementation under a Waterfall regime, but are also adaptable for use under the Scrum agile framework. **[Method]** At first a Systematic Mapping Study (SMS) was conducted for understanding the State-of-the-art regarding academic studies located in intersection of Project Management and Distributed Systems. Next, empirical qualitative interviews were held with members from the Information Technology Industry, aiming to confirm the SMS results as well as obtaining feedback regarding present day's challenges of Distributed Systems Projects. Desirable countermeasures for these challenges were also being searched for. **[Results]** As a third and final step, a generic SDLC as well as its associated practices, both tailored for projects involving DS, were proposed in direct response to the results obtained from the qualitative interviews. The tailored practices constitute the backbone of our contributions. The presented proposals went through the process of member-checking for validation and refinement, which led to the final version shown in this dissertation.

Keywords: Distributed System, Distributed Teams, Global Software Engineering, Project Management, Software Development Life Cycle, Project Life Cycle.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABREVIATIONS

SUMMARY

1. INTRODUCTION

The definition of [PMI15b] for a project is “a temporary endeavor in that it has a defined beginning and end time, and therefore defined scope and resources”. Complementary to this definition is the one related to Project Management (PM), as according to [PMI13], “it is the application of knowledge, skills, tools and techniques to project activities to meet the project requirements”. Project Management should thus be viewed by organizations as a strategic competence, considering its role in allowing them to better compete in their markets [PMI15a].

Since the dawn of mankind, Project Management has been present as a discipline [NID05], even if in an informal way. According to [KOZ11], the Egyptian Giza Pyramid, the Roman Colosseum and the United States Transcontinental Railroad are all historical projects from the past four millennia. More recently, the Americans Hoover dam and the atomic bomb (also known as the Manhattan project) [KWA05] are examples of what are considered delivered projects.

As pointed by [SNY87], “the efforts listed above were successfully managed and delivered prior to 1957, the point in time when PM is considered to have been established as a formal discipline”. At that time, the technology advancements that affected many industries, including the computer industry, were the main driver for this establishment [KOZ11]. Still according to [KOZ11], “in order to cope with many of these complex advancements, many PM software companies were founded during the 1970s, including Artemis, Scitor Corporation and Oracle”.

Starting on the 1970’s, the wide adoption of Distributed Systems became a fact, and Information Technology (IT) Project Managers around the world were forced to deal with it. According to [COU12], “Distributed Systems (DS) or Distributed Information Systems are the ones in which hardware or software components, located at networked computers, communicate and coordinate their actions only by passing messages”. This dynamic is transparent to users, and their perception is that they are interacting with a single and integrated computer system.

Independent failure of components and resource sharing are also key characteristics of DS [COU12], having scalability in their nature. Distributed programs are computer applications that run over DS. [COU12] lists some important DS examples covered by the definition above, such as web search, multiplayer online games, and financial trading systems, and also points out to the fact that “DS encompass many of the most significant, technological developments of recent years, ranging from a small intranet to the Internet”. This turns the intersection between PM and DS a relevant research area. Despite this relevancy, our hypothesis was that this key project system distribution characteristic may be "abstracted" by project teams, sharply increasing chances of project failure. This risk would come from the "hidden" treatment given to the system distribution requirements.

Although critical, the decisions regarding system distribution could become fully delegated to development teams, thus being absent from the regular flows of communication, being treated away from the whole team, which includes project managers, users and customers. The team's focus would be instead on supposedly "real, attention-worthy, value-driven requirements", such as screens, business rules, reports, and other "tangible" features, while system distribution ones and their impacts were being neglected.

This culture would also reflect upon the academy, with small attention being provided to the DS PM area. A variety of reasons could be behind this abstraction phenomenon, such as the advances in technology that could abstract the complexities of DS to a satisfactory state, or even the general principle that states that projects fail mostly due to people, and less due to technology.

In order to have a comparison measure for the volume of research on DS PM, we have used Project Management involving Distributed Teams (DT). Team distribution became a very popular research topic in the same measure it has been embraced by an increasing number of organizations distributing their software development processes worldwide aiming at heightened profit and productivity as well as cost reduction and quality improvements [HER01].

Although DS and DT are two distinct subjects, with no direct relation between them, both subjects are supposedly to be present in a great number of today's projects, the research on both having the characteristic of being able to intersect with PM. To understand the relation between DS, DT and PM, a Systematic Mapping Study (SMS) was carried out aiming to confirm the level of attention provided to DS PM in comparison to the volume of studies focusing in DT PM.

Next, as previously mentioned, qualitative interviews were conducted for confirming as well as complementing the SMS results. Only then proposals were made to address the identified pain points, such proposals still going through the member-checking process for validation and further improvement. All these steps are presented in detail in this dissertation.

This dissertation has the following structure: Section 2 describes the SMS. Section 3 explains the qualitative interview process. Section 4 explains the proposals made as a result of this work. Section 5 explains the member-checking process. Section 5 also brings the threats to the validity of this research. Per last, Section 6 presents the conclusions reached.

2. SYSTEMATIC MAPPING PROCESS

It is stated by [KIT07] that "a Systematic Literature Review (SLR) is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest". Still according to [KIT07], "a Systematic Mapping Study (SMS), on the other hand, is designed to provide a wide overview of a research area, to establish if research evidence exists on a topic and to provide an indication of the quantity of the evidence". While SLRs provide more complete results on a matter, SMSs are less laborious considering its results are more coarse-grained [PFM08]. As this is an exploratory research, we have chosen to run an SMS first, seeking direction for the entire research.

As mentioned in previous sections, this SMS was first proposed as a response to the growing trend of distribution across the Information Technology industry, as prior to it, our understanding was that many aspects of distribution and their impacts to Project Management seemed to have been covered by previous works, but that some of these aspects, despite being relevant, were overlooked, both by the industry itself, as well as by the academy. One such case was that of System Distribution and its impacts to Project Management.

In order to correctly interpret the still to-be found research volume on DS PM, first we established we needed a comparison measure coming from research on a different topic, but which beared similar overall characteristics. We found that comparison possibility in PM involving Distributed Teams (DT). The SLR, thus, was performed for confirming the level of attention provided to DS PM when compared to attention on DT PM.

2.1 SMS Research Questions

This SMS provides a general overview of these two important aspects of distribution, DS and DT, applicable to the IT industry, mapping their impacts against Project Management. The framework *Population, Intervention, Comparison, Outcomes* (PICO), proposed by [PET08], was followed for providing the basis for the definition of the two research questions that were part of the SMS. Three of the framework's four key components were adopted, plus the additional component *Context*, also proposed by [PET08]. Following are the components as well as the resulting SMS Research Questions (RQ):

- Population: Global Software Engineering (GSE).
- Intervention: Distribution aspects applicable to GSE.

- Outcome: Project Management.
- Context: Industry experiences, with complimentary papers from the Academy.

RQ1: "*Are distributed teams project management and distributed systems project management receiving the same level of research attention?*"

RQ2: "*What are the challenges brought to Project Management by team distribution and system distribution?*"

2.2 Review Protocol

[KIT07] states that "a review protocol specifies the methods that will be used to undertake a specific systematic review". Before an academic research is started, a pre-defined protocol must be available, since the researchers will seek to reduce as much as possible the possibility of bias [KIT07]. The review protocol for this dissertation is described in the following section.

2.2.1 Search terms

The research questions, derived from the PICO framework provided keywords for the research. These are listed in table 1.

Table 1: Search terms

KEYWORD	SYNONYMS
Global Software Engineering	Distributed Software Development Distributed Software Engineering Global Software Development Distributed Systems Engineering Global Systems Engineering
Distributed System	Distributed Systems Distributed Information Systems Distributed Information System Distributed Program Distributed Programs
Project Management	

2.2.2 Databases

Four on-line Computer Science databases were selected as resources to be searched, based on the fact that all of them had a web search engine that allowed keyword-based customized search strings to be used for papers retrieval. Those databases are:

- IEEEXplore Digital Library (http://ieeexplore.ieee.org/)
- ACM Library (http://portal.acm.org)
- Elsevier ScienceDirect (www.sciencedirect.com)
- Scopus (www.scopus.com)

2.2.3 Search strings

The original search string for use with IEEEXplore database was created by uniting the keywords and their synonyms with the logic operator “OR”, representing Population and Intervention, as well as the logic operator “AND”, for Outcome. The same string needed to be altered in order to become compatible with the other remaining databases. Both strings are described in Table 2.

Table 2: Search strings

SEARCH STRING	TARGET DATABASE
(("distributed systems" OR "distributed system" OR "distributed information systems" OR "distributed information system" OR "distributed program" OR "distributed programs" OR "distributed software development" OR "distributed software engineering" OR "global software development" OR "global software engineering" OR "distributed systems engineering" OR "global systems engineering") AND "project management")	IEEEXplore
(("distributed systems" OR "distributed system" OR "distributed information systems" OR "distributed information system" OR "distributed program" OR "distributed programs" OR "distributed software development" OR "distributed software engineering" OR "global software development" OR "global software engineering" OR "distributed systems engineering" OR "global systems engineering") AND ("project management"))	ACM Library ScienceDirect Scopus
String structure: Population **OR** Intervention **AND** Outcome	

2.2.4 Selection Criteria

With the search strings defined and after confirmation that they were working, the next step was to clearly define the inclusion and the exclusion criteria to be employed in the screening of the returned papers from the selected databases, thus reducing the returned amount to an acceptable and practical number for carrying on with the research. The Inclusion Criteria (IC) and Exclusion Criteria (EC) used on this work are listed below:

IC1: The study must be available on the Web;
IC2: The study must make some correlation between the distributed aspect(s) and the Project Management discipline, even if it is an implicit correlation or if this relation can be deducted;
EC1: Papers that are focused purely on software development technical aspects, with no relation whatsoever to Project Management or to any relevant distribution forms being studied;
EC2: Papers that are duplicated among two or more of the chosen databases.

2.2.5 Study Quality Assessment and Procedures

According to [WEL09], "a systematic review requires investigators to identify papers of sufficient quality to include in the analysis; because, "if the 'raw material' is flawed, then the conclusions of systematic reviews cannot be trusted". For this dissertation, a checklist of six qualitative questions was created in order to provide clear criteria for quality assessment of the selected works. The checklist items are described below.

QA1. Does the study present a practical problem that demands a solution?
QA2. Is the study recent, published after 2005, meaning it brings the latest insights over the research theme?
QA3. Is Project Management the main focus of the study?
QA4. Does the study present an adequate sample/experiment for reaching its conclusions?
QA5. Does the study present a clear, understandable conclusion?
QA6. Does the paper present any field-tested solution for its research problem?

The possible answers for each of the questions are listed in Table 3.

Table 3: Quality assessment criteria

QUALITY ASSESSMENT (QA)	RESPONSE	RESPONSE MEANING	POINTS EARNED (PER QA)
QA1	Yes	The resolution of a practical problem is an explicit objective of the paper.	1
	Partial	The resolution of a practical problem is an implicit objective of the paper.	0,5
	No	The paper just wants to list state-of-the-art practices, with no practical problem-solving intention.	0
QA2	Yes	The work has been published from 2005 onwards.	1
	Partial	The work has been published before 2005, but after 2000.	0,5
	No	The study is from the 21st century.	0
QA3	Yes	The study of Project Management is the main motivation of the paper.	1
	Partial	The study of a discipline other than Project Management is the main motivation of the study, but Project Management is a secondary motivation of the study.	0,5
	No	The study of Project Management is not the focus of the study, though this SMS´s researchers are able to relate the distribution challenges presented in the paper to the Project Management discipline.	0
QA4	Yes	The paper uses a broad enough sample to justify all of its conclusions.	1
	Partial	The paper uses a sample considered not to be on the ideal size for justifying some of its conclusions.	0,5
	No	The paper uses an insufficient sample which does not allow to justify any of its conclusions.	0
QA5	Yes	The paper´s conclusions are clear and understandable.	1
	Partial	Not all the paper´s conclusions are clear and understandable enough.	0,5
	No	The paper´s conclusions are not clear enough.	0
QA6	Yes	A fully field-tested solution is proposed in the paper.	1
	Partial	A solution is proposed in the paper, but it has only limited field-testing.	0,5
	No	There is no solution proposed in the paper.	0

All the five quality assessment questions were applied by one researcher to each of the 37 papers, the total points earned being added up by paper. The possible outcomes regarding the earned points are listed in Table 4, while the final results obtained are shown in Figure 1.

Table 4: Quality assessment possible results

TOTAL POINTS (ALL QAS TOGETHER)	PAPER CLASSIFICATION
0 to 1,0	Poor
1,5 to 2,5	Fair
3,0 to 4,0	Good

4,5 to 5,0	Very Good
5,5 to 6,0	Excellent

The results obtained are shown in Figure 1:

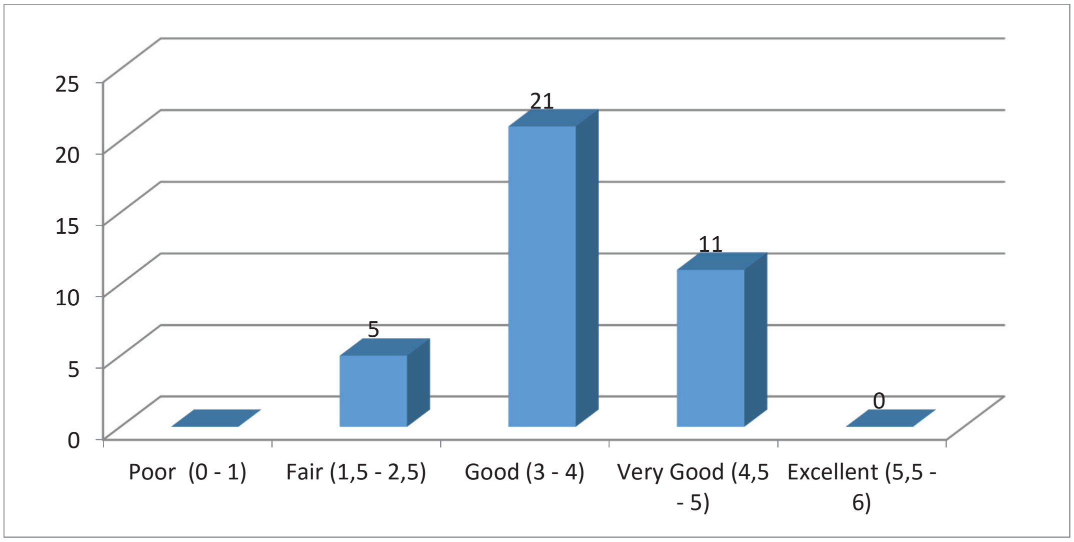

Figure 1: Primary studies quality assessment

2.3 Conducting the Review

This SMS had its goal defined by July 2014, and was planned for execution on August of the same year. The actual execution of the SMS lasted from September to November of 2014.

2.3.1 Selection of primary studies

The previously planned search strings were applied to the previously selected databases. When this was done a total of 21,324 papers were returned, as shown in Figure 2. The high number of papers being returned was due mainly to filtering issues in ACM Library database, as when the research was being conducted outside the University´s network, it seemed not to be able to correctly apply the conditional statements of the search string.

Due to the impossibility of the researchers to be full-time allocated within the University´s campus during the full-length of the research, a choice was made to work with the 21,324 papers. As a first selection phase, they all had their titles screened for redundancies and compatibility with the research´s scope, which resulted in 127 papers being selected.

As a second selection phase, those 127 papers had their Abstracts screened for compatibility with the research´s scope, which further narrowed down the list of eligible papers to 65. A third and last selection phase was carried out, now with the full-reading of the each paper, again aiming to evaluate the paper´s adherence to the research´s goals. After careful selection, as shown in Figures 3 and 4, 37 papers were selected to be the primary studies for this SMS.

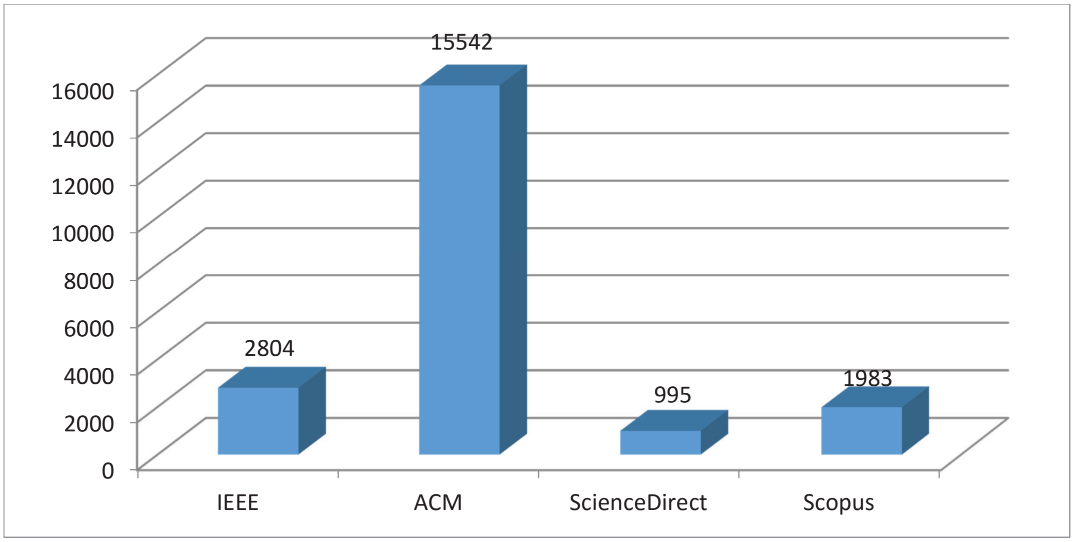

Figure 2: Search string returns

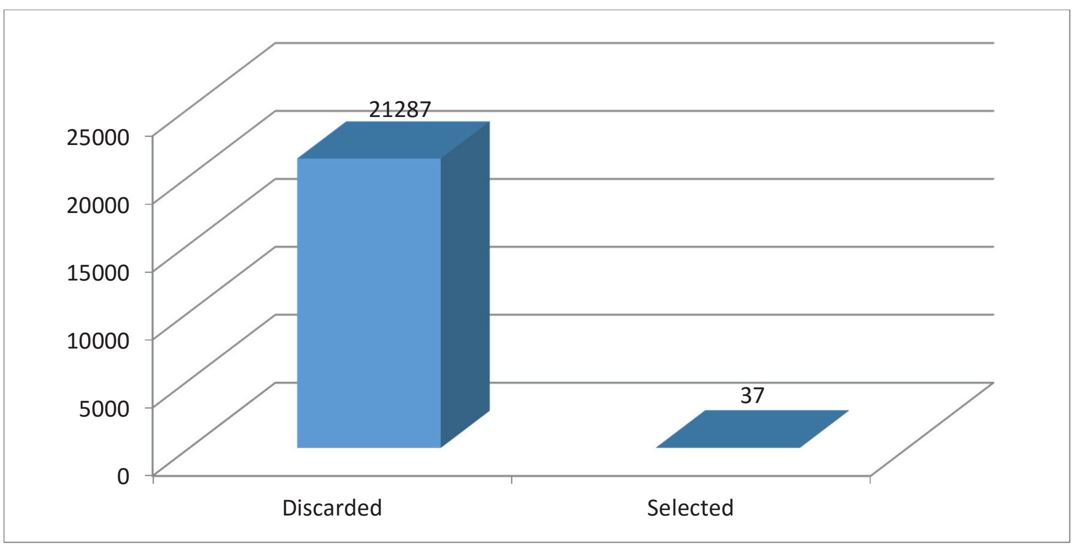

Figure 3: Ratio between discarded and selected primary studies

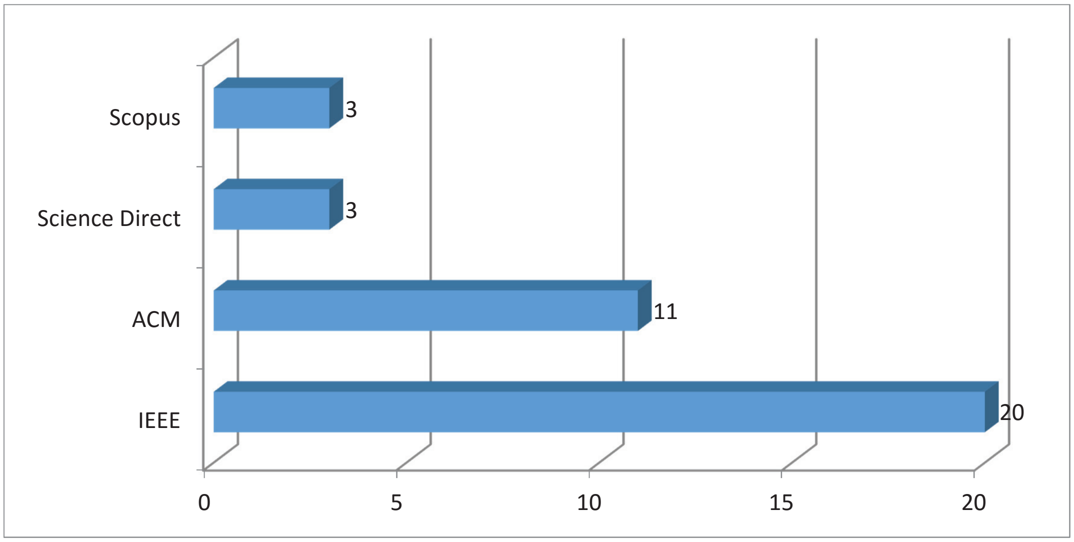

Figure 4: Distribution of primary studies per source

2.4 Reporting the Review

In this section, the answers to this SMS´s two research questions are presented.

2.4.1 Discussion

RQ1: "A*re distributed teams project management and distributed systems project management receiving the same level of research attention?*"

As shown in Figure 5, from the 37 selected papers, 29 focused on the intersection between team distribution and PM, 8 focused on the intersection of system distribution and PM and only 1 focused on the intersection of both team and system distributions with PM, altogether.

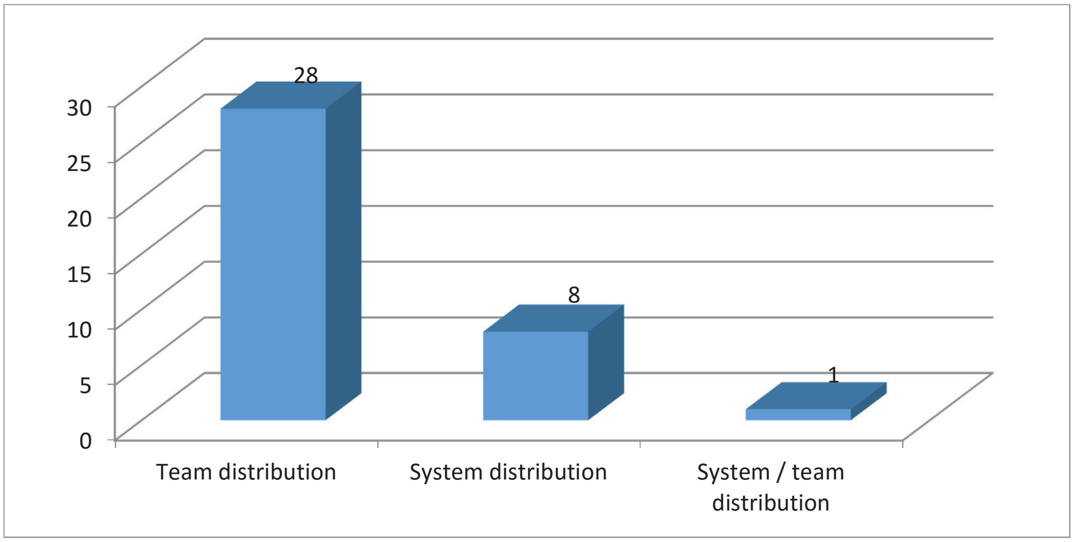

Figure 5: Distribution of aspects intersecting PM by primary studies

The big difference in the volume of team distribution and system distribution focused papers indicates a great focus of the academy´s research on team distribution project management, and a lack of interest in the study of distributed systems project management. Another strong indicator for this lack of interest is the fact that from the 8 selected papers, 50% of them were written before the 2000 decade, as represented in Figure 6. As technology evolved a lot since back then, the discussions on those papers may no longer apply to present day.

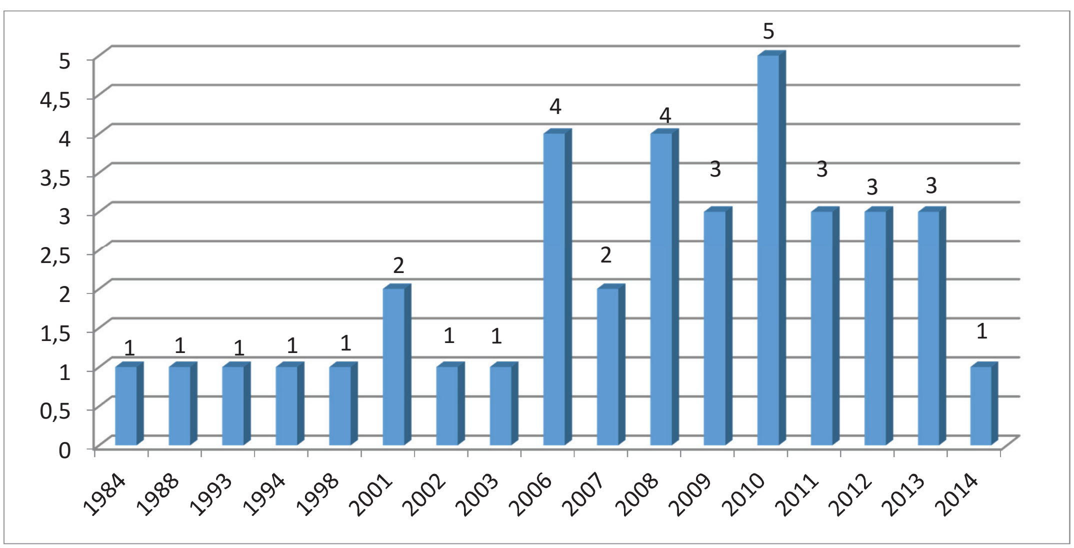

Figure 6: Number of primary studies per year of publication

RQ2: "*What are the challenges brought to Project Management by the team distribution and system distribution?*"

The challenges found were organized in three categories: Category #1, which relates to team distribution challenges (see Tables 2, 3 and 4); Category #2, which relates to system distribution challenges (see Table 5); and Category #3, which relates to challenges common to both team and system distributions (see Table 6).

Table 5: Management challenges of team distribution (Category #1)

CHALLENGE	RELATED PRIMARY STUDIES
Team coordination / dynamics	[BOD07], [COL14], [JIM09], [KOT08], [LEE06], [MIS11], [MOC01], [NIA13], [NUR11], [RAM07], [SIL10], [YAN04]
Team trust and cooperation	[BOD07], [CHR01], [DAM02], [JIM09], [MIS11], [MOC01], [NIA13], [NOL10], [PER10], [SIL10], [VER14]
Knowledge management	[DAM02], [JIM09], [MAN09], [NIA13], [NOL10], [PER10], [SIL10], [SOL10]
Tracking and control	[BOD07], [COL14], [NIA13], [NUR11], [SIL10]
Quality	[CAT09], [CHR01], [COL14], [EBE08], [ERI06], [JIM09], [SIL10]
Organizational distance	[NIA13], [NOL10], [RAL08], [SIL10], [VER14]
Project and Process management	[CAT09], [COL14], [JIM09], [KHA11], [NOL10], [VER14]
Different stakeholders	[DAM02], [PER10], [SIL10], [VER14]
Conflict management	[COL14], [DAM02], [NIA13], [SIL10]
Task assignment	[BOD07], [NIA13], [PER10], [SIL10]
Scope and change Management	[CAT09], [NIA13], [SIL10], [VER14]
Risk management	[JIM09], [NIA13], [SIL10]
Increased project cycle time	[COL14], [RAM07], [YAN04]
Software configuration management	[JIM09], [VER14]
Poor contract management	[KHA11], [VER14]
Poor relationship management	[KHA11], [VER14]
People management	[CHR01], [SIL10]
Schedule management	[BOD07], [SIL10]
Project planning	[CHR01], [SIL10]
Strategic inflexibility	[KHA11]
Turnover	[EBE08]

Table 6: Geographical challenges of team distribution (Category #1)

CHALLENGE	RELATED PRIMARY STUDIES
Socio-cultural differences	[CHR01], [DAM02], [DES10], [EBE08], [HOL06], [JIM09], [KHA11], [LEE06], [MHT12], [MIS11], [MOC01], [NIA13], [NOL10], [NUR11], [PER10], [RAL08], [SIL10], [SOL10], [VER14]
Geographical distance	[CAT09], [DES10], [EBE08], [ERI06], [HER03], [HOL06], [JIM09], [MIS11], [MHT12], [MOC01], [NIA13], [NOL10], [NUR11], [PER10], [RAL08], [SIL10], [YAN04]
Temporal distance	[CAT09] [DAM02], [DES10], [ERI06], [HOL06], [MIS11], [MHT12], [MOC01], [NIA13], [NOL10], [NUR11], [RAL08], [SIL10], [SOL10]
Language barriers	[CHR01], [DAM02], [DES10], [JIM09], [KHA11], [MOC01], [NOL10], [SIL10]
Lack or difference in technology infrastructure	[LEE06], [MOC01], [NIA13], [NOL10], [PER10], [RAL08], [SIL10], [VER14]
Knowledge distance / insufficient knowledge	[EBE08], [KHA11], [MHT12], [RAL08], [SIL10], [VER14]
Organizational distance	[NIA13], [NOL10], [RAL08], [SIL10], [VER14]
Overall visibility / team awareness	[JIM09], [NIA13], [SIL10]
Synchronization / interdependence of work among sites	[BOD07], [CAT09], [MOC01], [SIL10]
Country instability	[KHA11], [VER14]
Different governments, laws, rules and regulations	[SIL10], [VER14]
Ineffective decision-making meetings	[DAM02]
Team building activities	[NIA13]

Table 7: Other challenges of team distribution (Category #1)

CHALLENGE	RELATED PRIMARY STUDIES
Intellectual property rights	[EBE08], [KHA11], [NIA13], [SIL10], [VER14]
Delays / failures in delivery	[EBE08], [KHA11], [VER14]
Vendor opportunistic behavior (lock-in)	[EBE08], [KHA11], [VER14]
Product and product architecture evaluation and validation	[ALI08], [CHR01], [NOL10]
Application of agile practices	[SIL10], [VER14]
Hidden vendor costs	[KHA11], [VER14]
Cost and effort estimations	[COL14], [NIA13]
Identification of roles and responsibilities	[SIL10]

Wage and cost inflation	[EBE08]
Need of office space	[SIL10]
Integration of female gender	[DES10]
Poor supplier services	[EBE08]
Higher change rates	[EBE08]
Requirements elicitation	[SOL10]

Table 8: Challenges of system distribution (Category #2)

CHALLENGE	RELATED PRIMARY STUDIES
System specification / modeling / design	[AUE94], [ERF06], [FEL84], [STE87]
Social impact	[FEL84]
Metrics	[JIN93]
Integration and management of several different technologies and components	[ABD13], [FEL84]
Consolidation of distributed knowledge	[MAN09]

Table 9: Common challenges to team / system distribution (Category #3)

CHALLENGE	RELATED PRIMARY STUDIES
Different local requirements	[DAM02], [LEE06], [VER14]
Testing	[BRA98], [MAT13]
Difference / lack of resources in distributed sites	[ERI06], [FEL84]
Communication	[ABD13], [BOD07], [COL14], [DAM02], [HER03], [JIM09], [KHA11], [LEE06], [MIS11], [MOC01], [NIA13], [NUR11], [PER10], [RAL08], [SIL10], [SOL10], [YAN04], [VER14],

2.4.2 Conclusions from the SMS

The volume of works regarding team distribution project management far outnumbers the volume of distributed systems project management. While it was still possible to raise a good amount of challenges, exclusive of either team or system distribution and on some cases, of a mixed nature, the disproportional high volume of papers focusing on team distribution allows some conclusions to be made about that specific distribution aspect. The same is not true regarding distributed systems.

Geographical, temporal and socio-cultural distances, as well as communications present themselves as the main challenges of team distribution. Other challenges exist however, such as language barriers, team trust, team coordination, knowledge management, etc., all being important enough to receive attention from management for better coping to the global software engineering paradigm.

System specification, modeling and design have been raised as the main challenges brought by system distribution to project management. The low number of relevant papers, and the aging nature of half of them do not allow a definitive conclusion on this matter though, since many of the problems and conclusions presented on these papers could no longer apply to present day´s projects and technologies.

3. INTERVIEW-BASED FIELD STUDY

Considering the results presented in Chapter 2, that DS PM has not been given the deserved attention, we designed an interview-based field study with industry professionals to shed some light into the dynamics of the intersection of the DS and PM subjects. More specifically, we wanted to understand what today challenges of projects involving DS are, and what could be the best countermeasures to deal with such challenges.

A series of semi-structured interviews were set in order to obtain the perception of IT professionals about how DS aspects are handled when managing a project. We expected the data collected during the interviews would be used for assessing the validity of our hypothesis. The interviews were conducted with 16 experienced IT professionals located in Brazil (14) and in the United States (2). We selected the participants based on our network of industry contacts using the level of experience in the IT industry (at least 10 years) and ability to be critical (as perceived by the researchers) as the selection criteria. Their role distribution is as follows: 9 project managers, 2 development leaders, 2 test leaders, 1 business analyst, 1 architect, and 1 IT manager. In average, our participants had: 17.2 years of work experience, 12.5 years of technical work experience, 6.7 years of managerial experience, and 5.8 years of experience with the current employer. The summary of the interviewees sample can be seen in Figures 7, 8 and 9.

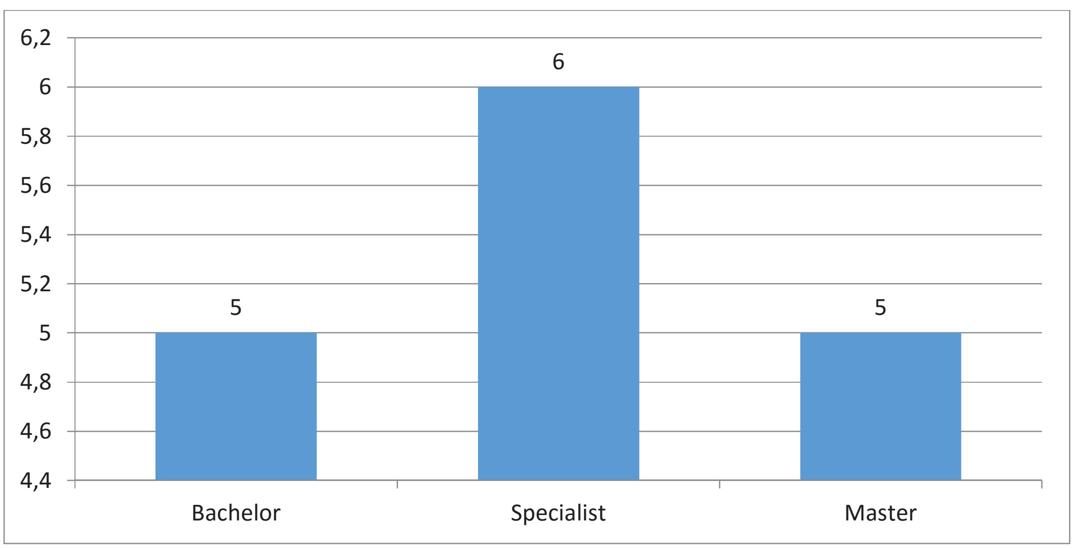

Figure 7: Interviewees' education levels.

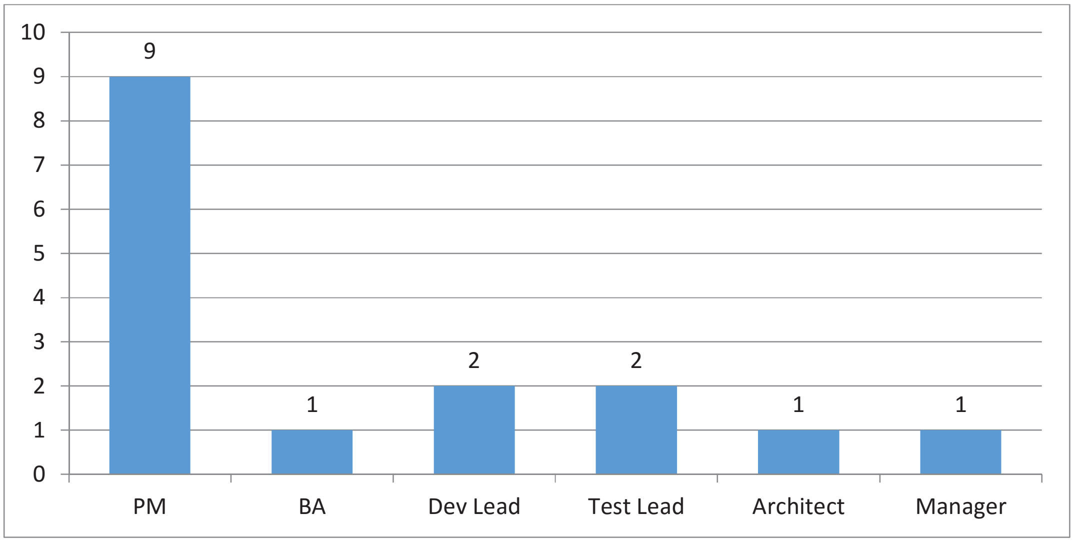

Figure 8: Interviewees' current professional roles.

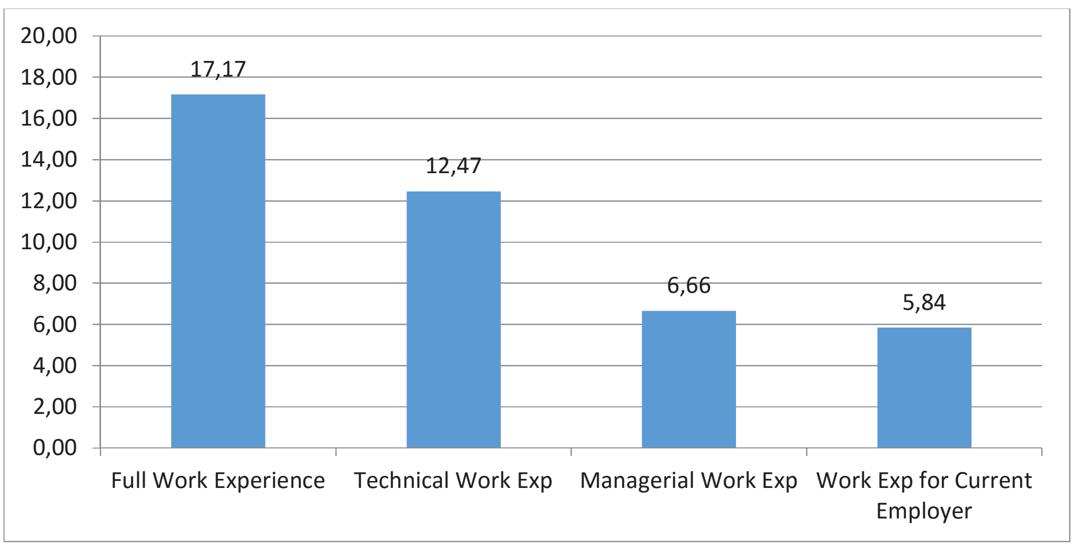

Figure 9: Interviewees professional experience.

A 13-question interview script was defined by the researchers after discussions among them, making use of their professional experiences. This script was defined for evaluating, among other things, the interviewee´s professional background, both technical and managerial, their familiarity with PM and DS, their perception about the challenges brought by these two topics, and also their visibility about possible countermeasures to be used to neutralize or mitigate those challenges. The Research Questions (RQ) are listed below:

RQ1: "*Tell me how you see in practice the Project Manager´s day-to-day involvement on technical aspects of software development projects, such as software architecture, system analysis, system modelling, software configuration, etc.*"

RQ2: "*Now, tell me how you think the ideal Project Manager´s day-to-day involvement on the same technical aspects of software development projects should be.*"

RQ3: "*Now let´s focus on one specific technical aspect. Are you familiar with distributed systems / applications? How would you define them?* "

RQ4: "*Now that we are in sync about the definition for Distributed Systems as well as for Distributed Applications, tell me what you would say is the percentage of all the software development projects you worked on in the last five years that involved distributed systems / applications.*"

RQ5: "*Tell me how you see the role a clear, well-defined architecture plays in facilitating the integration of PMs and team on technical questions, including DS related ones. Please note that by well-defined architecture I mean things like layer-organized, thoroughly-tested, well-documented, pattern-following architectures.*"

RQ6: "*Tell me if you see current state-of-the-art technology (e.g., present day´s development frameworks, integrated development and testing tools suites) facilitating the integration of PMs and team on technical questions, including DS related ones.*"

RQ6.1: "*How do you come to your conclusion? Please try to trace a parallel to 5 or 10 years ago, when such technology could be more immature or could even not be available at all. Is delivering a software development project easier or harder today?"*

RQ7: *"What do you think are the technical challenges (requirements elicitation and specification, implementation, testing and deploying) brought by projects involving DS?"*

RQ8: "*What do you think are the project management (planning, risks, status, procurement, change) challenges brought by projects involving DS?*"

RQ9: "*From all the projects involving DS you worked on and delivered in the last 5 years, what would you say is the percentage of project failure (scope, time, cost overruns or incompatibility, etc.)?*"

RQ9.1: "*On your opinion, why did those projects fail? Would you say there was any reason that could be linked to DS? "*

RQ10: *"If you had the chance to go back in time and fix what went wrong, what actions would you take in order to bring success to those projects involving DS that have failed, according to you? Think of the highest priorities."*

RQ11: "*Suppose you have a magic wand. If you could use it, what would you like to instantly create in order to help you with this kind of project going forward? Would it* *be documentation, frameworks, methods, etc.? Please explain how you think this would help you."*

We voice recorded the interviews and later analyzed the results. Our main findings from our field study are presented in the next sections. Qualitative data is presented, since the interviews had a qualitative nature, but quantitative data is also presented for easier and faster visualization of trends.

3.1. Technical Involvement of Project Managers

The perception of 68,75% of our interviewees (see Figure 10), is that project managers do not get involved with technical aspects in the projects they are managing, thus fully delegating them to the technical team (RQ1). The reasons perceived as culprits for that behavior vary from company structure, lack of technical formation or background, multiple allocations in different projects with different technologies, which in turn makes it impossible for the PM to be fully aware of all different relevant technological details, etc.

Despite this, as shown in Figure 11, 62,5% considered beneficial, project delivery wise, to have project managers that have technical knowledge, hence being able to get much more involved with the technical aspects of projects and, thus, becoming more of decision-makers, being less dependent on their technical team members (RQ2). Other benefits coming from this different behavior would include better ability to provide better project planning and better project risk management, and as such, PMs which fit this parameter are considered by other PMs and their teams to have an upper-hand over the ones who do not.

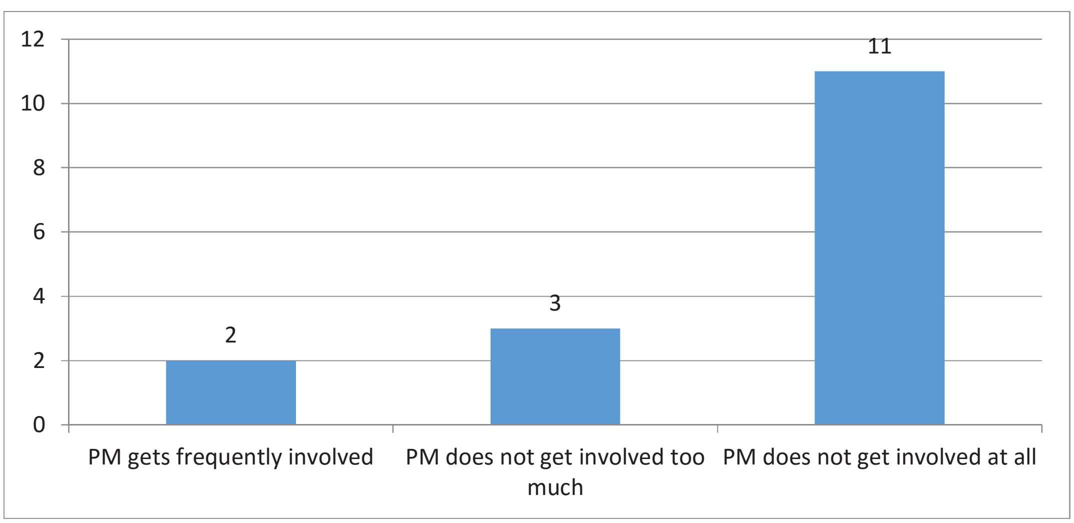

Figure 10: RQ1 – *How is the perceived Project Manager´s day-to-day involvement on technical aspects of software development projects, by number of answers*

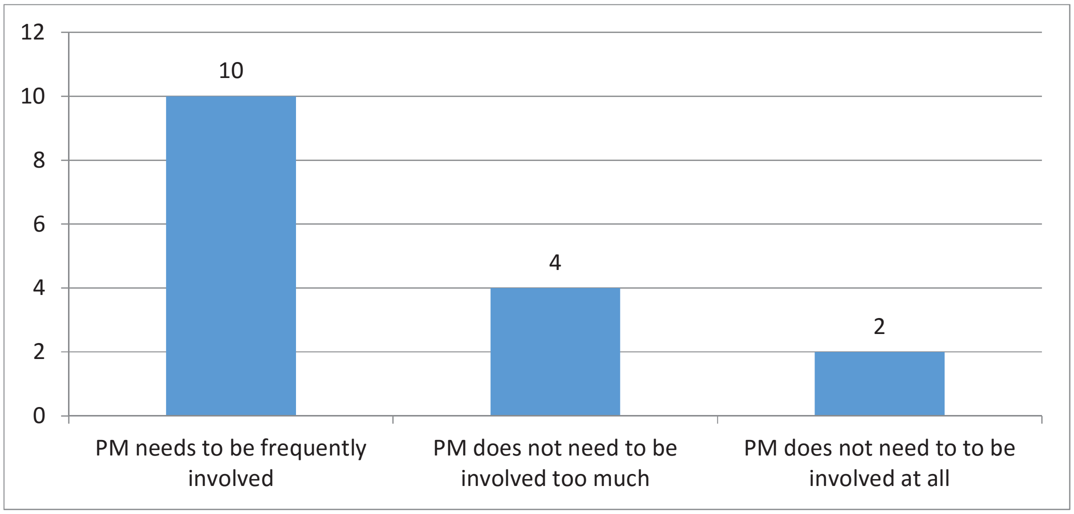

Figure 11: RQ2 - *How should be the ideal Project Manager´s day-to-day involvement on technical aspects of software development projects, by number of answers*

3.2. Awareness of System Distribution

Regarding awareness of what system distribution means (RQ3), 62,5% of interviewees were not even familiar with the term, as shown in Figure 12. A standard DS research explanation, close to what is described in Section 2 was presented, and after having all interviewees confirming they were now acquainted with these two concepts, an average of 84,06% was reached in terms of the percentage of projects involving Distributed Systems / Applications they worked on in the past 5 years (IQ4).

This is further proof of the big relevance the Distributed Systems aspect have in today´s software development projects, yet it allows us to conclude that the high volume of today's software development projects involving DS does make it difficult for even the most experienced professionals to realize how frequently they find themselves inserted in such a context. For them, these are "just projects", where system distribution is a normal, almost default and mandatory aspect, with nothing special attached to it. This in turn constitutes evidence towards the hypothesis of a tendency towards the "abstraction" of the DS feature.

Not only project managers, but also professionals with different roles, including technical ones such as development leader and software architect, are not fully conscious of the environment they are inserted in, and because of that tend to ignore the technical nature of the project they are delivering, increasing the risk of project failure.

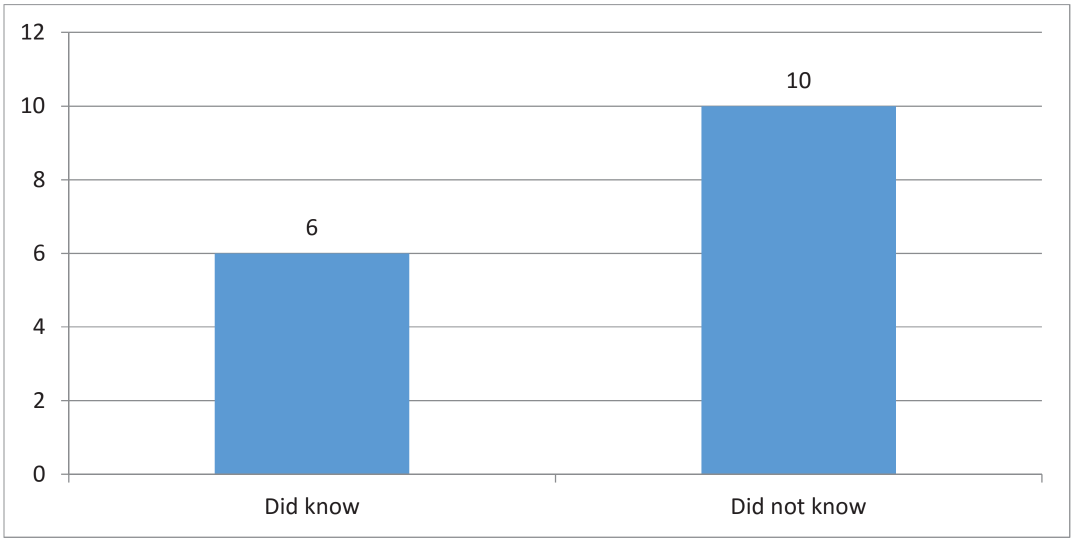

Figure 12: RQ3 – *Interviewees' knowledge prior to the interviews of what system distribution means, by number of answers*

3.3. The Roles Played by System Architecture and Technology in DS Projects

The majority of interviewees believed a solid system architecture to be key for the familiarization of PM and Team with the technical aspects of the projects they were in (RQ5), as shown in Figure 13. The same is true for state-of-the art technology (RQ6), as shown in Figure 14. The reasons for this importance varied per interviewee, but in general they included reasons such as easier traceability of the impact from changes against a highly integrated and distributed system, existence of an information start point for anyone joining the project, better grounds for test coverage planning and execution, enhanced development productivity, etc.

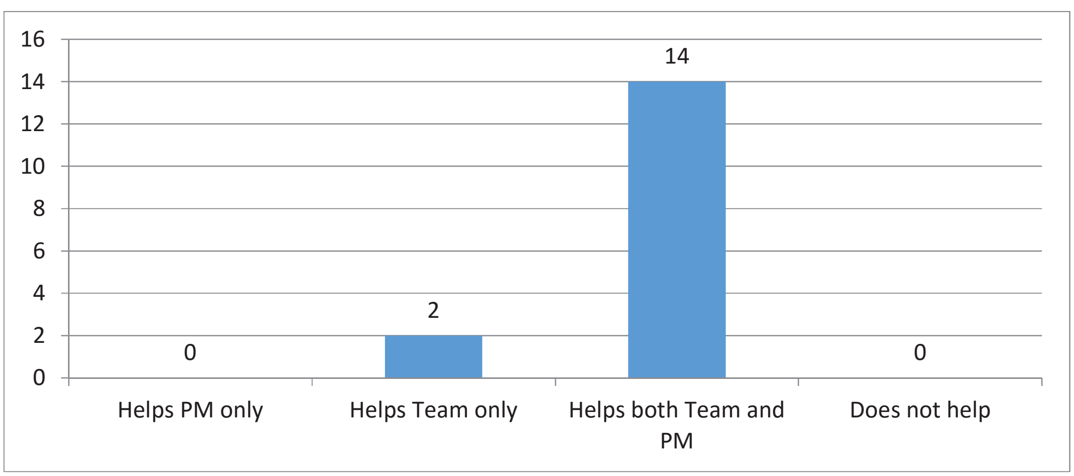

Figure 13: RQ5 – *What is the role a clear, well-defined architecture plays in facilitating the integration of PMs and team on technical questions, including DS related ones, by number of answers*

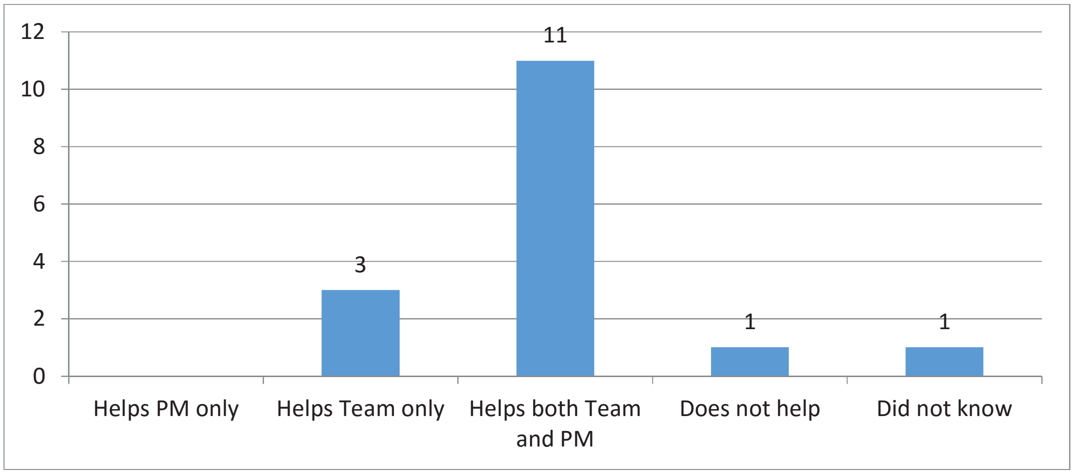

Figure 14: RQ6 – *What is the role a current state-of-the-art technology plays in facilitating the integration of PMs and team on technical questions, including DS related ones, by number of answers*

The majority of the interviewees do believe delivering projects, inclusive of Distributed Systems ones, is easier today (RQ6.1), as shown in Figure 15. The main reasons for that according to them is due to the high-degree of abstractions in existence in today´s software development, such as that of Application Programming Interfaces (API), the diminished learning curve and increased productivity and maintainability proportioned by those abstractions, the greater control over the different phases of a project provided by the different sets of tools being used, the vast information and references available for troubleshooting, easier transition to support, etc.

It is interesting though that some negative side effects, though not strong enough to balance the positive side effects, were also pointed out. Those were issues such as the need to rely on a larger number of specialists, as the ever increasing number of technical options for implementing a solution, such as development frameworks, different tools, etc. make it difficult to work with fewer people, considering the small probability a single individual to be able to dominate all the aspects and technologies needed for delivering alone a modern solution. Also, it was pointed out that the ever increasing complexity of systems and technology has outpaced the evolution of tools and processes used to manage efforts and to deliver results.

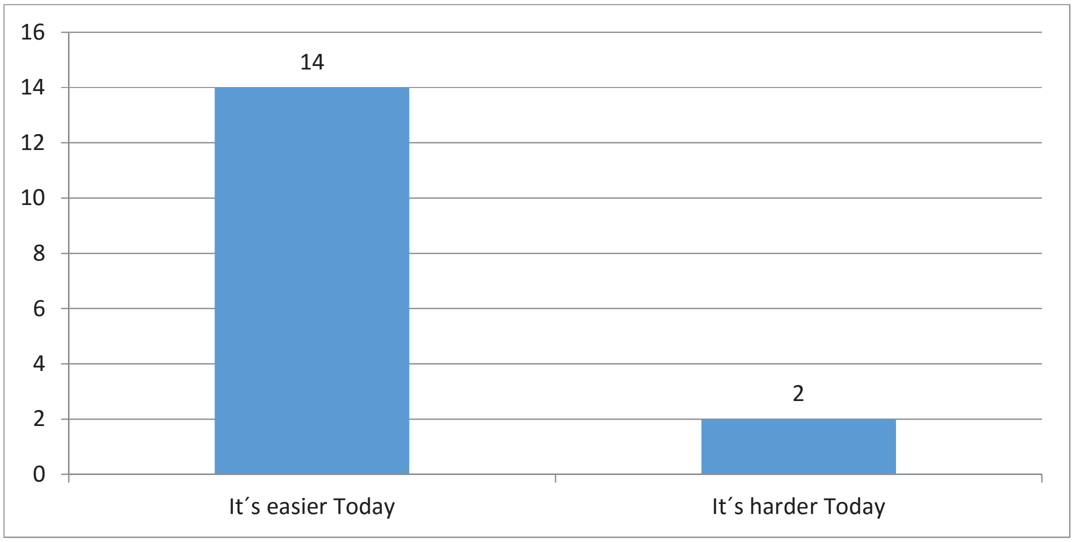

Figure 15: RQ6.1 - *Is delivering a software development project easier or harder today when compared to 5 or 10 years ago?, by number of answers*

3.4. The Challenges from DS Projects

The challenges discussed during the interviews related to DS projects were either related to technical (RQ7) or to managerial (RQ8) aspects of system development. Each interviewee was allowed to provide as many challenges as they could think of, including challenges that were related to a same item, but were still different between each other.

During the post-interview analysis, we first listed separately all challenges raised, and in a second moment we grouped into a single challenge the occurrences that were about the exact same challenge, making sure to have each occurrence counted as one. A third step was performed to group all similar but different challenges into categories, which led to the final categorization seen in Figures 16 and 17. These graphs show the overall number of occurrences for each category, with each interviewee being able to account for the provision of multiple occurrences per category.

Per the cited technical challenges, the most common were related to testing (e.g., cited by 68,75% of the respondents), IT infrastructure (e.g., 56,25%), and integration in distributed applications running on top of DS (31,25%). Other challenges were cited, such as defining the right architecture, convey the right visibility to customers and users of the complexity of DS requirements that meet the desired functionality and performance, etc. One, thus, can realize both functional and non-functional requirements of DS projects are covered by these challenges. The categorized list of the main technical challenges and the number of occurrences for each of them is shown in Figure 16.

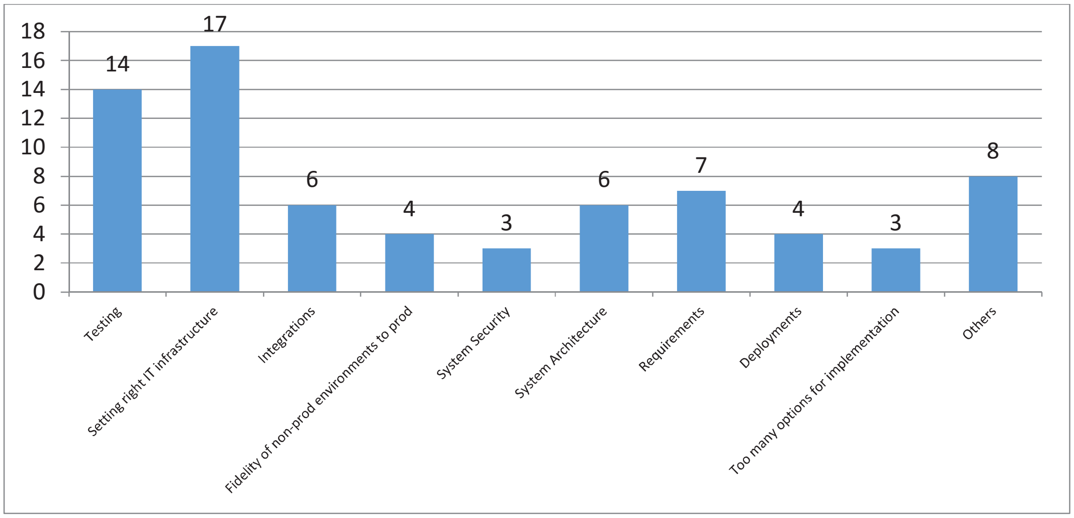

Figure 16: IQ7 - Technical challenges of DS projects.

We also discussed the managerial challenges related to DS projects. One of the most mentioned is related to assembling and keeping a skilled team (e.g., cited by 31,25% of the respondents), since many different technical profiles are needed to cope with the growing number and complexity of DS technologies. Communication was also pointed as a critical issue (e.g., 43,75%), as an interesting but weak causal relation that emerged from the answers, was that the larger and more global the company is, the more it would need large DSs. More likely it would be for those companies to have different teams (possibly distributed) working out different pieces of those large systems. The categorized list of the main managerial challenges and the number of occurrences for each of them is shown in Figure 17.

It is important to highlight that after the interviews, based on the new information available coming directly from the IT industry, the main definition of “system distribution” of our study came to be restricted to two very specific types of solutions, the ones most recurring in the interviewees answers: (i) those distributed regarding their IT infrastructure, e.g. a software distributed between an application server and a database server; and (ii) those distributed among different software, integrated with each other through interfaces or other mechanisms that allow exchanges, such as of data, tokens, etc.

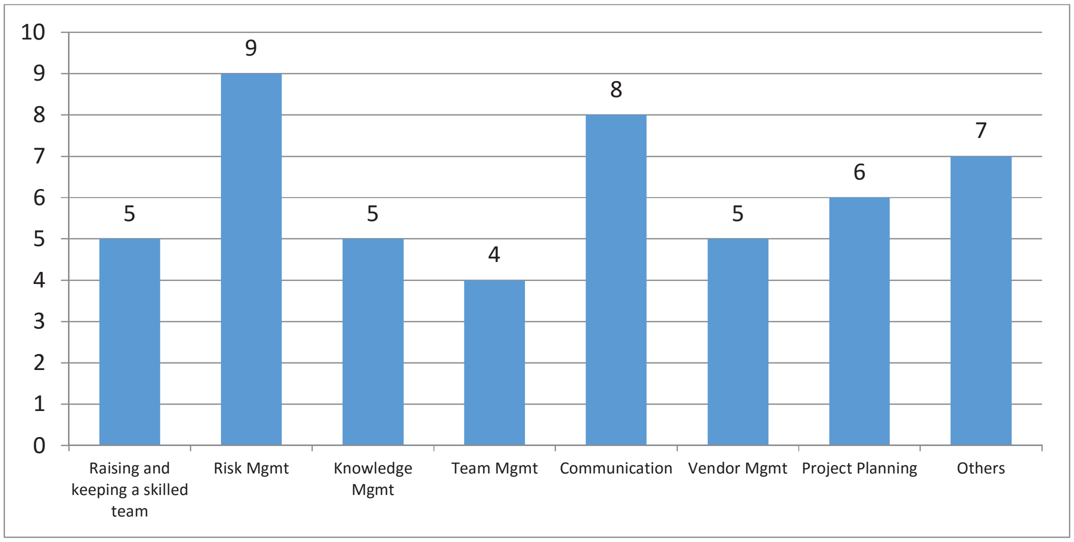

Figure 17: RQ8 - Managerial challenges of DS projects.

3.5. Failed DS Projects and Possible Countermeasures

From the DS projects that the interviewees participated in the last 5 years, an average project failure of 38,44% (RQ9) was reported, having 81,25% of the interviewees claiming, based solely on their perceptions, to see failure reasons that could be linked to the system distribution aspect and the aforementioned technical and managerial challenges (RQ9.1). In this research, we used the definition of project success from [ANA10], which is not restricted only to completing project within time, cost, and meeting scope and quality, but also includes the need to comply to stakeholder requirements.

These reasons included things like lack of technology know-how, bad integration requirements elicitation, too large and too complex distribution that involved too many nodes (and responsible teams for such nodes), difficulty in reproducing a good enough End-To-End (E2E) integration testing, production environment replication issues, environment availability issues, vendor lock-in, etc.

The same aforementioned DS projects failure reasons helped to drive the interviewees´ perceptions of what needed to be done to diminish those issues (RQ10). Their answers included items like better requirements gathering (especially the ones related to integrations), provide better care and full visibility about the IT infrastructure needs, provide better planning and execution of E2E tests, proportionate better team communications considering the multiple teams responsible for different system parts, have a better rationalization about what technology would be best for the project, work out the allocation of best capacitated PM and technical team, etc.

Based on this rationale, the interviewees provided a set of countermeasure choices, listed in Figure 14, they thought that better implemented these items, thus allowing project teams to better deal with the challenges and complexities of DS projects, and as such, increasing their chances of success of delivering those projects (RQ11). The most recurring choice was to have a Software Development Life Cycle (SDLC) specialized in DS as indicated in Figure 18.

An SDLC is a subset of the Project Life Cycle (PLC) [TAY04]. It is focused on accomplishing the product requirements [TAY04]. The PLC is the series of phases a project passes through from its initiation to its closure [PMI13], having its activities have more to do with planning, administration, and leadership [TAY04]. The SDLC activities, on the other hand, are focused on the technical aspects of producing the project deliverables [TAY04].

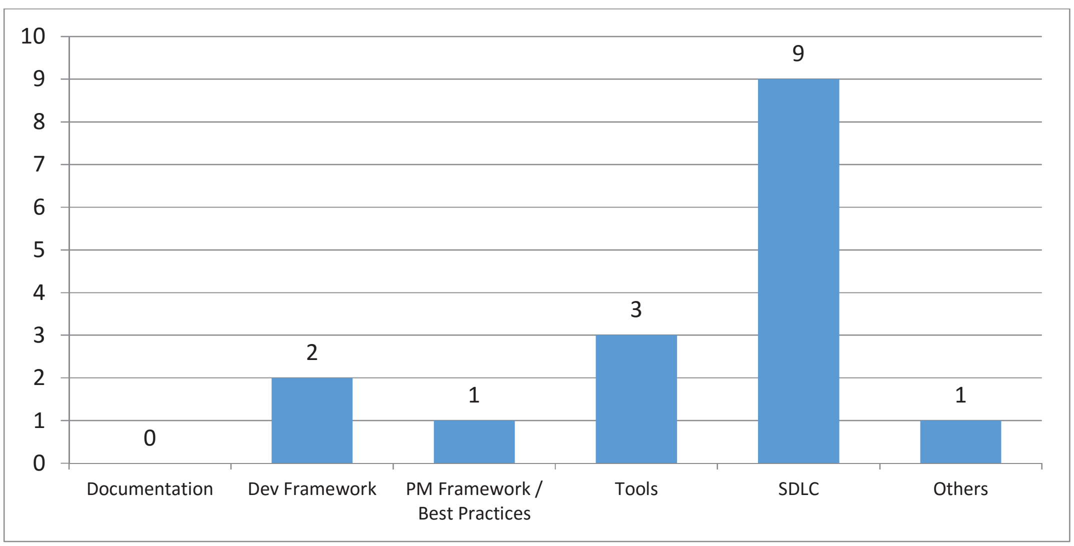

Figure 18: RQ11 - Suggested countermeasures for DS projects.

It is important to highlight that not only our SDLC is a countermeasure by itself, as it was the most recurring one desired by the interviewees, but also that the SDLC implements other countermeasures for the challenges raised by these same interviewees. This is done through the different activities, artifacts and associated practices, all of which are described in chapter 4. Examples of such countermeasure implementation can be found in chapter 4, in the practices that include exemplification figures.

4. AN APPROACH FOR DS SDLC

Based on the low number of literature in DS PM and on the findings from our field study, we decided to propose an SDLC optimized for running Software Development Projects involving DS. The top-level structure of our SDLC contains its key phases, activities, and deliverables, all adherent to the generic SDLC process defined by [TAY04]:

- Product requirements, feasibility analysis, product scope and systems architecture;
- Preliminary design, design approval, detailed design;
- Construction of the system;
- Unit, system and integration testing;
- Delivering the system.

Because of the generic nature of our SDLC, it is important to highlight that it does not represent our proposal's main contribution, since we do not intend to recreate something already proven to be effective. Our main contribution is thereby on the differentiated practices we are proposing, and that should in turn be used within the organized structure of our SDLC. These practices are needed adaptations on already well-known and widely disseminated items, such as a Project Architecture Document or a System Requirements Document for example, making them highly tailored for better use within DS Projects.

Our Phase-Activity-Deliverable structure has to be viewed as a non-prescriptionary guide. It should be easily mapped against different SDLC versions already in use in IT companies around the world, which means they could keep using their own processes while simply adding our proposed practices to them. It is also important to understand that we specified only activities that are mostly impacted by our tailoring, avoiding to specify activities that would not be impacted or be done differently from how they are done in any type of project.

4.1. Overall View of Our SDLC

The proposed SDLC is designed for an optimal implementation under a Waterfall [PRE14] regime, because of this cycle's more traditional nature and its widespread use up to this day. Adaptations are also proposed for use under Scrum [SUT15], since one cannot ignore its growing use in today's industry, which is reflected upon the latest report from [VER15]. A standard representation of Waterfall and Scrum cycles is represented in Figure 19, allowing an initial visualization of the differences between them.

In the next sections we present our proposed SDLC in detail. For the case of our diagrams, we use Unified Modeling Language (UML) [BOO05] compliant notations. We made one customization of these notations, related to the representation in the diagrams of inputs and outputs for each activity. Inputs are represented on top left and outputs on bottom right of each activity. We also list our suggested practices, and provide some practical examples for an easier contextualization. For our software integration related proposal (type of solution 'ii' as defined in Section 3.4), examples are based on software that are integrated data-wise (exchange of data through data interfaces, for example).

We do acknowledge that other types of integration exist, such as token exchanges instead of data exchange, just to mention one type of many different available. We do not intend to map all possible integration scenarios, thus what is important is to understand the principle behind each of our practices, which in turn should be further adapted to the reality of other integration types.

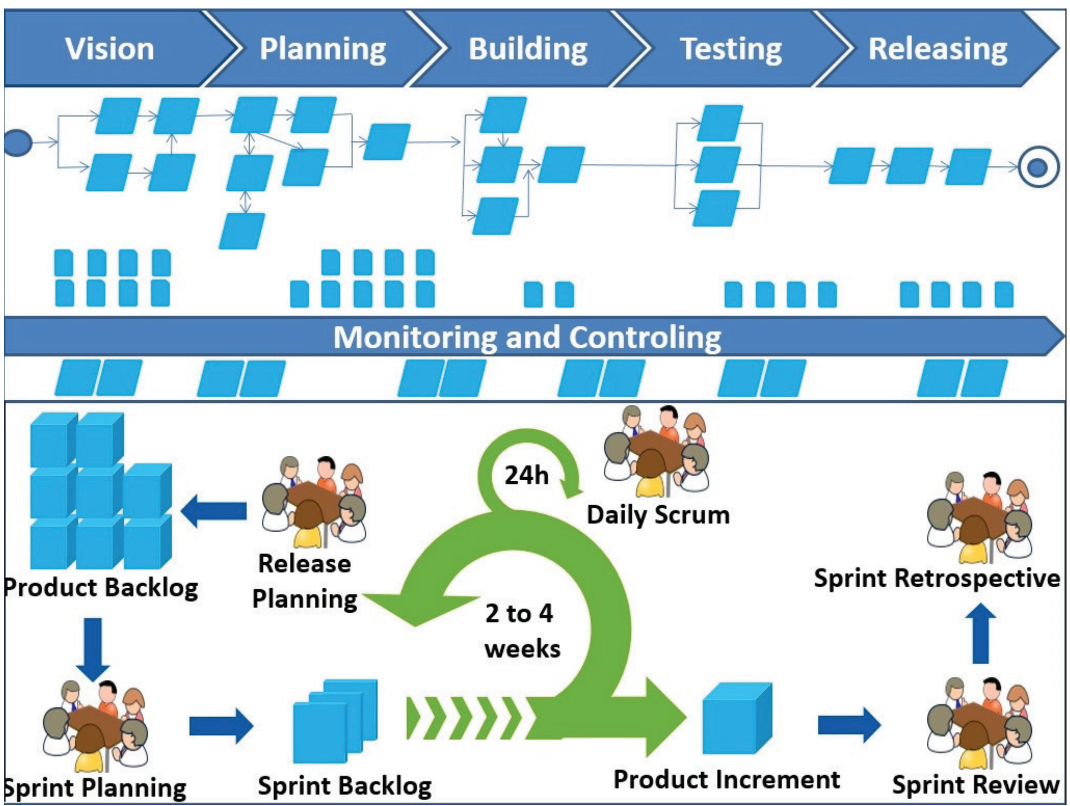

Figure 19: Overall view of standard SDLC cycles, Waterfall and Scrum.

4.1.1. Vision Phase

In the Vision Phase, the Project is initiated through the assignment of a Project Manager and initial project team. Preliminary project planning is made by obtaining high-level time and cost estimates, in response to business requirements being presented, as well as to existing project constraints. Consciousness about the system requirements, especially the non-functional ones, is vital, as they will drive the definition for the system distribution solution, thus generating visibility about the challenges associated with it. Due countermeasures can be thought of, planned and implemented, as earlier as possible in the project.

We did not formally define a precedent phase to the Vision phase. We understand it may exist though, formally or informally, in a wide number of companies that are potential adopters of our SDLC and practices. In this case, the information coming from this extra phase is acceptable as input to the Vision phase, having it already been mapped in the Vision phase activity breakdown section presented next. An overview of the activity flow of Vision Phase can be seen in Figure 20.

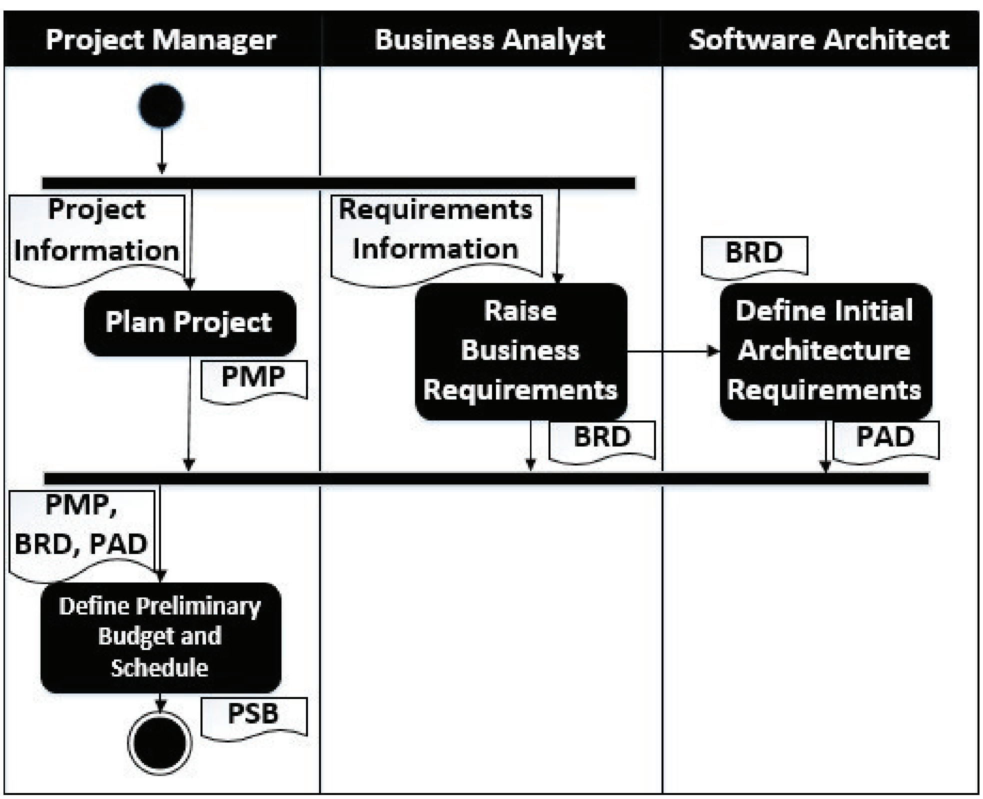

Figure 20: Vision phase of proposed Waterfall cycle.

4.1.1.1. Vision Phase Activities Breakdown and Description

Activity: Plan Project		
Roles	Primary • Project Manager.	Secondary • Business Analyst; • System Analyst; • Software Architect; • Developer; • Tester; • Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Project Information: - Pre-project data, if it exists; - Project roster; - Time and cost constraints; - Quality constraints; - Work estimates; - Historical data; - Project decisions.	Optional • None.
Outputs	Project Management Plan (PMP).	
Description	Through this activity, the initial Project planning occurs. While it is still not possible to fully plan the Project until its end, at this point it is already viable to plan the entire Vision Phase, and also to define high-level, initial milestones for the remaining of the Project. Auxiliary management plans, such as the Management Plans for Communication, Stakeholder and Time are all part of the Project Management Plan, which is the "master plan", an output of this phase. During the execution of this activity, the Project Manager will rely on diverse Project Information coming from various sources, such as data from a pre-project phase, if it exists, estimates gathered directly from the project team, historical data coming from different Project Stakeholders (customers, users, Project Management Office and others), constraints of various natures, etc.	

Table 10: Details of "Plan Project" Activity.

Activity: Raise Business Requirements		
Roles	Primary • Business Analyst.	Secondary • System Analyst; • Software Architect; • Developer; • Tester; • Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Requirements Information: - Functional requirements; - Non-functional requirements.	Optional • None.
Outputs	Business Requirements Specification (BRD).	

Description	This task is about the elicitation of business requirements (not system requirements), both functional and non-functional, for the product (system) being developed in the project. Special attention needs to be provided to the business non-Functional requirements, as they will highly impact the solution that will be later proposed, and thus, the system distribution type and scale that will be part of the solution. The BRD, which is the output of this phase, must be written in business terminology, not system terminology, allowing business users and customers to fully read and understand it.	

Table 11: Details of “Raise Business Requirements” Activity.

Activity: Define Initial Architecture Requirements		
Roles	Primary • Software Architect.	Secondary • System Analyst; • Developer; • Tester; • Infrastructure Analyst; • Project Manager.
Inputs	Mandatory • Business Requirements Document (BRD).	Optional • None.
Outputs	Project Architecture Document (PAD).	
Description	This activity is related to the definition of the initial software architecture of the Project. While the decisions taken at this point are not definitive, still being subject to change, as functional and non-functional requirements mature, they already play a key role in assessing the project’s feasibility regarding its solution to the functional and non-functional requirements. Although this assessment is done in high-level at this point, if well-done it can be a decisive factor regarding the continuation of a project to next phases, as depending on the case, the solution may already prove to be too expensive to acquire, or too disconnected from the requirements, in case a cheaper alternative is the only one at reach.	

Table 12: Details of “Define Initial Architecture Requirements” Activity.

Activity: Define Preliminary Budget and Schedule		
Roles	Primary • Project Manager.	Secondary • Business Analyst • System Analyst • Software Architect • Developer • Tester • Infrastructure Analyst
Inputs	Mandatory • Project Management Plan (PMP); • Business Requirements Document (BRD); • Project Architecture Document (PAD).	Optional • None.
Outputs	Project Schedule and Budget (PSB).	
Description	In this activity, the Project Manager, with input from other roles as well as from the contents of BRD, PAD and PMP, plans the initial Project budget and schedule. This is	

	not supposed to be the final version of the PSB, which means that at this point, there is no final commitment from the Project Manager regarding how much the Project will cost and how long it will take, as both requirements and solution might still change. It is important though to be able to compare the estimates obtained here to the final ones that will be available later in the Project, as this is an indicator regarding how much the requirements have evolved and how the solution discussed during this phase differs from the final solution, which in turn will allow for better planning during the next projects.

Table 13: Details of "Define Preliminary Budget and Schedule" Activity.

4.1.1.2. Our Vision Phase Recommended Practices for Waterfall

- Business Requirements Document (BRD) should have a section entitled "Business Integrations", which is filled in with key details about all identifiable integrations at a business level (business process, data flow, integration class, etc.). An overview is provided in figure 21. The idea is to promote an early identification of all integration needs, without compromising the business orientation of the BRD;

BI ID	Business Process Being Integrated	Data needed for Integration	Target of Integration (Department)	Source of Integration (Department)	System of Record	Integration Classification
BI 01	Car Assembly	Car configuration	Engineering	Sales department	Car Configurator	Class 1 (mandatory)
BI 02	Car Assembly	SKU Information	Engineering	Engineering	Skuvault	Class 1
BI 03	Invoicing	Electronic Invoice details	Engineering	Invoicing	Oracle	Class 3 (Desirable)
BI 04	Product Shipping	Customer address	Engineering	Sales department	SalesOnline	Class 2 (Important)

Figure 21: Proposed "Business Integration" section of BRD (exemplification).

- Due diligence should be provided to the documentation of Non-Functional Requirements (NFR) that are linkable to system distribution. They should be documented in the BRD for the application being developed and for each of its integrations. These NFRs will thus include items such as level of availability needed, number of simultaneous connections that will exist in any given time, data volume being exchanged among applications, data exchange periodicity, etc.). This will help in deciding the level of system distribution to be adopted, as well as will help in providing the users and customers with understanding about their requirements demand the system distribution;
- Project Architecture Document (PAD) should have a DS Section, containing visual, incremental architecture information, as shown in figure 22.

Business requirements are mapped against integration / infrastructure requirements. The idea is to help not only the project team, but mainly the users and customers in understanding how the system distribution characteristic is applied to project, and how complex this can truly become. This is achieved by directly mapping high-level business requirements to system distribution and infrastructure requirements. Diagrams are used, with each view adding more details;

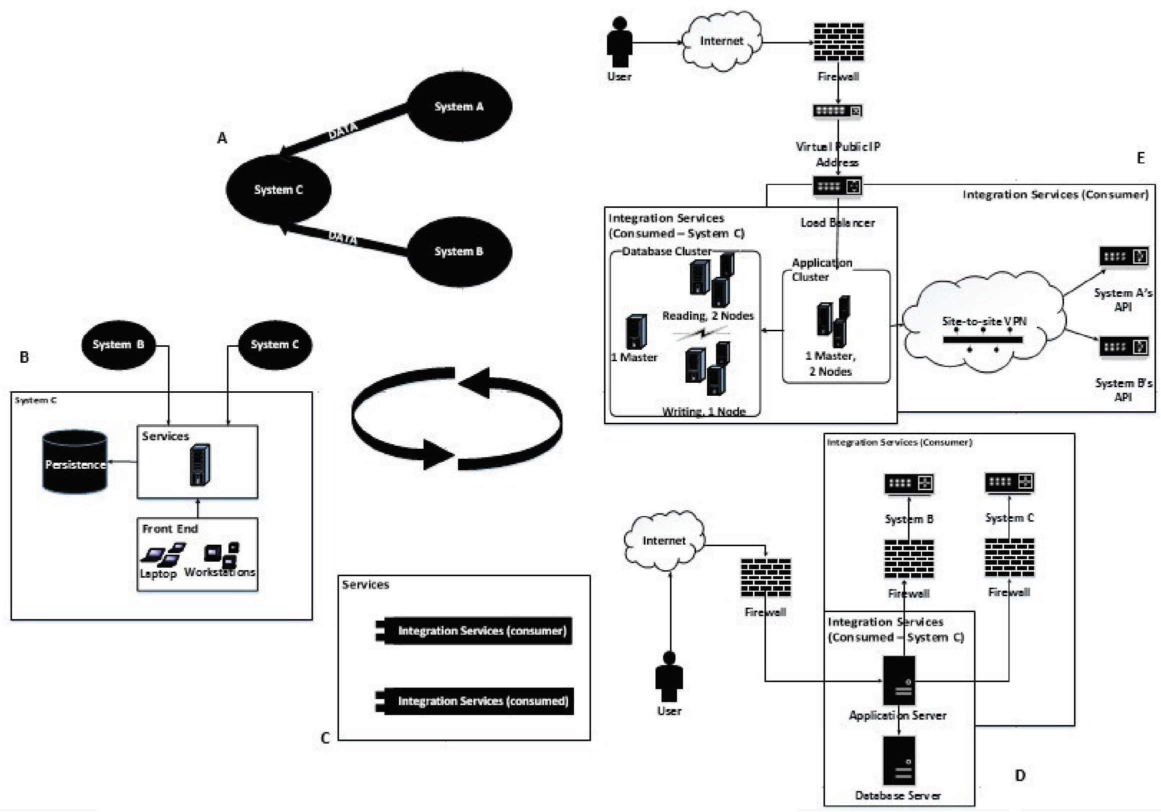

Figure 22: DS Section of PAD, with visual and incremental DS information (exemplification).

- PAD should include applicable integration/infrastructure technical information, such as data format, data contract, security measures, strategy for handling errors, log strategy, etc.;
- Due care is provided for Project Management Plan (PMP) auxiliary plans, such as Stakeholder and Communication Plans. A customer, a technical and a management liaisons should be appointed for each integration. This allows for better identification of stakeholders, thus better guaranteeing their involvement right from the beginning of the project onwards;

- Risk Register (RR) should start with a default list of DS risks, as shown in figure 23. Each of these risks, if applicable to the project, must have at least one action planned for dealing with it. This forces the team to face the issues instead of postponing actions directed at them. The list is continually refined by Project Management Office (PMO), a Project Stakeholder, through feedback coming from live projects. The PMO will also make sure to archive it and make it available for upcoming projects, thus recycling knowledge;

Risk Description				
Id	**Description**	**Condition**	**Consequence**	**Classification**
rsk001	Late identification on business / system interlocks	Identification of business / system interlocks during planning phase or later	Rework regarding requirements elicitation and specification Need for solution revalidation	Process
rsk002	Impossibility of production infrastructure replication in non-prod	Production environment is too big and too complex, which makes it too difficult or too costly to reproduce the same specs in non-production	Performance in non-production does not reflect what will be found in production. Some defects are not detected in non-prod, but will surface in production.	IT Infrastructure
rsk003	Complexity on reproducing E2E testing	Difficulty in having all the required datasets on time and with enough data variation to allow the testing of valid E2E scenarios.	Performance in non-production does not reflect what will be found in production. Some defects are not detected in non-prod, but will surface in production.	IT Infrastructure

Figure 23: Risk Register, with some default DS risks (exemplification).

- All aforementioned artifacts are potential inputs for the Project Schedule and Budget (PSB). The Project Manager should use the information contained on them in order to have a more realistic plan. This is especially important regarding the BRD requirements, as they represent the features that will be part of the final product, as well as the PAD, as it details the architecture, with special characteristics and constraints. These will drive project decisions, thus directly impacting the project budget and schedule;
- PMP, BRD and PSB should be baselined at this point. Although they still do not represent the final commitment from the Project Manager and the

Project Team towards definitive schedule, budget, quality, scope, etc., they do represent the current visibility about what the project requirements are and how the project team will be delivering solutions to those. This visibility will be refined in the upcoming phase, and this baseline will be able to be compared to the new baseline, to be established in the end of the planning phase, thus providing a measure of how the requirements and the solution to those requirements changed and evolved from one phase to another. The same is true about schedule, cost and other project constraints. This information will thus be usable by upcoming projects for better planning and execution.

4.1.1.3. Alignment of Our Vision Phase Recommended Practices to Waterfall

- Provision of detailed visibility around integrations / infrastructure demands, as early as possible;
- Better stakeholder identification, reducing the chances of late engagement and Change Requests;
- Helps all stakeholders in setting up their new mindset about the true complexities of their project, as early as possible;
- Schedule and budget are more realistic, as the distribution characteristic is now considered.

4.1.1.4. Adapting Our Vision Phase Recommended Practices to Scrum

- The Product Owner should already identify the business integration needs before the Project starts. This will already provide key information for the initial Release Planning (if in use in the project) and Sprint Planning meetings;
- The identified integration requirements should be treated as regular requirements, added to the Product Backlog and prioritized by the Product Owner, according to their perceived value. In this work, we propose the *user stories* requirements format. This will help in avoiding common pitfalls found in distributed systems Scrum projects, such as having integration requirements and associated implementation tasks becoming “lost” in the middle of functional or other requirements;

- The integration user stories are discussed during the first project meeting, usually the Release Planning one. This in turn should bring forward the discussion around the infrastructure requirements, needed for supporting not only the regular development work, but also for supporting these integrations;
- Infrastructure and support teams are encouraged to be on-board the discussion already, as early as possible in the project. As there is a strong trend to have everything moving a lot faster in Scrum projects than in Waterfall, it becomes crucial to have these teams on-board for delivering the right solution in the expected pace, and with the right quality levels;
- The identified infrastructure requirements should be treated as regular requirements, added to the Product Backlog and prioritized by the Product Owner, according to their perceived value. In this work, we propose the *user stories* requirements format. This will help in avoiding common pitfalls found in distributed systems Scrum projects, such as having infrastructure requirements and associated implementation tasks becoming "lost" in the middle of functional or other requirements;
- Definition Of Ready (DOR) should take in consideration the system distribution characteristics of the project in question. For example, it could include "complete data contract being available" and/or "data sample being available". This will help in avoiding common pitfalls found in Scrum distributed system projects, such as work stops and/or rework due to the lack of the availability for some basic work items that are enablers for the execution of project work. These items, such as data masses for example, tend to be overlooked or ignored in real projects until the time they are needed, possibly causing delays or other issues.

4.1.1.5. Alignment of Our Vision Phase Adapted Practices to Scrum

- "All" project aspects really become visible to everyone at all times, including the ones related to system distribution, which tended to be "suppressed" before;
- Delivered functionalities to customer will tend to be more stable, as DS projects' key characteristics will receive proper attention. Value delivered

and perceived shall increase, as all requirements now receive proper attention.

4.1.2. Planning Phase

In this Phase, one executes final project planning based on the system requirements, which are based in the final business requirements available since the end of Vision Phase. A "translation" of these business requirements into system requirements occurs in this phase, which allows for a much improved visibility around the technical needs of the project. Architecture and infrastructure requirements full assessments are now possible, and are done for further software and hardware specifications that will meet the project needs.

Full system design is now possible, also becoming complete in the end of this phase. Another output of this phase is the complete project test strategy, which shall be later employed for validating the completed system being delivered regarding its adherence to the requirements. An overview of the activity flow of Planning Phase can be seen in Figure 24.

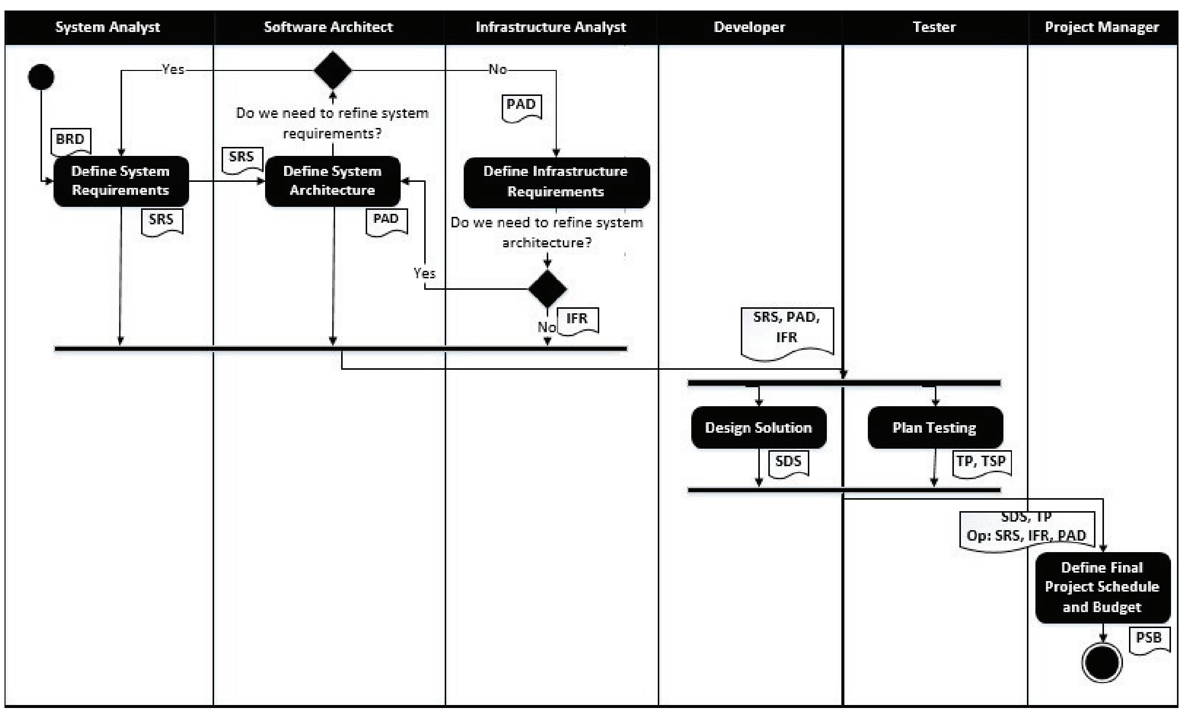

Figure 24: Planning phase of proposed Waterfall cycle.

4.1.2.1. Planning Phase Activities Breakdown and Description

Activity: Define System Requirements		
Roles	Primary • System Analyst.	Secondary • Business Analyst; • Software Architect; • Developer; • Tester; • Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Business Requirements Document (BRD); • Project Architecture Document (PAD).	Optional • None.
Outputs	System Requirements Specification (SRS).	
Description	In this activity, the business requirements from the BRD are "translated" into system requirements, feeding the SRS document. All business needs are re-written in a technical format, understandable by developers and testers alike. Use cases are one of the many possible formats. The document will thus state how the behavior of the system will support each of the business needs. System distribution obviously impacts system behavior and the degree to which the system will conform to business requirements, so careful attention should be provided here in order to specify system requirements that are in line with what can be accomplished in the project, given its known constraints, such as the ones related time, cost, etc.	

Table 14: Details of "Define System Requirements" Activity.

Activity: Define System Architecture		
Roles	Primary • Software Architect.	Secondary • Business Analyst; • System Analyst; • Developer; • Tester; • Infrastructure Analyst; • Project Manager.
Inputs	Mandatory • System Requirements Specification (SRS); • Project Architecture Document.	Optional • None.
Outputs	Project Architecture Document (PAD).	
Description	In this activity, the final refinement and definition of the system architecture is done based on the more mature requirements present in the SRS. Special attention needs to be provided for the system distribution requirements, as they will affect the entire solution's performance and behavior.	

Table 15: Details of "Define System Architecture" Activity.

Activity: Define Infrastructure Requirements		
Roles	Primary	Secondary

	• Infrastructure Analyst.	• Software Architect; • System Analyst; • Developer; • Tester.
Inputs	Mandatory • System Requirements Specification (SRS); • Project Architecture Document (PAD).	Optional • None.
Outputs	Infrastructure Document (IFR).	
Description	In this activity, the infrastructure requirements are compiled in direct response to the requirements in the SRS as well as the defined system architecture in the PAD. Special attention needs to be provided to the infrastructure parameters that influence system distribution. They need to tie out well to the business functional, and especially the non-functional requirements, so that the users will obtain a software that conforms to their needs.	

Table 16: Details of "Define Infrastructure Requirements" Activity.

Activity: Design Solution		
Roles	Primary • Developer.	Secondary • Software Architect; • System Analyst; • Tester; • Infrastructure Analyst.
Inputs	Mandatory • System Requirements Specification (SRS); • Project Architecture Document.	Optional • None.
Outputs	System Design Specification (SDS).	
Description	This activity is related to the final design of the system before system construction, transcribing the system requirements (both functional and non-functional) specifications into the group of components that, when integrated to each other, will compose some sort of detailed blueprint for the system construction and testing. This blueprint needs to be strictly adherent to the previously defined system architecture, which in turn will guarantee that the previously raised system distribution architectural details will remain respected.	

Table 17: Details of "Design Solution" Activity.

Activity: Plan Testing		
Roles	Primary • Tester.	Secondary • System Analyst; • Developer; • Project Stakeholder.
Inputs	Mandatory • System Requirements Specification (SRS); • Project Architecture Document (PAD); • Infrastructure Document (IFR).	Optional • None.

Outputs	• Test Plan (TP); • Test Specification (TSP).	
Description	This Activity is related to the planning of test execution. The Test Plan, besides regular contents, such as test coverage (what requirements will be tested), overall test strategy that will be used for verifying these requirements (regression, automation, etc.), test roster, etc., will also contain the strategy regarding production environment replication into the non-production test environments, encompassing system distribution related questions, such as requirements for End-To-End testing execution, data masses to be used, etc. The Test Specification (TSP) will detail what scenarios will be tested and how they will be tested, for each requirement included in the TP.	

Table 18: Details of "Plan Testing" Activity.

Activity: Define Final Project Schedule and Budget		
Roles	Primary • Project Manager.	Secondary • Business Analyst; • System Analyst; • Software Architect; • Developer; • Tester; • Project Stakeholder.
Inputs	Mandatory • System Design Specification (SDS); • Test Plan.	Optional • System Requirements Specification (SRS); • Project Architecture Document (PAD); • Infrastructure Document (IFR).
Outputs	• Project Schedule and Budget (PSB) 2.0	
Description	This Activity is related to the generation of the final project budget and schedule. It will be based mainly in the detailed information about the final system design, compliant to the architectural definitions. System distribution decisions are highly-regarded in the definition of the final project schedule and budget, as this characteristic not only directly impacts the system adherence to requirements and architectural solutions, but will also play a vital role in the assembly of the Information Technology (IT) environments destined to development, testing and production purposes. The costs and schedule associated with these items may or may not make the project not feasible.	

Table 19: Details of "Define Final Project Schedule and Budget" Activity.

4.1.2.2. Our Planning Vision Phase Recommended Practices for Waterfall

- Another view will be added to the PAD artifact, for registering the detailed business process flows that will be implemented in the solution. Swimlane diagrams can be used for fulfilling this purpose;
- The Infrastructure Document (IFR) should be created for registering detailed information about the required infrastructure. Specific needs for hardware, software and networking are mapped, especially the ones

affecting system distribution, such as servers' latencies and locations, ports, protocols, etc.;

- RR should be updated with newly emerging risks, such as DS related ones;
- System Requirement Specification (SRS) must be created and kept in close alignment with BRD and PAD, thus making sure no previously raised system distribution key definitions are lost. They should instead only be "extended" in the SRS, being "translated" from business requirements to expected system behavior, thus making it easier to elaborate the SRS;
- Test Plan (TP) must include detailed info about the needed environments, data masses, log testing etc. It also could include the plan for test environment redundancy, in case part of tests are located on the Project's critical path;
- Test Specification (TSP) must be created and kept in close alignment with SRS, thus making sure no previously raised system distribution key definitions are lost. They instead should only be "extended" in the TSP, thus making it easier to elaborate it;
- System Design Specification (SDS) must be created and kept in close alignment with the SRS, thus making sure no previously raised system distribution key definitions are lost. They should instead only be "extended" in the SDS thus making it easier to elaborate it;
- SRS, PMP and PSB are updated and re-baselined.

4.1.2.3. Alignment of Our Planning Phase Recommended Practices to Waterfall

- End of Planning phase has all major solution specifications and a complete design that takes into account the system distribution characteristics of the project. This is done through carefully tying out requirements between BRD, PAD, SRS and SDS;
- Improved visibility acquired regarding what are the main technical constraints and risks for the rest of the project, before execution.

4.1.2.4. Adapting Our Planning Phase Recommended Practices to Scrum

- For clear validation of system distribution characteristic, there should be acceptance criterion created for each infrastructure / integration story. For example, in an data integration project, they could include:

    - What should be the systems' behavior when the integrations are and are not available?
    - What should be the systems' behavior when the data contract is or is not being respected, regarding for example, data consumption and data transformation?
  - Integration / infrastructure scope continue to be treated as regular user stories and now are added into a Sprint Planning scope, if Definition Of Ready (DOR) criterion are met.

4.1.2.5. Alignment of Our Planning Phase Adapted Practices to Scrum

  - Clear prioritization of Integration and Infrastructure aspects in relation to regular software requirements, all based on their now perceived value for the solution;
  - Raised DS acceptance criterion will later be used during development and testing cycles. Due importance is provided to the validation of the system distribution key project characteristic.

4.1.3. Building Phase

The Building Phase is where most of the DS SDLC work is done. It is the phase where actual product build takes place. The project team, which includes the infrastructure team, will assemble both the required Non-Production and Production infrastructures, as well as will create the software product through the generation of the system source code. Developer level testing is performed for raising the stability of the system build that will be handed over to the test team in the next DS SDLC phase.

The finished Test Cases that are to be used during Software Integration Test (SIT) and User Acceptance Test (UAT), are also an output of this phase. They are created by who will perform those tests, in this case the test team and the business users team respectively. An overview of the activity flow of Building Phase can be seen in Figure 25.

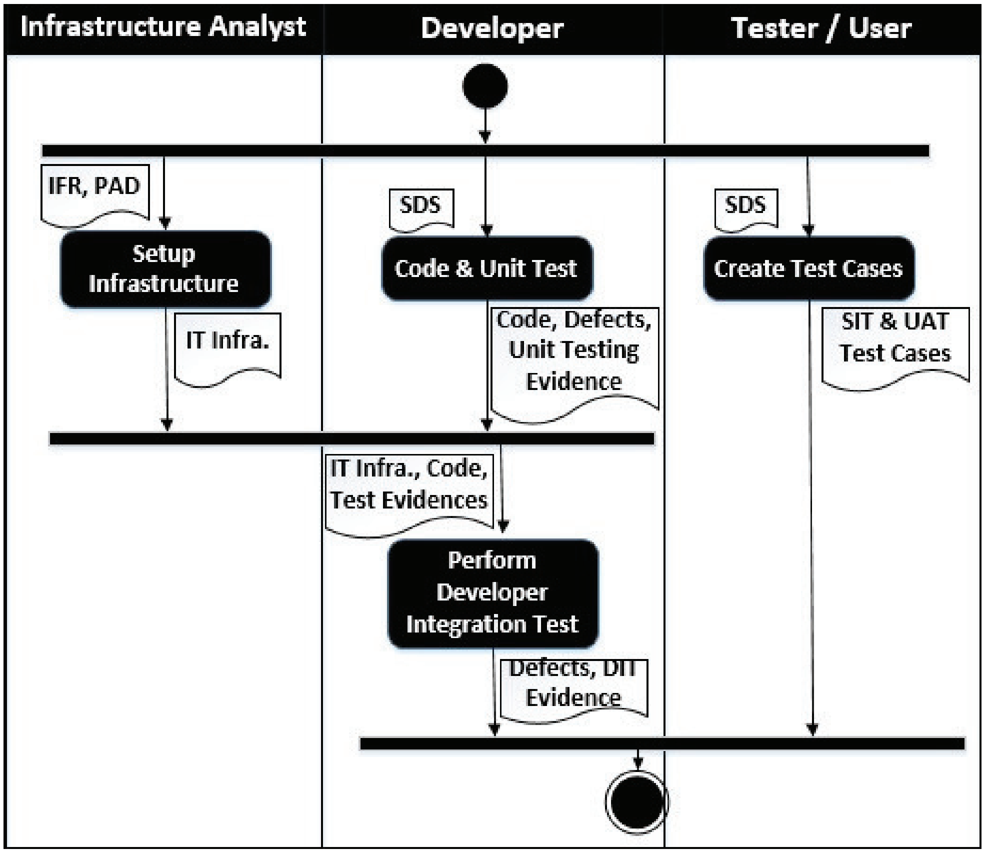

Figure 25: Building phase of proposed Waterfall cycle.

4.1.3.1. Building Phase Activities Breakdown and Description

Activity: Setup Infrastructure		
Roles	Primary • Infrastructure Analyst.	Secondary • Software Architect; • Developer; • Tester.
Inputs	Mandatory • Infrastructure Document (IFR); • Project Architecture Document.	Optional • System Design Specification (SDS).
Outputs	• IT Infrastructure.	
Description	This activity is related to the assembling of the Information Technology infrastructure, both non-production and production versions, required for subsequent project execution. While the actual non-production infrastructure will be put in place through this activity, the production infrastructure will be planned, being put in place only during Releasing phase.	

Table 20: Details of "Setup Infrastructure" Activity.

Activity: Code and Unit Test		
Roles	Primary • Developer.	Secondary • Tester;

		• Project Stakeholder.
Inputs	Mandatory • System Design Specification (SDS).	Optional • Project Architecture Document (PAD).
Outputs	• Source Code; • Unit Testing Evidence.	
Description	This activity is related to the actual codification of individual component specified in the SDS. The implementation of these components is required for assembling the system, which will be the final product of the project. Unit testing, performed by each developer in charge of each component, is also included in this activity.	

Table 21: Details of “Code and Unit Test” Activity.

Activity: Create Test Cases		
Roles	Primary • Tester (SIT Test Cases); • Business Analyst (UAT Test Cases).	Secondary • Developer; • Project Stakeholder.
Inputs	Mandatory • Test Plan (TP); • Test Specification (TSP); • System Design Specification (SDS).	Optional • Project Architecture Document (PAD); • Infrastructure Document (IFR).
Outputs	• System Integrated Testing (SIT) Test Cases; • User Acceptance Testing (UAT) Test Cases.	
Description	This activity is related to the generation of test cases that will be used during the Testing Phase. Both SIT and UAT Test cases are comprehended here, the Tester being responsible for the elaboration of the SIT Test Cases while the Business Analyst, with help from business users (Project Stakeholder), is responsible for the elaboration of the UAT Test Cases.	

Table 22: Details of “Create Test Cases” Activity.

Activity: Perform Developer Integration Test		
Roles	Primary • Developer.	Secondary • Tester; • Project Stakeholder.
Inputs	Mandatory • Test Plan (TP); • Test Specification (TSP); • System Design Specification (SDS).	Optional • Project Architecture Document (PAD); • Infrastructure Document (IFR).
Outputs	• Developer Integration Test (DIT) Evidence.	
Description	This activity is related to the execution of the Developer Integration Test. The software build being tested is made of the integration of all components previously developed and unit tested.	

Table 23: Details of “Perform Developer Integration Test” Activity.

4.1.3.2. Our Building Phase Recommended Practices for Waterfall

- IT Infrastructure, non-production (and production if possible), is raised during this phase. Attention to all needs mapped in the infrastructure document is essential, as failure to meeting one of those needs may directly affect the future performance of the finalized software product, or even its functionality. The effectiveness of the system tests may also be adversely affected;
- Logging and monitoring functionalities must be implemented according to the strategy previously mapped in the PAD. This will allow an easier traceability of defects in non-production environment, since it will be possible to quickly identify from which application being integrated is the defect coming from. Also, when in production, traceability of incidents will also be benefited by the same approach, directly benefiting the system support team;
- Developer Integrated Testing (DIT) first includes only mocked integrations, but in a second moment, if possible, will be done with all integrations in the non-production environment, thus simulating as best as possible what will be found in production. This will greatly increase the stability of software build being handed over to the test team, thus helping in decreasing the possibility of critical defects, and also of rework and of extra costs.

4.1.3.3. Alignment of Our Building Phase Recommended Practices to Waterfall

- Completion of the development step with a much more stable code. Proper attention is provided to the implementation of the key project system distribution characteristics. Developer level testing also carries with it substantial more attention to its standards, which should reflect upon less defects being delivered up-front to the test phase.

4.1.3.4. Adapting Our Building Phase Recommended Practices to Scrum

- Sprint Zero includes the assembly of non-prod infrastructure required for developing the solution;
- Sprint Zero includes, besides regular requirement analysis, test analysis for test scenarios generation. Upcoming sprints will maintain the same approach. The benefits from this practice is that not only the analysis team

will be ahead of the development team, but also the test team will be one step ahead, which should result in better product quality.

4.1.3.5. Alignment of Our Planning Phase Adapted Practices to Scrum

- System analysis team is one step ahead of the rest of the team, thus making sure requirements are well understood before actual implementation. Same happens to test team, and now the project benefits from the "planned in advance" testing.

4.1.4. Testing Phase

In this phase one performs detailed testing from both the test team's and users team's perspectives. Defect management and handling happen during the entire phase. Performance testing, when applicable, is also carried out on this phase. An overview of the activity flow of Testing Phase can be seen in Figure 26.

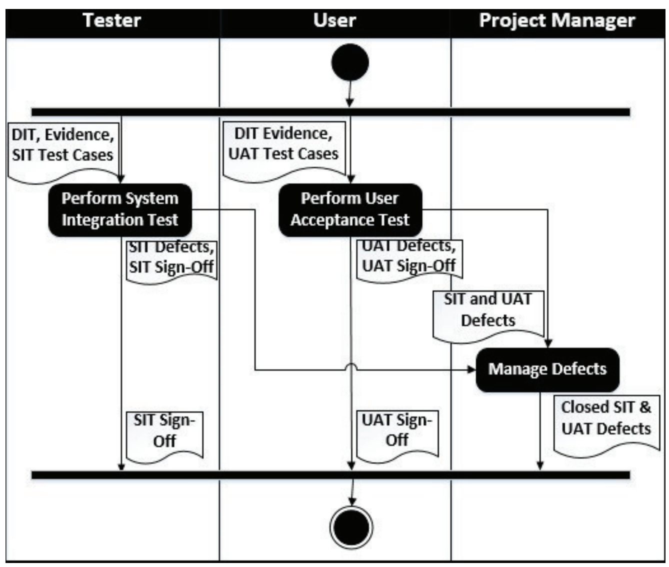

Figure 26: Testing phase of proposed Waterfall cycle.

4.1.4.1. Testing Phase Activities Breakdown and Description

Activity: Perform System Integration Test		
Roles	Primary • Tester.	Secondary • Project Stakeholder.
Inputs	Mandatory • System Integration Test (SIT) Test Cases.	Optional • None.

Outputs	• SIT defects; • SIT sign-off.	
Description	This activity is related to the execution of the SIT test cases. The previously defined test strategy will be followed, with key focus being destined to validate the system distribution aspect. The results will be analyzed, compiled, and feedback will be provided for the team via defects for correction of bugs and overall improvement of the system. When a satisfactory delivery is reached, the test team will provide its SIT Sign-Off, which attests the build is approved, from the test team's perspective, to be deployed to production environment.	

Table 24: Details of "Perform System Integration Test" Activity.

Activity: Perform User Acceptance Test		
Roles	Primary • Business Analyst.	Secondary • Project Stakeholder.
Inputs	Mandatory • User Acceptance Test (UAT) Test Cases.	Optional • SIT sign-off.
Outputs	• UAT defects; • UAT sign-off.	
Description	This activity is related to the execution of the UAT test cases. Usually the actual execution will be done by business users of the system, under the supervision of the Business Analyst. The results from UAT will be analyzed, compiled, and feedback will be provided for the team via defects for correction of bugs and overall improvement of the system. When a satisfactory delivery is reached, the Business Analyst will provide the UAT sign-off, which attests the build is approved, from the business' perspective, to be deployed to production environment.	

Table 25: Details of "Perform User Acceptance Test" Activity.

Activity: Manage Defects		
Roles	Primary • Project Manager.	Secondary • Developer; • Tester; • Business Analyst; • System Analyst; • Project Stakeholder.
Inputs	Mandatory • System Integration Test (SIT) Defects; • User Acceptance Test (UAT) Defects.	Optional • None.
Outputs	• Closed SIT defects; • Closed UAT defects.	
Description	This is related to managing defects backlog originating from SIT and UAT. The SIT and UAT testers are responsible for creating new defects, while the Project Manager usually is responsible for keeping track of the whole defects list, delegating each defect to the development team, following up on defect reject or acceptance and correction. The development team will assess each delegated defect, verifying if it is pertinent. If no, they will reject the defect, and if yes, will accept the defect, correcting it next.	

Table 26: Details of "Manage Defects" Activity.

4.1.4.2. Our Testing Phase Recommended Practices for Waterfall

- Testing should provide an important focus on the Non-Functional Requirements (NFRs), considering they have highly influenced the system distribution decisions and will highly influence the users experience with the final software product;
- Mocked data should be avoided at this stage. The use of data masses that are the closest possible to production is highly encouraged;
- Mocked integrations should also be avoided at this stage. It is very important to have all systems already integrated in the testing non-production environment, thus greatly improving the realism of the tests being performed, reducing by consequence the chances of instability issues when the system goes live;
- Test infrastructure and overall environments in use must be the closest possible to what will be found in production, thus greatly improving the realism of the tests being performed, reducing by consequence the chances of instability issues when the system goes live;
- Sign-offs should be received from who is performing the tests by the end of System Integrated Testing (SIT) and User Acceptance Testing (UAT). This generates commitment from the test and business users teams towards a thorough and effective execution of their tests.

4.1.4.3. Alignment of Our Testing Phase Recommended Practices to Waterfall

- An independent test team will validate the system that was built and delivered;
- There is a stabilization of system prior to handing it over to business users for the execution of UAT testing;
- Realistic testing will help in preventing many incidents in production.

4.1.4.4. Adapting Our Building Phase Recommended Practices to Scrum

- Production environment can be raised and continuously refined at this point, until it reaches its final desired configuration;

- The aforementioned waterfall test phase practices can be used equally in test under Scrum, without any adaptations.

4.1.4.5. Alignment of Our Planning Phase Adapted Practices to Scrum

- An independent test team will validate the system that was built and delivered;
- There is a stabilization of system prior to handing it over to business users for the execution of UAT testing;
- Realistic testing will help in preventing many incidents in production.

4.1.5. Releasing Phase

In this phase one provides the support team and users with training on the system being delivered. The system is also made available for use in production. Provision of a warranty period for the system, when all production incidents are still addressed by the project team, is encouraged. After compilation of Lessons Learned for the benefit of future projects, the Project Manager can perform the closure of the current Project. An overview of the activity flow of Release Phase can be seen in figure 27.

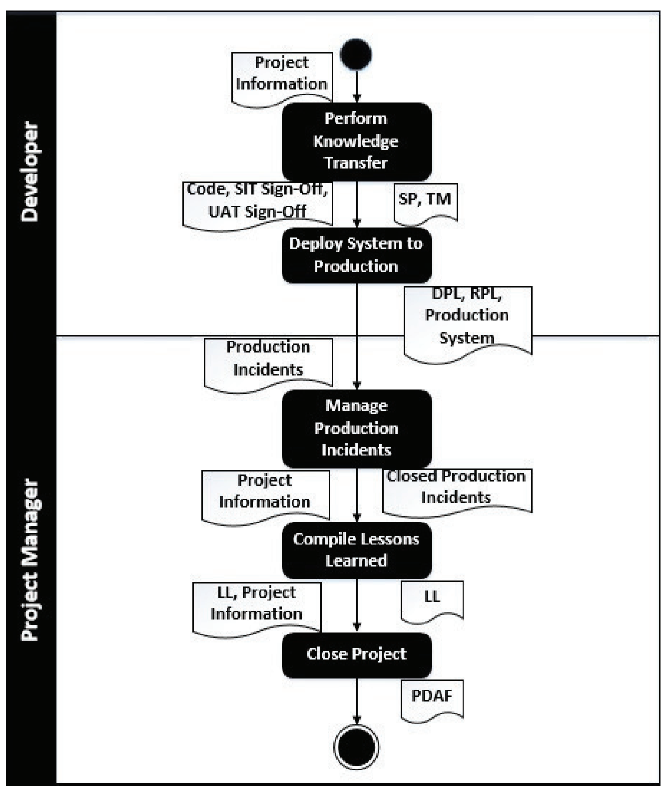

Figure 27: Releasing phase of proposed Waterfall cycle.

4.1.5.1. Releasing Phase Activities Breakdown and Description

Activity: Perform Knowledge Transfer		
Roles	Primary • Developer.	Secondary • System Analyst; • Business Analyst.
Inputs	Mandatory • Project Information: - Project artifacts; - Meeting notes; - Project decisions.	Optional • None.
Outputs	• System Profile (SP); • Training Materials (TM).	
Description	This activity is related to the creation and handover, from the development team to the support team, of the System Profile document, which is the primary document used for knowledge transfer between both teams. The activity also includes the creation and handover of training materials, from the development team to the business users and support teams, as well as the execution of the required training sessions between these teams. All the aforementioned artifacts will make extensive use of project information gathered from various sources during the entire project.	

Table 27: Details of "Perform Knowledge Transfer" Activity.

Activity: Deploy System to Production		
Roles	Primary • Developer.	Secondary • System Analyst; • Business Analyst; • Software Architect; • Infrastructure Analyst.
Inputs	Mandatory • Source Code; • SIT Sign-Off; • UAT Sign-Off.	Optional • None.
Outputs	• Deployment Plan (DPL); • Rollback Plan (RPL); • Production System.	
Description	This activity is related to the actual deployment of the system to production. A Deployment Plan is put in place for outlining the deployment tasks required to release the system. If the deployment is successful, the system will be fully operational, thus being able to be at the service of its business users. A Rollback Plan is also created as part of this activity, being put in place with all tasks required for, in case something goes wrong, removing the recently deployed system from production, as well as for restoring the environment to its previous state found prior to the unsuccessful deployment.	

Table 28: Details of "Deploy System to Production" Activity.

Activity: Manage Production Incidents		
Roles	Primary	Secondary

	• Project Manager.	• Business analyst; • Business analyst; • Developer; • Tester; • Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Production Incidents.	Optional • None.
Outputs	• Closed Production Incidents.	
Description	This activity is related to the management of incidents found in production for the recently deployed system. While the business users (Project Stakeholders) are the ones raising the new defects and including them in the defects list, it is up to the Project Manager to track these lists, assigning each defect to the development team, and following these defects up to their closure. The test team may be involved, depending on if tests (regression, performance, etc.) will be required due to defect / fix complexities.	

Table 29: Details of "Manage Production Incidents" Activity.

Activity: Compile Lessons Learned		
Roles	Primary • Project Manager.	Secondary • Business analyst; • Business analyst; • Developer; • Tester; • Software Architect; • Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Project Information: - Project artifacts; - Meeting notes; - Project decisions.	Optional • None.
Outputs	• Lessons Learned (LL).	
Description	This activity is related to the compilation of Lessons Learned by the entire Project team and stakeholders. It relies on project information coming from various sources, such as project artifacts, meeting notes, project decisions, etc.	

Table 30: Details of "Compile Lessons Learned" Activity.

Activity: Close Project		
Roles	Primary • Project Manager.	Secondary • Project Stakeholder.
Inputs	Mandatory • Lessons Learned (LL); • Project Information: - Project artifacts; - Meeting notes; - Project decisions.	Optional • None.

Outputs	• Project Delivery Acceptance Form (PDAF).
Description	This Activity is related to the compilation of Lessons Learned by the entire Project team and stakeholders. It relies on project information coming from various sources, such as project artifacts, meeting notes, project decisions, etc. It also requires Lessons Learned to have been previously raised, as the Project Manager will make them available for the next projects through the Project Management Office (PMO), a Project Stakeholder. As a final artifact for the project, the PDAF is filled in and signed by the customer, another Project Stakeholder. This marks the end of the project.

Table 31: Details of "Close Project" Activity.

4.1.5.2. Our Release Phase Recommended Practices for Waterfall

- A deployment and rollback plans should be available for tracking of all the deployment tasks and their impacts to each integration. This will help in the traceability of deployment issues and different responsibilities during deployment execution;
- A post-deployment plan should be available in order to test in production the systems being deployed and integrated. This will help in validating if at least their core functionalities remain unaffected and available for use;
- A "System Profile" document is created for describing, in business terminology, the system being implemented, its purpose, integration points, data flowing in and out of it, etc. This will be the base of the Knowledge Transfer (KT) for the support team and users alike;
- The Lessons Learned (LL) document captures learned items that will be inputs to upcoming projects. A system distribution option exists for documenting lessons related to DS.

4.1.5.3. Alignment of Our Release Phase Recommended Practices to Waterfall

- Close of project occurs only when the system is fully transitioned to production, stable and formally accepted by its business users;
- Completion of system transition from the project team to the support team is a requirement for closing out the project. This is an important consideration, as the product cycle goes on, even after the project was closed;
- Knowledge is captured regarding the experiences lived within the project, which will be of help for future projects.

4.1.5.4. Adapting Our Release Phase Recommended Practices to Scrum

- If there is not enough time in last project sprint, then a "Sprint-F" (of Final) should be made available for carrying out KT and the remaining documentation, including System Profile Document;
- The capture of Lessons Learned as well as the closure of the project closure are both performed as part of final Sprint Review and Sprint Retrospective, using Sprint-F for that as well, if needed.

4.1.5.5. Alignment of Our Release Phase Adapted Practices to Scrum

- Documentation is generated only until it generates value for the users / customers;
- Project closure happens when expected product value has been delivered;
- System maintenance is considered, as for that purpose there is the foment of KT and stabilization of the delivered system;
- Continuous improvement of projects is fomented through the raise of lessons learned, that will be archived and made available for future projects.

4.1.6. Monitoring and Controlling Phase

Through the Monitoring and Controlling (M&C) phase, which happens parallel to the Project, from its beginning to end, project management oversight is provided for all other phases. Change control is implemented, and change impacts to scope, cost, schedule and others areas are monitored, action being taken when needed. Project status is reported for providing the team and all project stakeholders with the same levels of visibility about project details and progress. An overview of the activity flow of Monitoring and Controlling Phase can be seen in figure 28.

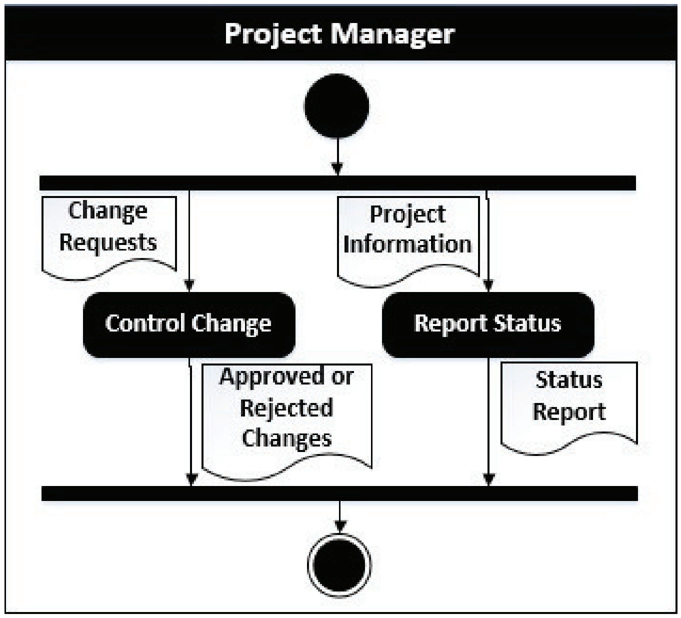

Figure 28: Monitoring and Controlling phase of proposed Waterfall cycle.

4.1.6.1. M&C Phase Activities Breakdown and Description

Activity: Control Change		
Roles	Primary • Project Manager.	Secondary • Business analyst; • Business analyst; • Developer; • Tester; • Software Architect; • Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Change Requests.	Optional • None.
Outputs	• Approved Change Requests; • Rejected Change Requests.	
Description	This Activity is related to the management of Change Requests by the Project Manager, more specifically, their reception, analysis, prioritization, scheduling, and approval or rejection. While any Project Stakeholder may raise new Change Requests, the Project Team will be mainly responsible for assessing each Change Request and its impact to the Project, implementing and validating the change after it has been implemented.	

Table 32: Details of "Control Change" Activity.

Activity: Report Status		
Roles	Primary • Project Manager.	Secondary • Business analyst; • Business analyst; • Developer; • Tester; • Software Architect;

		• Infrastructure Analyst; • Project Stakeholder.
Inputs	Mandatory • Project Information.	Optional • None.
Outputs	• Status Reports.	
Description	This Activity is related to the reporting of project status by the project manager to the team and to the stakeholders. It relies on project information coming from various sources, such as project artifacts, meeting notes, project decisions, etc.	

Table 33: Details of "Report Status" Activity.

4.1.6.2. Our M&C Phase Recommended Practices for Waterfall

- Status reports must address the system distribution project aspect, clearly describing what is the status on infrastructure needs, as well as on each integration. Items such as difficulties being faced by the project team, implementation steps completed, opportunities and risks, levels of engagement for required teams, and other relevant information should be part of the report.

4.1.6.3. Alignment of Our Release Phase Recommended Practices to Waterfall

- Synchronization of all stakeholders' visibility on all key project aspects, including the system distribution one;
- Foment of the whole team's participation on all project issues and decisions.

4.1.6.4. Adapting Our Release Phase Recommended Practices to Scrum

- Daily Scrums, Sprint Plannings and Release Plannings may have part of their time dedicated for the review of the teams' accomplishments regarding infrastructure / integrations items.

4.1.6.5. Alignment of Our Release Phase Adapted Practices to Scrum

- "All" project aspects really become visible to everyone at all times, including the ones related to system distribution, which tended to be "suppressed" before.

5. VALIDATION AND LIMITATIONS

We define this research as an empirical, qualitative one. This definition reflected upon our research methodology, steps, results and final academic contributions, represented by the presented preliminary version of the DS SDLC and its associated practices for DS projects. The aforementioned research aspects all lean towards the Grounded Theory Method (GTM). [LEH10] points out that "GTM is specifically geared to discover social theory from empirical data sourced in a wide range of contexts and activities".

The empirical nature of this research is also reflected upon its evaluation, since its aforementioned contributions, after created and sufficiently stable, went through the process known as "member checking" for evaluation and improvement. [SHU07] states that "member checking is a traditional validation technique used in empirical work". It suggests that those who had initially contributed to the work review the consolidated findings for accuracy. It is also common to invite additional people to ensure that there is no bias.

We invited 5 participants from the 16 original IT professionals who had previously participated on our interview-based field study, mentioned in Chapter 3, to participate again in the evaluation session. They were chosen due to the authors' perception of their highly critical opinions as well as the importance of their previous contributions. We also invited 2 additional professionals that had no previous contact whatsoever with this research. They were selected based on their seniority as IT professionals, each one having more than 15 years of work experience in IT. Both were from Brazil.

- The feedback obtained during the member-checking sessions was encouraging. The participating professionals all agreed that many practical benefits should come from the practical implementation, within IT industry projects, of our proposed SDLC and its associated practices. The participants also had their own inputs, with further improvements, to the proposed SDLC and practices, which in turn led to the version presented and discussed in the scope of this work. A few of these inputs that came directly from our member-checking participants are listed next:It was suggested that given our proposal's lack of a PLC, we could include in our DS SDLC some managerial activities and deliverables, since there were many project management challenges involved, . In his opinion, it could become harder for PMs to absorb the new delivery culture we were trying to convey, if they did not come to have any DS tailored activities and deliverables of their own. This suggestion was incorporated, and it became a key differentiating feature of our current proposal;

- It was suggested that our SDLC should remain as simple as possible, with activities and practices that could be easily adopted by any team, regardless of their maturity. According to the participant who proposed this item, it would be better if our proposals could be implemented at any company, something that would not be possible if we continued to incorporate, as we did at that point, suggestions regarding some more advanced, state-of-the art concepts, such as continuous integration, Test-Driven-Development (TDD), etc. We agreed to this point of view, and as a consequence, simplified our model, which now is mainly based on improvements directed towards artifacts and communication among the team and project stakeholders. Those improvements were relatively simple to be adopted without restrictions;
- One of the participants highlighted the importance of embedding the right mindset into the team and stakeholders. According to her, without a cultural shift, nobody, especially from business, would see the benefit of taking into consideration the system distribution aspects of a project, thus considering it to be a burden, something solely IT related, with no business value whatsoever. We attempted to covey this right mindset through the reinforcement, already from the Vision Phase onwards, of the idea that the system distribution aspects are directly tied to the functional requirements and also the non-functional requirements, such as usability, performance, etc. We designed our activities and practices to foment an early-thinking approach, aiming to bring forward some parts of key activities such as those of requirements definitions. That should bring benefits in cascade, so that when we finally arrive, for example, at the solution design stage, we have much more stable requirements, reducing the chances of rework and project failure.

We did not include in this research, the practical validation of our SDLC and practices with real projects. Many are the reasons for this, including the fact that we would need a considerable time for this extra step. As we had to comply with a mandatory two-year Master's degree research completion deadline, we did not have such time up to this moment. We will, however, continue this work in an extension of our original research project, bounded by a Doctorate program, which in turn will guarantee we will have time for practical experiments in the IT industry. Until then, we are aware our study and the respective product of it, the proposed SDLC and practices, have a set of limitations, as described next:

- No practical experiments with real projects and/or companies conducted so far;
- Our SDLC currently does not drill down to task-step structure. This will be done in the next research iterations, as real projects are used for validation and maturity is gained regarding more fine-grained details;
- We did not oblige ourselves, at this moment, to propose a number of practices that could cover absolutely all activities of our SDLC. We instead focused on proposing practices that could directly relate to the pain points raised during the interviews. We understand however that most likely these are not the only pain points in existence, and that as soon as we apply our SDLC and practices into real projects, new pain points will show up, demanding the proposal of new practices, and even of more activities;
- Participants in our interview-based field study were obtained from limited geographical locations, these being the ones that allowed us to have easier access to industry companies (Brazil and United States);
- Participants of the member checking activity were obtained from a single geographical location (Brazil);
- The diversity of companies participating on the qualitative interviews was not ideal, with one single American international company providing nine out of 16 interviewees;

We focused on establishing a generic SDLC with specific practices, meaning that we focused on a product cycle, not on a management cycle. Despite this, our SDLC has some management generic activities and deliverables, which we believe are at the bare minimal required for any company to carry on with a project. As this research matures, these should be migrated into a more comprehensive, independent project management cycle that would be executed in parallel with the product cycle, thus covering all aspects of real projects.

6. CONCLUSIONS

Information Technology has changed the way we see and interact with our world. Almost every aspect of human life is today, directly or indirectly, influenced by computers. This revolution started decades ago, and continues even now, as computational paradigms increasingly more advanced and sophisticated greatly increase computers performance and capabilities, allowing them to be at the service of all sorts of other industries and purposes. One of these many different computational paradigms is the one presented by Distributed Systems solutions.

Distributed Systems have flexibility and reliability in their nature, both being highly desired characteristics to any solution in a world that is increasingly connected and that is decreasingly tolerant to failure. It is no surprise that Distributed Systems are then the paradigm of choice in the majority of today's software development projects. In this work we demonstrated, that as initially hypothesized by the researchers, Distributed Systems do bring considerable amounts of challenges to software development projects.

Some of these challenges are from a technical nature, some more evident being related to testing, the different integrations required, and IT infrastructure. Other challenges are from a managerial nature, such as the management of the many risks, of the different required technical knowledge and of the team, just to mention a few. Despite their popularity and the considerable amount of challenges they do bring to software development projects, little practical knowledge from the IT industry as well as little academic research exist though on the intersection of Distributed Systems and Project Management.

The reason for this imbalance is not known, and was not a target for being answered by this current research. We were able to show, however, that even experienced professionals, both from managerial and technical backgrounds, are unaware of the fact that they participate on Distributed System projects in a day-to-day basis. After they became aware of this fact though, they recognized the importance of better understanding those challenges and taking systematic and organized actions in order to mitigate or even eliminate most of these issues.

After working in partnership with these professionals during the course of our research, we came to the conclusion that the best thing we could immediately propose to help in mitigating these challenges and issues was a Software Development Life Cycle (SDLC), tailored for software development projects involving Distributed Systems. We believe that our final version of the SDLC, here presented in this dissertation, is in line with our interviewees' wishes for an effective countermeasure for the identified challenges, addressing them by mainly broadening project teams´ awareness about the importance of properly handling the System Distribution technical aspect on their projects, enabling a new mindset to take place as something that must integrate the culture of software delivery.

The SDLC will also provide elements to facilitate communication with users and customers, allowing them to realize how complex a software truly is, not only from a regular requirements perspective, but from technical and infrastructure perspectives as well. Those benefits should mostly come from the adaptations that make our SDLC more DS friendly, as well as from the different tailored practices we proposed, and that have been designed to be used in association with the SDLC. They are a mean to further guide the project team's actions, as well as to potentialize the benefits we are trying to achieve.

As future work we plan to apply our SDLC and practices into real projects, thus practically validating them. We are aware that many improvements are yet to come due to new data that will come from these validations. We also intend to increment our model, having it to include a Project Life Cycle (PLC) that will be used for fully segregating Project Management activities from product delivery activities. This segregation will allow us expand and to better cover the Project Management aspect of DS Projects, which by the way, was the original intent of this research.

REFERENCES

[ABD13] Abdullah, E; Abdelsatir, E. "Extreme programming applied in a large-scale distributed system". In: International Conference on Computing, Electrical and Electronic Engineering (ICCEEE), 2013, pp. 442-446.

[ALI08] Ali Babar, M; Kitchenham, B; Jeffery, R. "Comparing distributed and face-to-face meetings for software architecture evaluation: A controlled experiment". *Empirical Software Engineering*, vol. 13, Feb 2008, pp. 39-62.

[ANA10] Anantatmula, V. "Project Manager Leadership Role in Improving Project Performance". *Engineering Management Journal*, vol. 22, n.1, Mar 2010, pp. 13-22.

[AUE94] Aue, A; Breu, M. "Distributed information systems: an advanced methodology". *IEEE transactions on software engineering*, vol. 20, Aug 1994, pp. 594 – 605.

[BOD07] Boden, A; Nett, B; Wulf, V. Coordination Practices in Distributed Software Development of Small Enterprises. In: International Conference on Global Software Engineering, 2007, pp. 235-246.

[BOO05] Booch, G., Rumbaugh, J., Jacobson, Ivan "The Unified Modeling Language User Guide". Boston: Addison-Wesley, 2nd edition, 2005, 496p.

[BRA98] Braxton, T. "The challenge of regulatory compliance testing of a large distributed system". In: IEEE International Conference on Electromagnetic Compatibility, 1998, pp. 711-716.

[CAT09] Cataldo, M; Nambiar, S. "Quality in Global Software Development Projects: A Closer Look at the Role of Distribution". In: International Conference on Global Software Engineering, 2009, pp. 163-172.

[CHR01] Christof, E; Casimiro, H; Roland, S; Harald, K. "Improving validation activities in a global software development". In: International Conference on Software Engineering (ICSGE), 2001, pp. 545-554.

[COL14] Colomo-Palacios, R; Casado-Lumbreras, C; Soto-Acosta, P; García-Peñalvo, F; Tovar, E. "Project managers in global software development teams: A study of the effects on productivity and performance". *Software Quality,* vol. 22, Mar 2014, pp. 3-19.

[COU12] Coulouris, G., Dollimore, J., Kindberg, T., Blair, G. "Distributed Systems Concepts and Design". Boston:Addison-Wesley, 5th edition, 2012, 1008p.

[DAM02] Damian, D; Zowghi, D. "The impact of stakeholders geographical distribution on managing requirementsin a multi-site organization". In: IEEE Joint International Conference on Requirements Engineering, 2002, pp. 319-328.

[DES10] Deshpande, S; Richardson, I; Casey, V; Beecham, S. "Culture in Global Software Development - A Weakness or Strength?". In: International Conference on Global Software Engineering (ICGSE), 2010, pp. 67-76.

[EBE08] Ebert, C.; Murthy, B; Jha, N. Managing Risks in Global Software Engineering: Principles and Practices. In: International Conference on Global Software Engineering (ICGSE), 2008, pp. 131-140.

[ERF06] Erfurth, I. "Customer-oriented development of complex distributed systems". In: Symposium and Workshop on Engineering on Computer Based Systems, 2006, pp. 27-30.

[ERI06] Erickson, J; Evaristo, R. "Risk Factors in Distributed Projects". In: Hawaii International Conference on System Sciences, 2006, 216p.

[FEL84] Felix, R; Harrison, W. "Project management considerations for distributed processing Applications". *MIS Quarterly*, vol. 8, Sep 1984, pp. 161-170.

[HER01] Herbsleb, J. D., Moitra, D. "Global Software Development", *IEEE Software*, vol. 16, n.2, 2001, pp. 16-20.

[HER03] Herbsleb, J; Mockus, A. An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on Software Engineering*, vol. 29, Jun 2003, pp. 481-494.

[HOL06] Holmstrom, H; Conchuir, E; Agerfalk, P; Fitzgerald, B. In: Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance. In: International Conference on Global Software Engineering (ICGSE), 2006, pp. 3-11.

[JIM09] Jiménez, M; Piattin,i M; Vizcaíno, A. "Challenges and improvements in distributed software development: a systematic review". *Advances in Software Engineering*, vol. 2009, Jan 2009.

[JIN93] Jingde, C. "Dependence-Based Complexity Metwircs for Distributed Programs". In: International Conference on Parallel Processing, 1993, pp. 265-268.

[KIT07] Kitchenham, B; Charters, S. "Guidelines for performing Systematic Literature Reviews in Software Engineering". In: Keele University and Durham University Joint Report, 2007, 9p.

[KHA11] Khan S, K; Niazi, M; Ahmad, R. **"**Barriers in the selection of offshore software development outsourcing vendors: An exploratory study using a systematic literature review". *Information and Software Technology*, vol. 53, Jul 2011, pp. 693-706.

[KOT08] Kotlarsky, J; Fenema, P; Willcocks, L. "Developing a knowledge-based perspective on coordination: The case of global software projects". *Information and Management*. vol. 45, Mar 2008, pp. 96-108.

[KOZ11] Kozak, M. "The History of Project Management". Ontario:Multi-Media Publications Inc, 2011, 648p.

[KWA05] Kwak, Y; Carayannis, E; Anbari, F. "A Brief History of Project Management. The Story of Managing Projects: An Interdisciplinary Approach". Westport:Praegers Publishers, 2005, 373p.

[LEE06] Lee K., Probir; B, Kai H.; Kuldeep, Lim; Hillegersberg, J, Kee Wei, K. "Aligning IT components to achieve agility in globally distributed system development". *Communications of the ACM*, vol. 49, Oct 2006, pp. 48-54.

[LEH10] Lehmann, H. "Grounded Theory and Information Systems Are We Missing the Point". In: 43rd Hawaii International Conference on System Sciences, 2010.

[MAN09] Manuja, M; Kalra, R. "Knowledge Management in Distributed Data Source Environments". In: IEEE International Advance Computing Conference, 2009, pp. 1485-1491.

[MAT13] Mathrani, A; Mathrani, S. "Test strategies in distributed software development environments". *Computers in Industry*, vol. 64, Jan 2013, pp. 1-9.

[MIS11] Mishra, D; Mishra, A. "A review of non-technical issues in global software development". *International Journal of Computer Applications in Technology*, vol. 40, Mar 2011, pp. 216-224.

[MHT12] Matusse, E; Huzita, E; Tait, T. "Metrics and indicators to assist in the distribution of process steps for distributed software development: A systematic review". In: Conferencia Latinoamericana E Informatica (CLEI), 2012, pp. 1-10.

[MOC01] Mockus, A; Herbsleb, J. "Challenges of global software development". In: Software Metrics Symposium, 2001, pp. 182-184.

[NIA13] Niazi, M; Mahmood, S; Alshayeb, M; Riaz, R. "Challenges of project management in Global Software Development: Initial results". In: Science and Information Conference (SAI), 2013, pp. 202-206.

[NID05] Nidiffer, K., Dolan, D. "Evolving Distributed Project Management". IEEE Software, vol. 22, n.5, Sep-Oct 2005, pp. 63-72.

[NOL10] Noll, J; Beecham, S; Richardson, I. "Global software development and collaboration: barriers and solutions". *ACM Inroads*, vol. 1, Sep 2010, pp. 66-78.

[NUR11] Nurdiani, I; Jabangwe, R; Smite, D; Damian, D. "Risk Identification and Risk Mitigation Instruments for Global Software Development: Systematic Review and Survey Results". In: International Conference on Global Software Engineering, 2011, pp. 36-41.

[PET08] Petticrew, Roberts H. "Systematic Reviews in the Social Sciences: A Practical Guide". Indianapolis:Wiley-Blackwell, 1st edition, 2008, 336p.

[PER10] Persson, J; Mathiassen, L. "A Process for Managing Risks in Distributed Teams". *IEEE Software*, vol. 27, Jan-Feb 2010, pp. 20-29.

[PFM08] Petersen, K; Feldt, R; Mujtaba, S; Mattsson, M. "Systematic mapping studies in software engineering". In: International Conference on Evaluation and Assessment in Software Engineering, 2008, pp. 1-10.

[PMI13] PMI Inc. "A Guide to the Project Management Body of Knowledge: PMBoK Guide". Pennsylvania: Project Management Institute, 5th edition, 2013, 589p.

[PMI15a] PMI Inc. "What is Project Management?". Captured in: https://www.pmi.org/gac/students.aspx, October 2015.

[PMI15b] PMI Inc. "What is Project Management?". Captured in: http://www.pmi.org/About-Us/About-Us-What-is-Project-Management.aspx, October 2015.

[PRE14] Pressman, R, Maxim, B. "Software Engineering A Practitioner's Approach". New York: McGraw-Hill, 8th edition, 2014, 976p.

[RAL08] Ralyte, J; Lamielle, X; Arni-Bloch, N; Leonard, M. "A framework for supporting management in distributed information systems development". In: International Conference on Research Challenges in Information Science, 2008, pp. 381-392.

[RAM07] Ramasubbu, N., Balan R. "Globally distributed software development project performance: an empirical analysis". In: 6th joint meeting of the European software engineering conference and the ACM SIGSOFT Symposium on The foundations of software engineering, 2007, pp. 125-134.

[SIL10] Silva F, Costa C; França, A; Prikladinicki, R. "Challenges and solutions in distributed software development project management: A systematic literature review". In: International Conference on Global Software Engineering (ICGSE), 2010, pp. 87-96.

[SHU07] Shull, F., Singer, J., Sjoberg, D. "Guide to Advanced Empirical Software Engineering". New Jersey: Springer, 2008 edition, 2007, 388p.

[SUT15] Sutherland, J., Schwaber K. "The Definitive Guide to Scrum: The Rules of the Game". Captured in: http://www.Scrumguides.org/docs/Scrumguide/v1/Scrum-guide-us.pdf, October 2015.

[SNY87] Snyder, J; Kline, S. "Modern Project Management: How Did We Get Here - Where Do We Go?". *Project Management Journal*, vol. 18, Mar 1987, pp. 28-29.

[SOL10] Solís, C; Ali, N. "Distributed requirements elicitation using a spatial hypertext wiki". In: International Conference on Global Software Engineering (ICGSE), 2010, pp. 237-246.

[STE87] Stevens, R. "Design of a large distributed information system". In: Conference on Communication and Data Communication, 1987, pp. 123–135.

[TAY04] Taylor, J. "Managing Information Technology Projects: Applying Project Management Strategies to Software, Hardware and Integration Initiatives". New York: AMACOM, 2004, 304p.

[WEL09] Wells, K; Litell, J. "Study quality assessment in systematic reviews of research on intervention effects". *Research on Social Work Practice*, vol. 19, Jan 2009, pp. 52-62.

[YAN04] Yang, X; Xu, B; He, Z; Maddineni, S. "Extreme Programming in global software development". In: Canadian Conference on Electrical and Computer Engineering, 2004, pp. 1845-1848.

[VER14] Verner, J; Breretona, O; Kitchenham, B; Turnera, M; Niazia, M. "Risks and risk mitigation in global software development: A tertiary study". In: International Conference on Global Software Engineering, 2014, pp. 54-78.

[VER15] Version One. "The 9th Annual State of Agile Report". Captured in: http://info.versionone.com/state-of-agile-development-survey-ninth.html. October 2015.

APPENDIX A – INTERVIEW-BASED FIELD STUDY

INTERVIEW SCRIPT

Preliminary Interviewee´s Background Screening

Interviewee Name:	
Current Employer:	
I1. Education level:	
I2. Year of Birth:	
I3. Gender:	
I4. Country of Residence:	
I5. Years of Working Experience:	
I6. Years of Working Experience in SE (Technical):	
I7. Years of Working Experience in SE (Managerial):	
I8. Years of Working Experience for Current Employer:	
I9. Years of Working Experience in Current Position:	
I10. Current Position Description:	

INTERVIEW PURPOSE

To obtain clarification on whether DS is perceived as an impacting success factor for software development projects, and if considered as such, then which PM practices are the most impacted by it.

INTERVIEW DOMAIN

Distributed applications (distributed software running on top of distributed systems).

INTERVIEW SETUP

Explain to the interviewees that this interview is part of the process required for the interviewer´s Master´s Degree Course completion, and that it is an intersection of two different research areas: Software Engineering and Parallel and Distributed Programming.

Explain to the interviewees that they were selected based on the interviewer´s personal perception of their seniority as professionals, as well as their involvement with both research areas in their day-to-day work.

Explain to the interviewees that this will be a semi-structured, open-ended interview, meaning that are no predefined answers, and they may freely answer the questions. Explain also that the whole session will be recorded, and although their names won´t be disclosed, the interviews records will be transcripted and become part of the interviewer´s final dissertation.

RESEARCH QUESTIONS

Block 1: Contextual info about the interviewee´s background.

Purpose: To find out what level of maturity do PMs and technical teams have in order to be able to perceive their participation on Distributed Software projects.

RQ1: Tell me how you see in practice the Project Manager´s day-to-day involvement on technical aspects of software development projects, such as software architecture, system analysis, system modelling, software configuration, etc.

RQ2: Now, tell me how you think the ideal Project Manager´s day-to-day involvement on the same technical aspects of software development projects should be.

RQ3: Now let´s focus on one specific technical aspect. Are you familiar with distributed applications? How would you define them? *<After response, provide standard explanation for conceptual leveling – See backup.>*

RQ4: Now that we are in sync about the definition for Distributed Systems as well as for Distributed Applications, tell me what you would say is the percentage of all the software development projects you worked on in the last five years that involved distributed applications.

RQ5: Tell me how you see the role a clear, well-defined architecture plays in facilitating the integration of PMs and team on technical questions, including DS related ones. Please note that by well-defined architecture I mean things like layer-organized, thoroughly-tested, well-documented, pattern-following architectures.

RQ6: Tell me if you see current state-of-the-art technology (e.g., present day´s development frameworks, integrated development and testing tools suites) facilitating the integration of PMs and team on technical questions, including DS related ones.

RQ6.1: How do you come to your conclusion? Please try to trace a parallel to 5 or 10 years ago, when such technology could be more immature or could even not be available at all.

Block 2: DS Challenges to SE / PM.

Purpose: To find out the rate of project delivery failure or success that can be linked to DS.

RQ7: What do you think are the technical challenges (requirements elicitation and specification, implementation, testing and deploying) brought by projects involving DS?

RQ8: What do you think are the project management (planning, risks, status, procurement, change) challenges brought by projects involving DS?

RQ9: From all the projects involving DS you worked on and delivered in the last 5 years, what would you say is the percentage of project failure (scope, time, cost overruns or incompatibility, etc.)?

RQ9.1: On your opinion, why did those projects fail? Would you say there was any reason that could be linked to DS?

Block 3: Countermeasures for DS Challenges.

Purpose: To gather different insights on possible ways to handle DS projects in an effective way

RQ10: If you had the chance to go back in time and fix what went wrong, what actions would you take in order to bring success to those projects involving DS that have failed, according to you? Think of the highest priorities.

RQ11: Suppose you have a magic wand. If you could use it, what would you like to instantly create in order to help you with this kind of project going forward? Would it be documentation, frameworks, methods, etc.? Please explain how you think this would help you.

APPENDIX B – MEMBER-CHECKING INTERVIEW SCRIPT

Preliminary Interviewee´s Background Screening

Interviewee Name:	
Current Employer:	
I1. Education level:	
I2. Year of Birth:	
I3. Gender:	
I4. Country of Residence:	
I5. Years of Working Experience:	
I6. Years of Working Experience in SE (Technical):	
I7. Years of Working Experience in SE (Managerial):	
I8. Years of Working Experience for Current Employer:	
I9. Years of Working Experience in Current Position:	
I10. Current Position Description:	

MEMBER-CHECKING PROCESS APPLICATION PURPOSE

To validate how feasible it is the application of our DS SDLC to the software development industry. More specifically, how its proposed specialized actions and practices would be effective in dealing with common pitfalls of projects involving DS, but at the same time, how generic is our SDLC proposal so that its essence can be adapted to different versions of SDLCs being ran in different companies around the world.

PROCESS SETUP

Explain to the interviewees that this interview is part of the process required for the interviewer´s Master´s Degree Course completion, and that it is an intersection of two different research areas: Software Engineering and Parallel and Distributed Programming.

Explain to the interviewees that they were selected based on the interviewer´s personal perception of their seniority as professionals, their involvement with both research areas in their day-to-day work and also their important contributions to the previous qualitative interviews phase of this research.

Explain to the interviewees that there are no predefined answers, and they may freely answer the questions, and actually are encouraged to be highly critical regarding what is being proposed. Explain also that the whole session will be recorded, and although their names won´t be disclosed, the interviews records will be transcripted and become part of the interviewer´s final dissertation.

PROCESS INTRODUCTION

Explain what Distributed Systems are.

Present the main conclusions that came out from the qualitative interviews, including percentage of projects that deliver DS solutions, rate of project failure attributable to DS, main challenges and wanted countermeasures for treating these challenges. Explain that an SDLC come out as the most desired countermeasure by the interviewees.

Explain that as such, an SDLC has been designed to better cope with the challenges coming from software projects that are distributed regarding its infrastructure, but also regarding different softwares being integrated (this second one receiving a higher focus from the researchers.

Explain why the SDLC is designed mainly for a waterfall cycle, and why adaptations are proposed for Scrum:

- Waterfall represents tradition in software development and is still highly used;
- Waterfall is implemented in a very sequential way, which makes it easier for the researchers to demonstrate their proposal, as its tasks, deliverables and practices are implemented step by step, covering all major software engineering tasks (Product requirements, feasibility analysis, product scope and systems architecture, Preliminary design, design approval, detailed design, Construction of the system, Unit, system and integration testing and Delivering the system);
- Scrum on the other hand is the novelty, and is already the most used agile framework. Its trend is to become more and more popular, which only increases its relevance for today's software industry;
- Due to Scrum's high visibility, proposing such practices just for the waterfall model would make this research to feel dull, outdated and incomplete, not in sync with state-of-the-art.

PROCESS REVIEW

Block 1: Presentation of VISION phase, with its tasks, deliverables, practices and adaptations for Scrum.

- Explain each task and related deliverables of the waterfall vision phase;
- Explain each proposed practice, the pain points they are supposed to be addressing and show the demonstration examples available;
- Present the adaption of these tasks, deliverables and practices to the Scrum cycle;
- After explanation, proceed to the Member-Checking Questions.

MCQ1: Do you perceive the presented tasks, deliverables and practices as value-generators in day-to-day projects? Why?

MCQ2: What would you do differently from what has been proposed? Would you add, change or exclude anything?

MCQ3: Do you see the proposed adaptions of the waterfall practices as being feasible to be implemented within Scrum projects? Why?

MCQ4: Do you see any risk of projects loosing their agility if these practices are implemented?

Block 2: Same questions, Planning Phase;

Block 3: Same questions, Building Phase;

Block 4: Same questions, Testing Phase;

Block 5: Same questions, Releasing Phase;

Block 6: Same questions, Monitoring and Controlling Phase.