Towards an SDLC for Projects Involving Distributed Systems

Rodrigo Augusto dos Santos, Avelino F. Zorzo and Sabrina Marczak
Faculdade de Informática, Pontifícia Universidade Católica do Rio Grande do Sul, Ipiranga Avenue, Porto Alegre, Brazil

Keywords: Distributed Systems, Distributed Teams, Project Management, Life Cycle, SDLC, PLC.

Abstract: Since the 1970's, Distributed Systems have been turning into a more viable and reliable option for the implementation of information systems. This evolution continued ever since, and now they are applicable to a variety of purposes, such as online games, 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. Thus, it is relevant to study the impacts that Distributed Systems bring to Project Management. In this paper we discuss those impacts and challenges, as well as propose a Software Development Lifecycle and some associated practices that are to be used for software development projects involving Distributed Systems. Such practices are optimized for implementation under a Waterfall model, but are also adaptable for use with well known agile framework Scrum. The preliminary validation with industry professionals suggests that our proposals do support more appropriate management and execution of projects involving Distributed Systems solutions.

1 INTRODUCTION

A project is defined by PMI (2015) as being "a temporary endeavour in that it has a defined beginning and end in time, and therefore defined scope and resources". According to PMBoK (2013), "Project Management is the application of knowledge, skills, tools and techniques to project activities to meet the project requirements".

By 1970, the wide adoption of Distributed Systems (DS) became a fact, and Information Technology (IT) Project Managers around the world were forced to deal with it. DS, according to Couloris et al (2012), "are the ones in which hardware or software components, located at networked computers, communicate and coordinate their actions only by passing messages".

Couloris et al (2012) also provides some examples that fit this definition, such as web search, multiplayer online games, and financial trading systems, thus stating that DS includes "many of the most significant technological developments of recent years", "ranging from a small intranet to the Internet". This obviously turns the intersection between PM and DS into a relevant research area.

Our hypothesis though is that system distribution in a project may be regularly "abstracted" by IT project teams, with decisions regarding it becoming delegated to development teams only. The rest of the project team would focus on supposedly "attention-worthy, value-driven requirements", such as screens, reports, and other "tangible" features, thus, greatly increasing the risk of project failure.

This abstraction culture would also reflect upon academia, with small attention from researches on the intersection of DS and PM. In order to shed some light into our hypothesis, we performed a Systematic Mapping Study (SMS) (Section 2.1), seeking to understand how the intersection between DS and PM has been studied in academia. We also performed an interview-based field study (Section 2.2) to understand industry's perception about the topic.

The results from the SMS and field-based study led us to propose a Software Devolopment Life Cycle (SDLC) and some practices associated with it, both tailored for Software Develpment Projects involving DS (Section 3). These proposals were preliminarily validated through the process of member checking (Section 3.2). The limitations and future work are described in Section 3.2 and Section 4, respectively.

2 RESEARCH BACKGROUND

In this section we present the methodologies used in our Systematic Mapping Study (SMS) and interview-based field study, as well as their results.

Santos, R., Zorzo, A. and Marczak, S.
Towards an SDLC for Projects Involving Distributed Systems.
In *Proceedings of the 18th International Conference on Enterprise Information Systems (ICEIS 2016) - Volume 1*, pages 158-165
ISBN: 978-989-758-187-8

2.1 Systematic Mapping Study

As a comparison measure for the volume of research on DS PM, we have used PM involving Distributed Teams (DT). Since DT has been adopted by many organizations distributing their software development projects worldwide, seeking cost and quality advantages (Herbsleb,2001), DT PM became a popular research topic.

Although DS and DT are two distinct subjects, with no direct relation between them, both topics are present in a great number of today's IT projects, having the research on each of them the same characteristic of being able to intersect with PM. The SMS, thus, was performed for confirming the level of attention provided to DS PM when compared to the volume of studies focusing on DT PM.

The number of papers selected as a result of systematic search was 37 out of 127. Out of these, 28 focused on PM intersection with DT, 8 focused on PM intersection with DS and only 1 focused on PM intersecting both DT and DS at the same time. These results demonstrate an imbalance in the academic interest towards both DT and DS. Another imbalance indicator is that out of the 8 DS PM papers, 50% of them were published before year 2000.

2.2 Interview-based Field Study

Due to the SMS results, we designed an interview-based field study with IT industry professionals. Our intent was to better understand the practical relation of the DS and PM areas, what are today's challenges of projects involving DS, as well as what could be used as possible countermeasures for such challenges.

Semi-structured interviews were conducted with 16 professionals from Brazil (14) and United States (2). The selection criteria was based on their IT industry experience (at least 10 years) and ability to be critical (as perceived by the researchers).

Their role distribution was: 9 project managers, 2 development leaders, 2 test leaders, 1 business analyst, 1 architect, and 1 IT Manager. In average, they 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 exployer. Next, we briefly present the findings of our field-based study.

2.2.1 Technical Project Managers

The perception of 68.75% of our interviewees is that project managers usually are not involved with technical aspects in the projects they manage. Still, 62.5% considered beneficial, project delivery wise, to have project managers with technical knowledge.

2.2.2 Awareness of System Distribution

Regarding awareness of what DS is, 62.5% of the interviewees were not even familiar with the concept. After Section 1 definition was provided, all interviewees confirmed they now understood the concept, having 84% of them claimed to have participated in DS projects in the past 5 years.

Therefore, the high volume of today's software development projects involving DS does make it difficult even for experienced professionals to realize how frequently they are inserted in such context. For them, these are "just regular projects", where DS is a almost a mandatory solution aspect. This constitutes evidence of an "abstraction trend" of the DS feature.

2.2.3 The Challenges from DS Projects

The discussed challenges of DS projects were either technical or managerial aspects of software development. Each interviewee was allowed to provide as many challenges as they wanted, including ones for a same item. The challenges were then grouped into categories.

The list of categorized main technical challenges and their individual occurences is as follows: Testing (14), IT infrastructure (17), integrations (6), fidelity of non-production to production environments (4), system security (3), system architecture (6), requirements (7), deployments (7), existence of too many implementation options (3) and others (8).

We also discussed managerial challenges related to DS projects. The list of categorized main managerial challenges and their individual occurences is as follows: obtain a skilled team (5), risk management (9), knowledge management (5), team management (4), communication (8), vendor management (5), project planning (6) and others (7).

After the interviews, the main definition of "system distribution" of our study was restricted to solutions that are: (i) distributed regarding their IT infrastructure, e.g. a software distributed between an application server and a database server; and (ii) distributed among different softwares, integrated with each other through interfaces or other mechanisms that allow exchanges, such as of data, tokens, etc.

2.2.4 Failed DS Projects

From the DS projects that the interviewees participated in the last 5 years, an average project failure of 38,44% was reported, having 81,25% of the

interviewees claimed, 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.

The interviewees then provided a set of countermeasure choices they would like to have for dealing with DS projects challenges. The most recurring one was a Software Development Life Cycle (SDLC) specialized in DS (9 occurrences). The other choices were a development framework (2), PM framework (1), diverse tools (3) and others (1).

According to Taylor (2004), "an SDLC is a subset of the project life cycle", "focused on accomplishing the product requirements". The main difference from Project Life Cycle (PLC) activities is that SDLC activities focus on technical aspects of project deliverables while PLC ones are more related to management and leadership (Taylor, 2004).

3 AN APPROACH FOR DS SDLC

Given the discussed results, we 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 Taylor (2004).

Because of this generic nature, our main contribution is thereby on the differentiated practices we are proposing, and that should be used in association with the organized structure of the SDLC. These practices are adaptations on well-known and disseminated items, such as a Project Architecture Document or a System Requirements Document for example, tailoring them for use within DS Projects.

Our Phase-Activity-Deliverable structure has to be viewed then as a non-prescriptionary guide. It can be used as is, but it also is easily mappable against different SDLC versions in use by IT companies around the world, which means they could keep using their own processes while simply adding our proposals to them, as they see fit.

3.1 Overall View of Our DS SDLC

The proposed SDLC is designed for an optimal implementation with Waterfall (Pressman, 2001). Adaptations are also proposed for use with Scrum, since one cannot ignore its growing use in today's industry, as demonstrated by VersionOne (2015).

We present our DS SDLC and its associated practices next, all in high-level detail due to space restrictions. We have represented the activitiy flows of each phase through activity diagrams, compliant with Unified Modeling Language (UML) notations. One customization to the notations was made, 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.

Our suggested SDLC practices and some examples related to them are textually described. For the software integration proposals (type of solution 'ii' as defined in Section 2.2.3), examples are based on software integrated data-wise (exchange of data through data interfaces, for example).

3.1.1 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. Visibility on DS Project challenges should exist, so that due countermeasures can be planned and implemented, as early as possible in the project. An overall view of the Vision Phase activities, with its inputs and outputs can be seen in Figure 1.

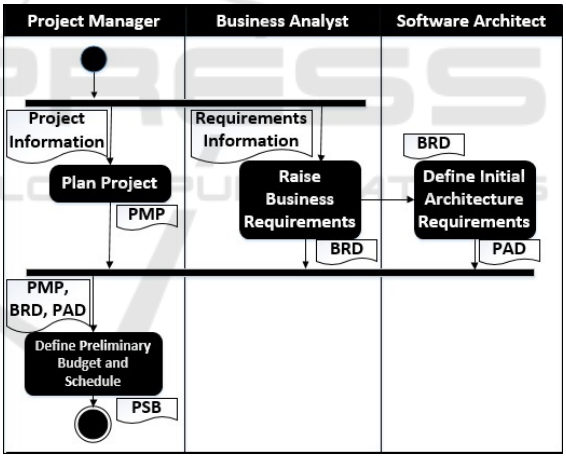

Figure 1: Vision phase of proposed Waterfall cycle.

Our Recommended Practices for Waterfall

- Business Requirements Document (BRD) should have a section for "Business Integrations", which is filled in with key details of all identifiable integrations at a business level (business process, data flow, integration class, etc.);
- DS Non-Functional Requirements (NFR) should be documented in the BRD for the application being developed and for each of its integrations (level of availability needed, number of simultaneous connections, data volume and data periodicity, etc);

- Project Architecture Document (PAD) should have a DS Section, containing visual, incremental architecture information. Business requirements are mapped against integration / infrastructure requirements;
- PAD should include applicable integration/infrastructure technical information, such as data format, data contract, security measures, error handling and logging 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 are appointed for each integration;
- Risk Register (RR) should start with a default list of DS risks. The list is continually refined by Project Management Office (PMO) through feedback coming from live projects. It becomes available for upcoming projects;
- All documents above are potential inputs for the Project Schedule and Budget (PSB);
- PMP, BRD and PSB are baselined.

How These Practices Support Waterfall?

- Provision of 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.

Adapting These Practices for Scrum

- Product Owner identifies integration needs before Project start;
- Integration / infrastructure needs are discussed during the first project meeting, usually the Release Planning one;
- Infrastructure and support teams are encouraged to be on-board the discussion already in this phase, early in the project;
- Integration / infrastructure requirements should be treated as user stories, added to the Product Backlog and prioritized according to their value;
- Definition Of Ready (DOR) should take in consideration the DS characteristics of the project in question. For example, it could include "complete data contract being available" and/or "data sample being available".

How These Practices Support 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 will tend to be more stable, as DS key characteristics receive proper attention. Value delivered and perceived increase.

3.1.2 Planning Phase

In this Phase, one executes final project planning based on information now available. Full architecture and infrastructure requirements assessments are now possible, and any gaps before execution should be addressed. System design is complete. An overall view of the Planning activities, with its inputs and outputs can be seen in Figure 2.

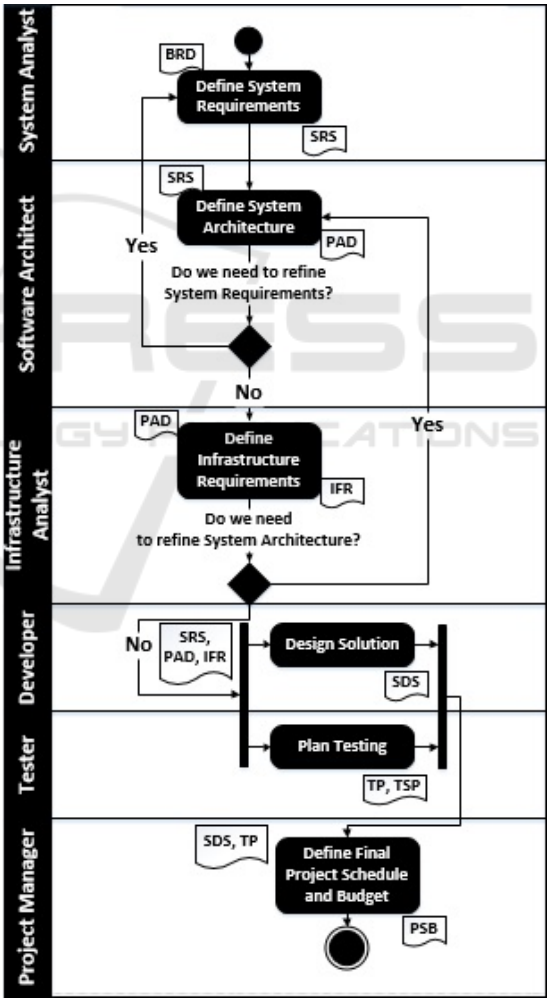

Figure 2: Planning phase of proposed Waterfall cycle.

Our Recommended Practices for Waterfall

- PAD registers the detailed business process flows that will support the solution;

- The Infrastructure Document (IFR) should register detailed information on required infrastructure. Hardware, software and networking needs are mapped, especially the ones affecting system distribution, such as servers' latencies and locations, ports and protocols, etc;
- RR is updated with new risks, including DS ones;
- System Requirement Specification (SRS) must be created and kept in close alignment with BRD and PAD, thus making sure no previoulsy raised system distribution key definitons are lost. These should instead only be incremented in the SRS, thus making the work to create this artifact easier;
- Test Plan (TP) must include detailed information 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 in the Project's critical path;
- Test specification must be created and kept in close alignment with SRS, thus making sure no previoulsy raised system distribution key definitons are lost;
- System Design Specification (SDS) must be created and kept in close alignment with the SRS, thus making sure no previoulsy raised system distribution key definitons are lost. They should instead only be incremented in the SDS;
- SRS, PMP and PSB are updated and re-baselined.

How These Practices Support Waterfall?

- End of Planning phase has all major solution specifications and a complete design, all considering the DS characteristics of the project;
- Improved visibility acquired regarding what are the main technical constraints and risks for the rest of the project, before execution.

Adapting These Practices for Scrum

- There should be acceptance criterion created for each infrastructure / integration story, such as:
  - 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 are treated as user stories and are added into a Sprint Planning scope, if Ready criteria is met.

How These Practices Support 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 characteristic.

3.1.3 Building Phase

In the Building Phase, one assembles the required Non-Production and Production infrastructures, as well as creates the software product through codification and developer´s level testing. Finished Test Cases are also an output of this phase. An overall view of the Building phase activities, with its inputs and outputs can be seen in Figure 3.

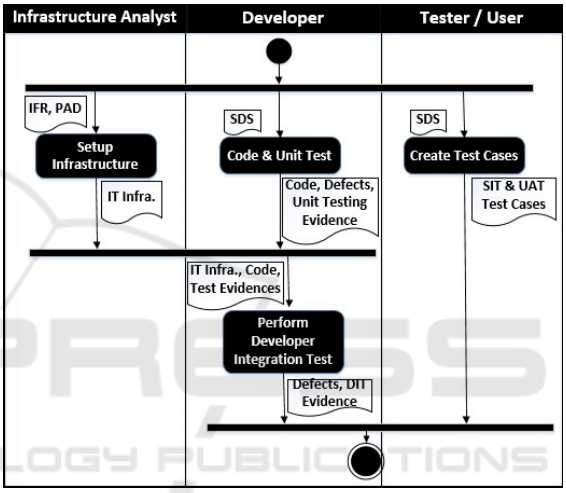

Figure 3: Building phase of proposed Waterfall cycle.

Our 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;
- Logging and monitoring functionalities must be implemented according to the strategy previously mapped in the PAD. This allows easier traceability of defects in non-production environment, as it will be possible to quickly identify from which application the defect comes. Also, when in production, traceability of incidents will also be benefited by the same approach;
- Developer Integration Test (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 what will be found in production.

How These Practices Support Waterfall?

- Completion of the development step with a much more stable code, mainly due to attention given to key distribution details.

Adapting These Practices for Scrum

- Sprint Zero includes the assembly of non-production infrastructure;
- Sprint Zero includes test analysis for test scenarios generation. Next sprints have the same approach. This generates better coverage during test execution.

How These Practices Support 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.

3.1.4 Testing Phase

In this phase one performs detailed integrated testing from both the test team's and user's perspectives. Defect management and handling happen during the entire phase. Performance testing, when applicable, is also carried out on this phase. An overall view of the Testing Phase activities, with its inputs and outputs can be seen in Figure 4.

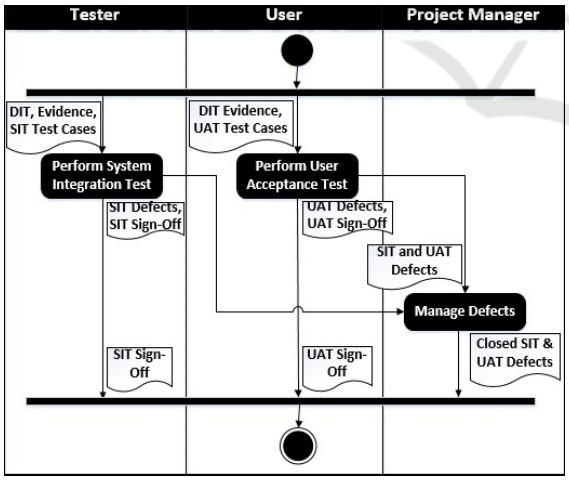

Figure 4: Testing phase of proposed Waterfall cycle.

Our Recommended Practices for Waterfall

- Testing should provide an important focus on the Non-Functional Requirements (NFRs), considering they highly influence the system distribution decisions;
- Mocked data should be avoided at this stage. The use of data masses that are the closest possible to production is encouraged;
- Mocked integrations should be avoided at this stage. It is ideal to have all systems integrated in the testing non-production environment;
- Test infrastructure and overall environment in use must be the closest possible to production;
- Sign-offs should be received from who is performing the tests by the end of System Integration Testing (SIT) and User Acceptance Testing (UAT).

How These Practices Support Waterfall?

- An independent test team will validate what was built and delivered;
- Stabilization of defects prior to handing the system over to users for UAT testing;
- Realistic testing will help in preventing many incidents in production.

Adapting These Practices for Scrum

- Production environment can be raised and be continuously refined at this point;
- Proposed waterfall test practices can be used equally in Scrum, without adaptations.

How These Practices Support Scrum?

- The benefits are the same coming from the proposed practices in Waterfall Testing phase.

3.1.5 Releasing Phase

In this phase one provides the support team and users with the application training. Application is made available for use in production. Provision of warranty for the application, through the solution of production incidents. Project closure is executed. An overall view of the Releasing Phase activities, with its inputs and outputs can be seen in Figure 5.

Our 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;
- A post-deployment plan should be available in order to help validating if all core functionalities from the deployed / integrated systems are unnafected and available;
- A "System Profile" Document (SPD) describes, in business terms, the implemented system, its purpose, integration points, data flowing in and out, etc. This is the base of the Knowledge Transfer (KT) for the support team and users;

- Lessons learned document captures learned items that will be inputs to upcoming projects. A DS section exists in the document.

How These Practices Support Waterfall?

- Project closure occurs when the system is fully transitioned to production and accepted by users;
- System transition to the support team is also needed for closing out the project.

Adapting These Practices for Scrum

- If there is not enough time in last sprint, then create a "Sprint-F" (for Final), for carrying out KT and the remaining documentation, including SPD;
- Lessons learned are filled out as part of final Sprint Review and Sprint Restrospective, using Sprint-F for that as well, if needed.

How These Practices Support 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 there is the foment of KT for that purpose;
- Continuous improvement of projects through the raise of lessons learned.

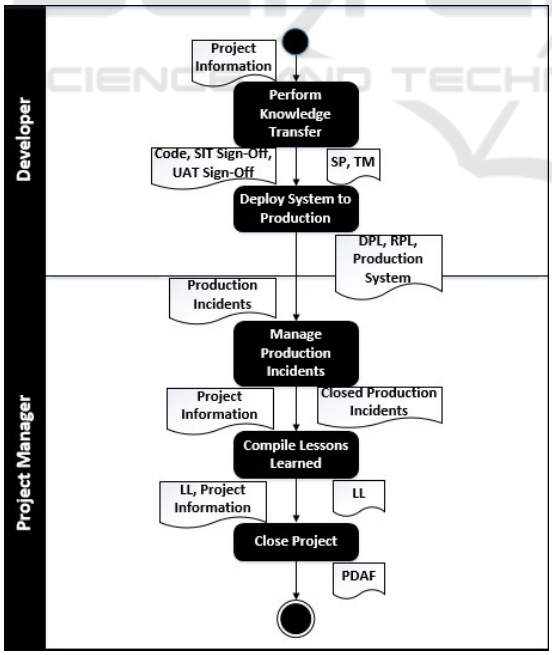

Figure 5: Releasing phase of proposed Waterfall cycle.

3.1.6 Monitoring and Controlling Phase

This phase happens in parallel to the project, providing oversight for all phases. Change impacts are monitored, action being taken when needed and status being reported. An overall view of the Monitoring and Controlling (M&C) Phase activities, with its inputs and outputs can be seen in Figure 6.

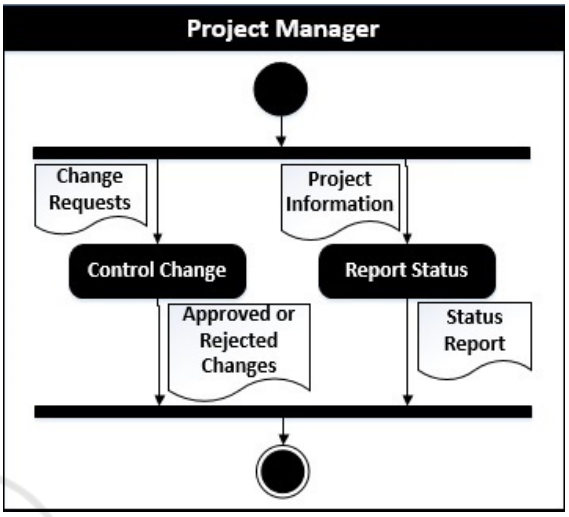

Figure 6: M&C phase of proposed Waterfall cycle.

Our Recommended Practices for Waterfall

- Status reports addresses the DS aspect, describing what is the status on infrastructure as well as on each integration. Items such as difficulties faced, steps completed, opportunities, risks, teams engagement, etc should be on the report.

How These Practices Support Waterfall?

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

Adapting These Practices for 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.

How These Practices Support Scrum?

- "All" project aspects become visible to everyone, including the ones related to DS.

3.2 Validation and Limitations

We define this research as an empirical, qualitative one. As such, our DS SDLC and practices, after created, went through the process of "*member checking*", a traditional validation technique used in

empirical work (Singer, 2007).

We invited 5 participants from the 16 IT professionals, who had previously participated on our interview-based field study, to participate again in the validation session. They were chosen due to the authors' perception of their highly critical opinions as well as the importance of their previous contibutions.

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 was encouraging. These professionals all agreed that many practical benefits should come from the implementation of our proposed SDLC and practices. All of them also had their own inputs with improvements, which in turn led to the version of our model discussed in this paper.

We did not include in preliminary validation the application of the SDLC in real-life projects given time constraints. However, the field study provided us with an initial rich data set that suffices before we continue our work. Our next step is, then, to observe how the SDLC is welcomed in real-life projects and what suggestions industry professionals will make to improve and further its scope, if any. For now, the current limitations of our study are as follows:

- No practical experiments with real projects and/or companies conducted so far;
- Field study participants were from Brazil and the United States only while member checking participants were from Brazil only;
- Small diversity of companies, (one American company providing 9 out of 16 participants);
- Our SDLC currently does not drill down to task-step structure;
- The SDLC has some generic management activities and deliverables. These will migrate into an independent PLC in the future.

4 CONCLUSIONS

In this work we discussed the many challenges brought by DS to Software Development Projects. Little research exists though on the intersection of Distributed Systems and Project Management.

As presented in our results, professionals from the IT industry do recognize the importance of understanding those challenges and taking systematic actions in order to mitigate or eliminate most of them.

We believe that our SDLC and related practices are in line with the industry needs for an effective countermeasure for the identified challenges, addressing them by broadening project teams´ awareness about the importance of properly handling the System Distribution aspect on the projects they are inserted in.

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.

More research is still needed for verifying the effectiveness of our proposals, as well as their easiness of use, both when used by themselves as well as when simply coupled to other SDLCs. This is the cornerstone of our research's next step.

REFERENCES

PMI Inc., 2015. What is Project Management?. Consulted on October 13, 2015. Available at http://www.pmi.org/about-us/about-us-what-is-project-management.aspx.

PMI, 2013. A Guide to the Project Management Body of Knowledge: PMBoK Guide. *Project Management Institute. Pennsylvania, 5th edition*.

Coulouris, G., Dollimore, J., Kindberg, T., Blair, G., 2012. *Distributed Systems Concepts and Design, Addison-Wesley, Boston, 5th edition*.

Taylor, J., 2004. Managing Information Technology Projects: Applying Project Management Strategies to Software, Hardware and Integration Initiatives. AMACOM. New York.

Version One, 2015. "*The 9th Annual State of Agile Report*". Consulted on October 07, 2015. Available at http://info.versionone.com/state-of-agile-development-survey-ninth.html.

Herbsleb, J. D., Moitra, D. 2001. Global Software Development. *In IEEE Software, V.16, n.2, 16-20*. IEEE.

Pressman, R, Maxim, Bruce, 2014. Software Engineering A Practitioner's Approach. *McGraw-Hill. New York, 8th edition*.

Shull, F., Singer, J., Sjoberg, Dag, 2007. Guide to Advanced Empirical Software Engineering. *Springer. New Jersey, 2008 edition*.