Productivity Gains of DevOps Adoption in an IT Team: A Case Study

Miguel Angelo Silva *marsa@iscte-iul.pt*
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal

João Pedro Faustino *jpcfo11@iscte-iul.pt*
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal

Rúben Pereira *ruben.filipe.pereira@iscte-iul.pt*
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal

Miguel Mira da Silva *mms@ist.utl.pt*
Instituto Superior Tecnico (IST), Universidade de Lisboa (UL)
Lisbon, Portugal

Abstract

The main purpose of this research is to determine the productivity gains from the merge of two traditional IT teams: Development and Operations into a single DevOps team implementing 6 DevOps capabilities
In this research the authors go through the existing DevOps literature to better frame this research to the reader. To answer the formulated research questions the authors analyzed the team capacity divided by tasks before and after a DevOps transition and interviewed 5 senior team members to collect their opinion about this transition. The main object objective is to analyze if there were productivity gains of that team after making the transition to a DevOps approach.
Keywords: DevOps, Agile, SCRUM, Continuous Delivery, Continuous Integration

1. Introduction

Many organizations which develop and use Information Systems make a structural division of their software departments. One pattern which is often repeated is the separation between software development and system operations [5]. Historically there is a gap of collaboration between Dev and Ops. Development team members often have an attitude where change is the thing they have to achieve whereas Operations team members often have an attitude of avoiding change in anticipation of maintaining stability of systems and/or applications. With the increasing popularity of Agile Software Development, we are seeing a trend towards an increase in the amount of development release cycles. The result of which is that the traditional setup of a defined split between Ops Teams and Dev Teams each pursuing their own differing objectives is no longer efficient. While Ops teams represents the running side of IT Services, its objectives are to preserve the operational status of the IT Service and to provide that same continuous IT service to the Business. The Dev team is responsible to interrupt the running of the IT Service on behalf of the business with the objective to deploy new features to that same IT Service [5]. These continuous obstacles and problems lead to the rise of a new approach called DevOps. DevOps is a clipped compound of Development (Dev) and Operations (Ops) of business or technology systems and applications and was originally defined by Patrick Debois and Andrew Shafer in 2008 [5].

1.1. Problem and Motivation

While having two separate teams doing Development and Operations problems and lack of synergies can occur. Separations on a technical, organizational level and use of different tools had been arising between Dev and Ops teams. These separations make the fluent and coherent communication, collaboration and issue resolution between the two teams difficult if not impossible [1]. This lack of cooperation and communication between development and operations personnel results in uncoordinated activities [19]. In the Development side poor communication between the development and operations teams produces undesirable results. Non-functional requirements e.g. performance or availability might be overlooked as the responsibility of running the product is shifted to the operations team, letting developers with a lack of knowledge of what is really happening in Production systems, and without proper access to it and to error logs, developers become frustrated and lack the complete picture of the problems that can occur in those same productive systems [19]. On the Operational side, the lack of IT Operations involvement in the requirements specification and also a poor knowledge transfer from the Development side, or sometimes even no knowledge transfer at all, brings major issues when running productive systems [11].

1.2. Research Questions

The question posed in this research is the analysis of the impact in the time spent per each activity when passing from a traditional approach (Development team separated from Operations Team) to a DevOps approach. The analysis was achieved by comparing the productivity situation before and after the DevOps merge, it was measured by a Key Performance Indicator (KPI) before and after implementing DevOps capabilities. To fulfil the proposed objective this research also presents a research question which must be answered. The research question can be seen at Table 1.

Table 1. Research Question

RQ 1	What are the main productivity gains when merging Development and Operations teams into a DevOps team?

2. Theoretical Background

DevOps has not yet been properly studied in scientific literature, limited research articles exist regarding DevOps. There is no universally agreed upon definition of DevOps or DevOps process to be followed, which makes it difficult for companies to adopt the best practices of DevOps [5] [11]. However, we can see DevOps as a conceptual framework that is supported on a culture of collaboration, automation, measurement, and information sharing. DevOps is not a one size fits all solution and it will not act as a silver bullet to solve all IT problems within a company. It should be faced as an adapted artifact to match the unique challenges of each specific environment. [2]. The reviewed literature advocates that DevOps brings advantages over the traditional methods of software development, but at the cost of several significant challenges as a cultural shift will be required by the Company[3].

To understand the importance of DevOps in major companies, International Data Corporation (IDC) has conducted an insight that demonstrates the usage of DevOps within the Fortune 1000 organizations with revenue of at least $1.39 billion. The result was that 43% of the respondents are currently using DevOps practices, while another 40% are currently evaluating DevOps implementation. [16].

2.1. People, Processes and Technology

Even though there is not a defined and standardized definition of DevOps and its related processes, some authors identify the cornerstones of DevOps as People, Processes and Technology [14].

People: DevOps at his core can be considered as a cultural movement, carrying changes on the way people work. The relations between colleagues should be based on trust and confidence. Transparency should be faced as the rule of thumb for a DevOps team. The members of the team should also have common goals and incentives, and not only developers for delivering in time, with quality new features and operations personnel for having an uptime of excellence.

Processes: The DevOps process can be considered a business process because it aims to affect the entire lifecycle of an application as being a collection of activities or tasks that produce a specific result for customers. When the DevOps approach is in place within an organization, all parties involved from the highest level of the business down to the operations should be able to have transparency and cooperate in the entire lifecycle of a change [14].

Technology: Automation is an essential keystone of DevOps, it guarantees that the creation, testing and deployment of the software is always done in the same way, this avoids defects created by human error. Automation speeds up software delivery and increases quality by reducing human errors and providing more frequent and earlier feedbacks.

2.2. The DevOps Reference Architecture

The reference architecture mentioned on [14] provides a template of a proven solution by using a set of capabilities. This architecture helps practitioners access and use the guidelines, directives and other material that is needed to design and create a DevOps platform that accommodates people, processes and technology as described in Figure 1.

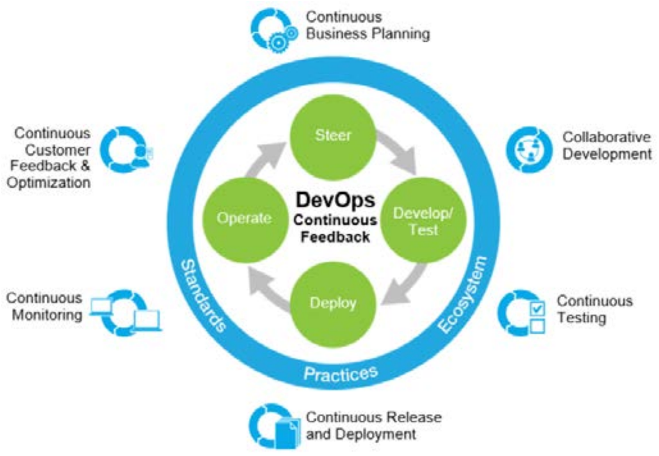

Continuous Business Planning
Continuous Customer Feedback & Optimization
Collaborative Development
Steer
Operate
DevOps Continuous Feedback
Develop/ Test
Deploy
Standards
Ecosystem
Practices
Continuous Monitoring
Continuous Testing
Continuous Release and Deployment

Figure 1. The DevOps Reference Architecture [14]

The proposal of this reference architecture is to follow a constant workflow of steering, which consists in establishing business needs and adjust them taking into consideration the feedback from the users. Development / testing where is included the quality assurance in the development process and contains the practices which allow the pipeline of software deployment. Operations has the responsibility to monitor the performance of applications and funnel the users feedback in order to get information for the business and if needed change the direction of the business strategy. Completing this architecture and expanding upon the core elements of DevOps we have 6 capabilities that compose the core elements, those capabilities are explained below [14].

a) Continuous Business Planning: Practice that focuses on establishing business goals and adjusting them based on customer feedback. In a traditional software development approach, the information needed to define a correct strategy is fragmented and inconsistent due to the low automatization and processes standardization, the feedback is not received on time to be incorporated on the next release, failing this way to deliver value to the customer.

b) Collaborative Development: Practice that aggregates all the elements of the different teams in the process of Software Development: Business owners, business analysts, enterprise and software architects, developers, QA practitioners, operations personnel, security specialists, suppliers, and partners. Practitioners from these teams work on multiple platforms and may be spread across multiple locations.
c) Continuous Testing: Means to test as soon as possible and continuously during the development lifecycle, this leads to a development cost reduction as well to a better software quality. This practice is viable using techniques such as test automation and virtualization to simulate the production environments for the tests to be executed in a scenario more real as possible [14] [12].
d) Continuous Release and Deployment: The objective is to allow new functionalities to be deployed as fast as possible. It was this practice that originated the DevOps movement; this capability brought the concept of continuous integration to the next step allowing the possibility to create a complete automated pipeline of new features delivery in production [14].
e) Continuous Monitoring: Collects data and metrics that are coming from the different stages of the application lifecycle which allows all the parties involved to react fast to improve or modify the functionalities which are being used [14].
f) Continuous Customer Feedback and Optimization: The new technologies provide the ability to monitor the customer behavior which allows the business team or any other interested parties to take the necessary actions to improve the software.

3. Related Work

In Section 2 we have shown that DevOps has not yet been properly studied in the scientific literature. The such lack of studies (and low maturity) makes it difficult to companies to adopt DevOps since they might not know which practices or process they should implement [2] [11]. Nevertheless, Table 2 presents some case studies where DevOps was applied.

4. Research Methodology

Case Study research is the preferred method when a question "why" or "how" is being done over a set of contemporary events [20]. These general questions are meant to open the door for further examination of the phenomenon observed as well as to start a study on a determined phenomenon observed where there is not prior work [4, 5]. As mentioned in section 2.1, DevOps has not yet been deeply studied among the literature, in this scenario this research uses Case Study research methodology, this type of research is used to study determined phenomenon observed in areas with lack of research [20] as well as questions like "what" from an exploratory nature to develop propositions for future researches.

Considering the above scenario, the used research methodology intends to be an exploratory Case Study, which is compatible with the questions that are stated in this research [9] [20, 21]. This research is a single case study since the focus is a single team, which is a single unit of analysis as described by [20].

Triangulation means taking different angles towards the studied object and thus providing a broader picture [7] [13] [21]. This method is important to increase the precision of empirical research and to limit the effects of one interpretation of one single data source. If the same conclusion can be drawn from several sources of information (triangulation) this conclusion is stronger than a conclusion based a single source. During this research the authors have performed interviews with a subset of team members to collect their analysis from a qualitative point of view. The authors have also had access to productivity reports and documentation.

Table 2. Generic Information About DevOps Case Studies

CS ID	Title	Bibliography	Focused Capability

CS.1	Tool Support for Traceability Management of Software Artefacts with DevOps Practices	[8]	N/A
CS.2	Adopting DevOps Practices in Quality Assurance	[12]	Continuous testing
CS.3	End to End Automation On Cloud with Build Pipeline: The case for DevOps in Insurance Industry	[15]	Collaborative development Continuous testing Continuous release and deployment
CS.4	DevOps in Regulated Software Development: Case Medical Devices	[6]	Continuous release and deployment Collaborative development
CS.5	User Stories to User Reality: A DevOps Approach for the Cloud	[10]	N/A
CS.6	DevOps for IoT Applications using Cellular	[4]	N/A
CS.7	Agile Implementation in a Large, Regulated Industry: DevOps and Accelerating Delivery Case	[17]	Continuous release and deployment Continuous testing
CS.8	DevOps and Moving to Agile at a Large Consumer Website: Getting Faster Answers at Yahoo Answers Case	[17]	Continuous release and deployment Continuous testing

As suggested by Tellis and Runeson [13] [18], this case study is divided in four stages (Table 3). The remaining document is organized according the Case Study phases presented on the same table.

Table 3. Case Studies Stages adapted from [13] and [28]

Stage	Stage Description
Design the Case Study	This stage aims to define the case study objectives and to plan the case study itself.
Conduct the Case Study	Preparation of the data collection, procedures and protocols for data collection is defined and execution of the data collection takes place.
Analysis of collected data	Define an analytic strategy to evaluate the data gathered in the previous stages of the research
Develop Conclusions	Develop conclusions regarding the data analysis made on the previous stages to establish a bridge between the researcher and the user to explain the benefits or problems found during the research.

For the case study validity, the authors have followed the 4 validity tests proposed by Yin. Table 4 synthesize this research validity under Yin's tests.

5. Design the Case Study

The objective of this research is to answer the research question formulated in Table 1. This research was conducted in a multinational company acting on the areas of Industry Automation; Electrification and Digitalization with approximately 365.000 employees, this company has 3 IT nearshore centers, distributes across the world, this research took place in an IT nearshore center located in Portugal which provides global IT services for internal customers. This company has in internal ITSM tool that supports the ITSM processes, this tool is named Service Now (www.servicenow.com), to support this application and following the traditional approach two teams were created in 2013 with two different purposes:

- Operations Team: With 4 members, the goal of this team was to monitor the application regarding interfaces with external providers, maintain the frontend Service Catalog (Catalog management), managing and deploying new releases with new features and correction of bugs (Release and Deployment), patch management and 3rd level incident resolution.

- Development Team: With 13 members, the goal of this team was to develop new features for the application and corrective maintenance as well as keep the application version up to date (migrate the application for newer version whenever required)

Table 4. Validity tests adapted from Yin [20]

Test	Description
Construct Validity	Multiple sources of evidences were used. The authors have conducted semi-structured interviews and have also analyzed some reports.
Internal Validity	Yin states that this test should not be applied to exploratory case studies. Since this case study is exploratory, this test wasn't addressed by the authors.
External Validity	The authors have analyzed the literature (Section 3) and have not found any research about a complete DevOps implementation. Therefore, it proves the novelty of this research and the relevance of our findings for the body of knowledge.
Reliability	A path was created during all the research showing how the researchers have lead their investigation, so future researchers can proceed with the investigation and get similar results. Yin [20] guidelines were adopted all over the performed case study and respective report.

After a management decision in 2017 this two teams were merged into a single DevOps team with the goal to improve productivity and lower costs, at the time of the merge all the members of the 2 previous teams were included, creating a DevOps team of 13 members. After the merge the DevOps the implementation of the capabilities presented in section 2.2 started immediately. The implementation of these capabilities was divided into two phases:

Phase1: It was decided to implement 4 capabilities that the company considered to be the less complex to implement: Continuous business planning, Collaborative development, Continuous monitoring, Continuous customer feedback and optimization.

Phase2: After a market research to find the appropriate tools to support this phase, the last two and more complex capabilities were implemented: Continuous testing and Continuous release and deployment.

The decision of splitting the DevOps implementation in two phases comes from the manager of this team which has decided to do a phased approach instead of a big-bang approach, the decision had several reasons behind: The fact that the customer is very conservative with a high resistance to change, to avoid disruptions in the quality of the delivered service and lastly by technologically. As a result there was only the possibility to start immediately with Phase1 because all the necessary tools were already available and while implementing Phase1 the tools to give support to Phase2 implementation should be assessed and acquired from external vendors.

To measure the performance of the team before and after the transition to a DevOps approach, a Key Performance Indicator (KPI) was measured before (our baseline) and after that same transition (AS-IS). We called that KPI: "Percentage of time spent by the team grouped by activities"

This KPI was defined by the company management as the strategic KPI to be measured and by consequence evaluate the success of this merge. According to the company management a positive result of this KPI is to lower the time spent on the activities: Operational Work and Defect solving and to increase the time spent in the activities: Stories, Business Analysis, Architecture and Automated Testing. The KPI was calculated in Excel by the researchers using the documentation collected in this research as explained in section 6. The detailed analysis of this calculation can be seen in section 7.2. The KPI was calculated and analysed before and after each DevOps implementation phase and detailed in Table 5.

The focus of this research is the analysis of the KPI delta only after implementing Phase1 because Phase2 implementation is still under implementation and we have no data at the time this paper is being written, nevertheless it is mentioned as future work in section 8 this research including Phase2.

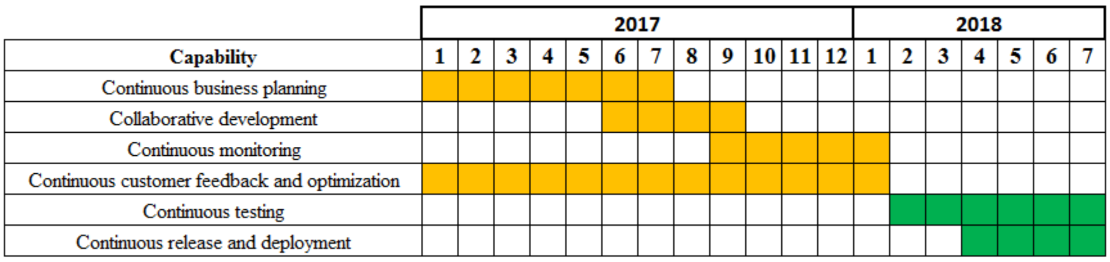
Due to space constraints some topics weren't covered: How each capability was implemented, division between development and operations activities and how the company has promoted a unified collaboration between teams.

From a timeline perspective Table 5 describes the phases on which the core activities and correspondent capabilities were implemented and respective time for implementation.

The granularity of the timeline for the implementation of each capability is also listed and is presented in figure 2. This timeline comprehends the period between January 2017 when the merge of both teams (Development and Operations) has started, and July 2018 and the merge was finished, when we considered that the teams are merged and are only one team and after the 6 capabilities are implemented and live.

Table 5. Phases of DevOps implementation

Phase	Capability	Core Activity	Implementation time
1	Continuous business planning	Steer	12 Months (Jan 2017 – Jan 2018)
1	Collaborative development	Develop/Test	
1	Continuous monitoring	Operate	
1	Continuous customer feedback and optimization	Operate	
2	Continuous testing	Develop/Test	6 Months (Feb 2018 – Aug 2018) [Predicted]
2	Continuous release and deployment	Deploy	



	2017												2018						
Capability	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7
Continuous business planning	🟧	🟧	🟧	🟧	🟧	🟧	🟧												
Collaborative development						🟧	🟧	🟧	🟧										
Continuous monitoring									🟧	🟧	🟧	🟧	🟧						
Continuous customer feedback and optimization	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧	🟧						
Continuous testing														🟩	🟩	🟩	🟩	🟩	🟩
Continuous release and deployment																🟩	🟩	🟩	🟩

Figure 2. Timeline implementation with a monthly granularity per capability

The data used in this research was collected from two sources:

1. Using the team capacity planning information for 2017 and 2018, where are planned all the team hours with a breakdown by activities: Meetings, Development (Stories and Defects), Release and Deployment, Team Management, Business Analysis, Automated Testing, Architecture, Operational Work. This information allows us to infer all the hours planned for the team by task and provide input for the KPI calculation.
2. This research also present data collected in interviews with 5 senior team members which had hands-on experience during the time of this merging process. Yet due to space constraints we have combined only the most important parts of the interviews and detailed them to section 7.

6. Conduct the Case Study

In this stage the authors gathered, compiled and calculated all the data needed to analyse the KPI mentioned in section 5. The interviews conducted to the 5 senior members of the IT team had the purpose to get their own conclusions about this merge. In section 7 those testimonies are crossed with the analytical analysis of the KPI evolution over the months.

To collect the team planning capacity, we have been on site in the organization headquarters in Portugal, every month at least 2 days since the beginning of this research (January 2017). This information was provided by the team manager regarding year 2017 and 2018 (until February) in excel and with the detailed hours spent by each type of activity.

The interviews were also done on site, via questionnaire in January 2018, exactly one year after the merge has started where the open questions presented in table 6 were asked. As already stated in Section 5 due to space constraints, only the most important and relevant parts of the interviews are highlighted on this research, yet all the fundamental information for this research extracted from those interviews is in Section 7.

Table 6. Interview Questions

Question Number	Question
Q1	What is your general opinion about the baseline situation (before the DevOps merge)
Q2	Comment the percentage of time spent on each activity in the baseline situation
Q3	What is your general opinion about the AS-IS situation (After implementation of Phase1)
Q4	Comment the percentage of time spent on each activity in the AS-IS situation

7. Analysis of collected data

On this stage it is supposed to analyse the data that was collected on the previous stage. The Analysis is divided in two different times of the implementation timeline:

1. Baseline Status: Status of the team with time spent for each activity before the merge.
2. Status after Phase1: Status of the team regarding time spent for each activity after the 4 capabilities are implemented.

7.1. Baseline Status

As a baseline, before the teams were merged, operational work consumed the majority of the activities according to the answers of the team members to question Q1 presented on table 6: "*Due to the high amount of time invested in operations there was no possibility to put more effort in activities considered crucial to increase the team service quality as well evolve the team to more valuable activities than operational work: Business Analysis, Automated Testing and Architecture. Also the operational work has a lot of manual repetitive tasks which are of no interested and demotivate the team members who have to do those tasks. This situation had to be changed and to increase the budget was not considered by the company management, with this scenario in the table the decision to move to a DevOps approach was taken with the vision that this change would allow to better optimize the team capacity*".

Table 7 shows the percentage of time spent by the team for each activity as well the most relevant comments of the team members. This table contains the answer to Q2 (table 6).

7.2. Status after Phase 1

After Phase1 was implemented the authors have compared the differences of percentage of time spent per activity in the baseline situation and after phase 1, to have a more graphical overview about this situation a report was created and is showed in Figure 3.
After the 4 capabilities were implemented there was some improvements noticed by the team members according to the answers to Q3 presented on table 6: "*The implementation of the 4 capabilities has marked a milestone, phase 1 was completed and the results were shared with the management. The operational work was dramatically reduced which allowed to shift the team capacity to more valuable tasks, this was a positive point because with the same budget the team provided more services. This decrease on the operational work was also a strong driver for the management to decide implement DevOps in other projects inside the company and boosted the team motivation because some very concrete result could be seen. The management also communicated to the team that the end user satisfaction perception also improved significantly. As a negative side the team took too long to implement 4 DevOps capabilities: 12 months, the reason for this was the lack of experience of the team in the DevOps framework as well some obstacles caused by the customer that also had difficulties in adapting to this new mindset. The team aims to implement phase 2 in 6 months, even if the 2 remaining capabilities are complex to implement they now have the experience from phase1*".

Table 7. Comment collected from the team members of the Baseline status of the team

Activity	%	Team members Comments
Meetings	13%	*“This percentage is expected and aligned with the team members work experience.”*
Defects	17%	*“At this moment the team consumed a high number of hours in solving Defects, this was one of the numbers that the team wanted to reduce to release the team to develop new features as well increase customer satisfaction.”*
Stories	14%	*“The time used for stories was very low at this moment, this number should increase to provide more features to the end user in a faster way and with a higher quality.“*
Release and Deployment	4%	*“This percentage is expected and aligned with the team members work experience.”*
Team Management	3%	*“This percentage is expected and aligned with the team members work experience.”*
Business Analysis	0%	*“Crucial activity that was not being done by the team. Hiring a Business Analyst was not a solution due to budget restrictions, so the idea is to optimize the team to reduce operational work and free time for Business Analysis.”*
Automated testing	0%	*“Crucial activity that was not being done by the team. Hiring a Tester was not a solution due to budget restrictions. This activity is crucial to implement phase 2, without automated testing phase 2 could never be implemented. In this situation the idea is again to optimize the team to reduce operational work and free time to implement Automated Testing.”*
Architecture	0%	*“Crucial activity that was not being done by the team. Hiring an Architect was not an option due to budget restrictions, so the idea is to optimize the team to reduce operational work and free time for Architecture tasks.”*
Operational Work	50%	*“Too high percentage, the best practice stands that only 20% of a team capacity should be used for operations. Operational, work is demotivating and from a financial point of view is negative. The goal is to reduce this percentage to 35% or less after phase1 is finalized.”*

Figure 3. Comparing Baseline and Phase1

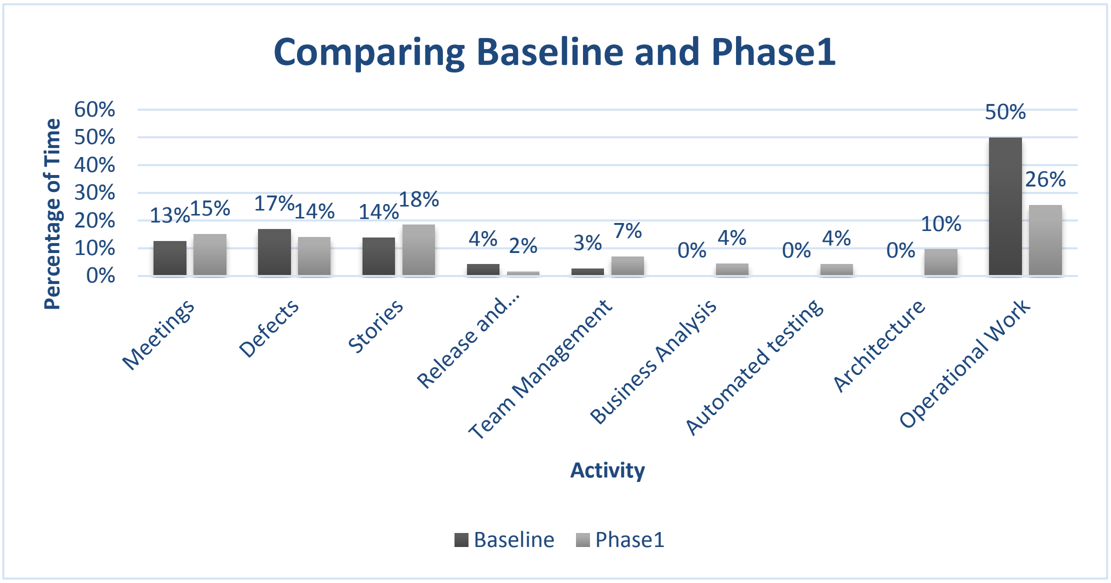


Table 8 shows the percentage of time spent by the team for each activity, the delta with the baseline status as well the most relevant comments of the team members. This table contains the answers to Q4 presented on table 6.

8. Develop Conclusions

In section 7 a breakdown of the percentage of time applied to each activity of the IT team was presented. It was also shared the views of 5 team members, which had a close contact with the transition, crossing these two data sources some conclusions were derived.

Table 8. Comment collected from the team members after Phase 1 implementation

Activity	%	Δ% to Baseline	Team members Comments
Meetings	15%	+2%	*"It was noticed a need for more communication among team members not only because we are working on a different approach, but one of the DevOps core is communication so it was expected an increase of meetings that the team used to align objectives. For this increase the capability that contributed the most was Collaborative Development"*
Defects	14%	-3%	*"The percentage of Defects was reduced in 4%, it was expected a bigger decrease, yet a possible cause might be that in 12 months a considerable amount of new services is live and that implies that more code is in the system which can obviously originate more Defects. Anyway, this number is considered a success. For this decrease the capabilities that contributed the most were Continuous Business Planning and Continuous Monitoring."*
Stories	18%	+4%	*"The percentage of developed stories increased 4% this was one of the objectives that the team wanted to achieve and it's a positive indicator. It was expected a bigger increase but along the 12 months it was decided to divide the available capacity to other activities like Business Analysis and Automated Testing and not only stories development. For this increase the capabilities that contributed more were Collaborative Development and Continuous customer feedback and optimization, with the implementation of this last capability one interesting situation happened, the feedback loop between developers and customer was frequent and short so a lot of rework and specification changes was avoided which substantially reduced the waste of development hours."*
Release and Deployment	2%	-2%	*"Contrary to the expected, the time spent for this activity decreased, after analysis we understood that automation had a crucial role in this decrease, the release technical experts automatized almost by complete the entire release and deployment process which lead to a big decrease on the time of this activity. This capability (continuous release and deployment) was not included in phase 1 however by need to avoid a bottleneck the team had to do some work on this area as well. It's not fully implemented but the laying foundations are already in place. Also, a very positive result on this activity."*
Team Management	7%	+4%	*"It was already expected that the need for steering and organization of team could increase first because of some points: DevOps coaching was needed and that tasks is seen as a Team Management task. There was also the need to improve communication with all team members that was now only one joint team. All the 4 capabilities implemented on phase 1 were responsible for the increase of this activity."*
Business Analysis	4%	+4%	*"Very good indicator in this activity, because capacity was release from the operational work the team has shifted capacity to start performing the mentioned activity, the goal is to increase this value after phase 2, yet a very good indicator praised by the management."*
Automated testing	4%	+4%	*"Very good indicator in this activity, because capacity was release from the operational work the team has shifted capacity to start performing the mentioned activity, it's our goal to increase this value after phase 2, yet a very good indicator praised by the management."*
Architecture	10%	+10%	*"Very good indicator in this activity, because capacity was release from the operational work the team has shifted capacity to start performing the mentioned activity, it's our goal to increase this value after phase 2, yet a very good indicator praised by the management."*

Operational Work	26%	-24%	*"The golden nugget of phase1. 24% decrease of the operational work was an impressive mark and clearly above expectations. This fact allowed the shift of the team capacity to 3 very important activities that the team was not performing: Business Analysis, Automated Testing and Architecture. In the future with phase 2 the goal is to decrease this value until 18%-20%. The main DevOps capabilities responsible for this decrease were Continuous Customer Feedback and Optimization and Continuous Monitoring."*

The operational work decreased from 50% of the overall time of the team to 26% and this was the major success of this initiative. The time saved on operational work gave the team the possibility to start doing other kinds of more valuable activities, that were considered crucial for the service delivery quality of the team but that were not being performed due to lack of capacity (most part of the capacity was being spent on operational work): Architecture, Business Analysis and Test Automation.

We also noticed a decrease on defects (-3%) and an increase on development capacity (+4% in stories) and this represents a positive trend in shifting capacity from defect solving to development of new features, which increases delivery quality, increases customer satisfaction with a product that has less bugs and more features. This also has a positive impact in the team engagement because it is always more desirable to create new features instead of correcting bugs.

One interesting result was the increase on the management activities (+4%), yet according to the team members this was because a coaching of the team was needed to perform the merge to a DevOps model and this coaching activity was part of the team manager activities. In the near future when the team is more mature on the DevOps model it is expected that this value will decrease.

It was possible to infer from the interviews that the team members are satisfied to apply these practices due to the agility of DevOps and the involvement of all the stakeholders, they feel their work has more impact and it's recognized by all the organization. Is their feeling that they are now leveraging the business and not being a bottleneck for the business as is the case with traditional IT.

As a negative side we could infer that the transition took more time than expected, 12 months to implement 4 DevOps capabilities, this was because of 2 main reasons, lack of experience on the team side of the DevOps framework and also the difficulty to change the mindset of a very conservative customer that worked in the traditional way since the last 30 years, and resistance to change is always a difficult obstacle.

This research also has some limitations, only 4 out of 6 DevOps capabilities are analyzed on this research, this is because 2 capabilities are still under implementation and the company could not provide yet the data for analysis and this research is based on the transformation of a single team and other perspectives may exist with different results.

The future work for this research will be to analyze the resource planning data after phase2 is completed (implementation of all 6 capabilities) and understand if the positive trend maintains for this team. After we have the data for phase2 we are in conditions to do the complete analysis of the benefits and impacts of this team's merge to a DevOps team.

References

1. Artač, M., Borovšak, T., Di Nitto, E., Guerriero, M., Tamburri, D.A.: DevOps: Introducing Infrastructure-as-Code. In: Proceedings of the 39th International Conference on Software Engineering Companion, pp. 497-498. ACM, Buenos Aires (2017)
2. Erich, F., Amrit, C., Daneva, M.: Report: DevOps Literature Review. Twente: University of Twente (2014)
3. Hüttermann, M.: DevOps for Developers. Apress, New York (2012)
4. Karapantelakis, A., Liang, H., Wang, K., Vandikas, K., Inam, R., Fersman, E., Mulas-Viela, I., Seyvet, N., Giannokostas, V.: DevOps for IoT applications using cellular

networks and cloud. In: Proceedings of the 4th International Conference on Future Internet of Things and Cloud, pp. 340-347. IEEE, Vienna (2016)

5. Kim, G., Behr, K., Spafford, G.: The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win. IT Revolution Press, Portland (2013)
6. Laukkarinen, T., Kuusinen, K., Mikkonen, T.: DevOps in Regulated Software Development: Case Medical Devices. In: Proceedings of the 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track, pp. 15-18. IEEE, Buenos Aires (2017)
7. Modell, S. (2005). Triangulation between case study and survey methods in management accounting research: An assessment of validity implications. Management Accounting Research, 16 (2) 231-254 (2005)
8. Palihawadana, S., Wijeweera, C.C., Sanjitha, M.G., Liyanage, V.K., Perera, I., Meedeniya, D.A.: Tool support for traceability management of software artefacts with DevOps practices. In: Proceedings of the 3rd International Moratuwa Engineering Research Conference, pp. 129-134. IEEE, Moratuwa (2017)
9. Perry, D.E., Sim, S.E., Easterbrook, S.M.: Case studies for software engineers. In: Proceedings of the 26th International Conference on Software Engineering, pp. 736-738. IEEE, Edinburgh (2004)
10. Punjabi, R., Bajaj, R.: User stories to user reality: A DevOps approach for the cloud. In: Proceedings of the 2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, pp. 658-662. IEEE, Bangalore (2016)
11. Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare, L. E., Tiihonen, J., Männistö, T.: DevOps adoption benefits and challenges in practice: A case study. In: Abrahamsson, P., Jedlitschka, A., Nguyen, Duc A., Felderer, M., Amasaki, S., Mikkonen, T. (eds.) Product-Focused Software Process Improvement. PROFES 2016. Lecture Notes in Computer Science vol 10027. Springer, Cham (2016)
12. Roche, J.: Adopting DevOps Practices in Quality Assurance Merging the art and science of software development. ACM Queue 11 (9), 1-8 (2013)
13. Runeson, Per and Höst, Martin. 2008. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study. Empir Software Eng 14: 131 (2009)
14. Sharma, S.: DevOps for Dummies. John Wiley & Sons, Inc, New Jersey (2015)
15. Soni, M.: End to End Automation on Cloud with Build Pipeline: The Case for DevOps in Insurance Industry, Continuous Integration, Continuous Testing, and Continuous Delivery. In: Proceedings of the 2015 IEEE International Conference on Cloud Computing in Emerging Markets, pp. 85-89. IEEE, Bangalore (2016)
16. Stephen, E.: DevOps and the Cost of Downtime: Fortune 1000 Best Practice Metrics Quantified. International Data Corporation (2014)
17. Stoneham, J., Thrasher, P., Potts, T., Mickman, H., DeArdo, C., Limonchelli, T.A.: DevOps Case Studies: The Journey to Positive Business Outcomes. IT Revolution Press. Portland (2017)
18. Tellis, W.: Information technology in a university: a case study. Campus-Wide Information Systems 14 (3), 78-91 (1997)
19. Tessem, B., Iden, J.: Cooperation between developers and operations in software engineering projects. In: Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering, pp. 105-108. ACM, Leipzig (2008)
20. Yin, R.K.: Case Study Research: Design and Methods (4th Edition). Sage Inc., Thousand Oaks (2009)
21. Zaidah, Z.: Case study as a research method. Journal Kemanusiaan 9, 1-6 (2007)