Department of Science and Information Technology

Productivity Gains of DevOps Adoption: A case study

Miguel Ângelo Silva

Dissertation submitted in partial fulfilment of the requirement for the Degree of

Master in Computer Engineering

Director of studies:
Dr. Ruben Filipe de Sousa Pereira, Assistant Professor
ISCTE-IUL

September, 2018

Acknowledgements

My first words of gratitude go to my wife Catarina and my son Diogo, It was crucial to have her support and understanding to complete this work and to my son I will give him back twice the time that I did not spent with him, and I will do it gladly.

My all family that gave me a fantastic education and tools to be a successful person nowadays: my grandparents, my mother and father, and by last the very special person that grew up with me and has taught me many things: My sister Patricia.

Also express my sincere gratitude to my advisor, Prof Ruben Pereira for a continuous support and openness, it’s very easy to work like that.

Resumo

O objectivo principal desta pesquisa é aferir os ganhos de produtividade ao efetuar a fusão de duas equipas tradicionais de IT: Desenvolvimento e Operações numa única equipa DevOps implementando as 6 capacidades de DevOps.

Esta pesquisa foi efetuada utilizado o método Caso de Estudo, o autor efetua uma revisão da literatura para melhor enquadrar esta pesquisa e o leitor. Para responder a pergunta enunciada nesta pesquisa o autor analisou o planeamento da capacidade da equipa dividido por tipo de tarefas antes e depois da fusão para DevOps, foram também efetuadas 5 entrevistas a membros seniores da equipa para recolher a sua opinião acerca desta transiçao para um modelo DevOps. E foi efetuada também a observação no terreno pelo autor do estudo.

A maior contribuição desta pesquisa reside na extração dos resultados da aplicação das capacidades de DevOps num cenário real onde duas equipas foram fundidas numa só equipa DevOps. Como exemplo podemos observar a diminuição das tarefas operacionais das equipas decresceram de 50% para 20%

Palavras-Chave: DevOps, Agile, SCRUM, Automação de Testes, Entrega Continua, metodologia ágeis

Abstract

The main purpose of this research is to assess the productivity gains from the merge of two traditional IT teams: Development and Operations into a single DevOps team implementing 6 DevOps capabilities

This research was conducted using the Case Study methodology, the author goes through the existing DevOps literature to better frame this research and the reader. To answer the formulated research question the author analyzed the team capacity planning divided by tasks type before and after a DevOps transition and interviewed 5 senior team members to collect their opinion about this transition. Observation was also part of this research and it was done by the author. The main objective is to analyze if there were productivity gains of that team after making the transition to a DevOps approach.

The main contribution of this research is to analyze and extract results of the application of the DevOps capabilities in a real-world scenario where two teams are merged into a single DevOps team. As an example, we can observe that the operational work of the team decrease from 50% to 20%

Keywords: DevOps, Agile, SCRUM, DevOps, Agile, SCRUM, Continuous Delivery, Continuous Integration, Agile Methodologies

Publications

The objective of this section is to present the papers submitted and accepted in the context of this research. This information is detailed in Table 1

Table 1 - Submitted papers

Conference	**Rank**	**Decision**
Information Systems Development (ISD 2018)	A	Accepted

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Acronyms

DevOps	-	Development + Operations
IS	-	Information Systems
IT	-	Information Technology
ITIL	-	Information Technology Infrastructure Library
ITSM	-	Information Technology Service Management
PO	-	Product Owners

1. Introduction

Many organizations which develop and use Information Systems to make a structural division of their software departments. One pattern which is often repeated is the separation between software development and system operations (Floris, Chintan, & Maya, 2014). Historically there is a gap of collaboration between Dev and Ops. This gap often manifests itself in many avatars such as stiffer competition, de-motivated teams and excessive time overruns of delivering changes. Development team members often have an attitude where change is the thing they must achieve whereas Operations team members often have an attitude of avoiding change in anticipation of maintaining stability of systems and/or applications. This issue is more complex in case of large organizational programs due to manifold stakeholders involved having conflicting or varied interests (Hussaini, 2015).

This gap of collaboration between Dev teams and Ops teams caused a lack of performance especially concerning deployment of releases, not only from a technical point of view but also on a frequency and speed point of view. With the increasing popularity of Agile software development that leads to an increase number of releases that a traditional setup, where Ops Team and Dev Team have different objectives, was not able to handle on an effective manner. 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. Dev team is responsible to interrupt the running IT Service in behalf of the business with the objective to deploy new features to that same IT Service (Kim, Behr, & Spafford, 2013).

This continuous obstacles and problems lead to the rise of a new approach called DevOps. DevOps is an acronym for Development (Dev) and Operations (Ops) of business or technology systems and applications and it was originally defined by Patrick Debois and Andrew Shafer in 2008 (Kim, Behr, & Spafford, 2013). DevOps is an approach based on lean and agile principles in which business owners and the development, operations, and quality assurance departments collaborate to deliver software in a continuous manner that enables the business to more quickly seize market opportunities and reduce the time to include customer feedback (Sharma & Coyne, 2015). In the context of DevOps traditional software engineering roles are merged and communication is enhanced to improve the production release frequency and maintain software quality (Riungu-Kalliosaari, Makinen, Lwakatare, Tiihonen, & Mannisto, 2016).

DevOps aligns business requirements with IT performance, with the goal of adopting practices that allow a quick flow of changes to a production environment, while maintaining a high level of stability, reliability, and performance in these systems. It is important to emphasize that DevOps is not about standards or tools; it is about enabling communication and collaboration between departments in an organization (Kim, Behr, & Spafford, 2013).

1.1. Problem and Motivation

While having two separate teams doing Development and Operations some problems and lack of synergies can occur. Some distances on a technical, organizational level and use of different tools had been arising between Dev and Ops teams. These distances make the fluent and coherent communication, collaboration and issue resolution between the two teams practically difficult if not impossible (Matej Arta, 2017). This lack of cooperation and communication between developers and operations personnel results in uncoordinated activities (Tessem, 2008). 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 (Tessem, 2008). 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 (Riungu-Kalliosaari, Makinen, Lwakatare, Tiihonen, & Mannisto, 2016).

One of the most impacting consequences of this situation is that many organizations software development projects fail, which are mostly connected to software deployment and delivery. Moreover, most enterprises feel that software development and delivery are critical for the daily business (Bass, Weber, & Zhu, 2015). But a recent IBM survey of the industry found that only 25 percent believe that their teams are effective (Sharma & Coyne, 2015).

DevOps importance is being recognized by different studies performed during the last years (Riungu-Kalliosaari, Makinen, Lwakatare, Tiihonen, & Mannisto, 2016). Developments and Operations teams can integrate and consolidate development processes in order to fine tune the performance of services. Monitoring the production systems at real-time enables developers to react whenever anomalies are detected (Cukier, 2013). On-demand infrastructures and timely feedback from monitoring support continuous software delivery and deployment. Release cycles can shorten to hours instead of weeks and months (Neely, 2013) which is seen as the biggest advantage of DevOps (Callanan, 2016).

Today, the capability to frequently deploy new releases into the production environment has become a competitive advantage for companies (Lwakatare, et al., 2016), enabling faster time to market of new features, increased customer satisfaction, market share, employee productivity and happiness among collaborators, as well allowing orgs to win marketplace, and why? Since technology has become the dominant value creation process and in increasingly important and often primary means of customer acquisition in most business. In contrast organizations that require months or weeks to deploy software are at a significant disadvantage in the marketplace (Kim, Behr, & Spafford, 2013).

We do know that is possible to break the conflict between Development and Operations teams, the proof is that high performance companies like Amazon, Google, Twitter and Netflix are adopting a set of DevOps techniques, and they are routinely deploying hundreds or even thousands of production changes per day, while preserving world class reliability stability and security (Kim, Behr, & Spafford, 2013).

Nowadays there are not many researches about DevOps (Floris, Chintan, & Maya, 2014) and in the literature review done in the context of this research no concrete studies that focus the productivity gains while merging two teams into a DevOps context were found. There is consensus in academic literature and among practitioners that because it is a relatively new concept, many companies are still trying to establish what DevOps is;

what it means to the business; and what its impact will be on business (Langerman & Kamuto, 2017)

1.2. Research Questions

The research question that this research intends to answer is related to the analysis of productivity gains when merging from a traditional approach (Development team separated from Operations Team) to a DevOps approach.

The analysis will be done using a KPI before and after implementing DevOps capabilities since the main objective of this research is to measure the delta of productivity before and after merging the teams to a DevOps approach.

To fulfil the proposed objective this research also presents a research question which must be answered. The research question can be seen at Table 2.

Table 2. Research Question

ID	**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 the scientific literature, few researches exist about DevOps. There is no specific definition of DevOps or process to be followed, which makes it difficult to companies to adopt DevOps since they might not know which practices or process they should implement (Floris, Chintan, & Maya, 2014) (Riungu-Kalliosaari, Makinen, Lwakatare, Tiihonen, & Mannisto, 2016). 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 artefact to match the unique challenges of each specific environment. (Floris, Chintan, & Maya, 2014). Anyway, the reviewed literature advocates that DevOps brings advantages over the traditional methods of software development, as well as creates several challenges due to the fact that a significant cultural change inside the company is needed (Hüttermann, 2012).

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. (Elliot S. , 2014) . This information is detailed in Table 3.

Table 3. Question: How long have you been using DevOps? (Elliot S. , 2014)

Answer	**% of Respondents**
Currently evaluating DevOps practices	40.0
Less than a year	10.0
12-24 months	13.3
25-48 months	13.3
More than 4 years	6.7
Don’t Know	16.7

In the same study it was also found that 36% of the respondents have a DevOps team at their organization (Table 4). Plus, some major companies are using DevOps in several areas (Table 5).

Table 4. Does your IT organization have a defined DevOps Team? (Elliot S. , 2014)

Answer	**% of Respondents**
Yes	36
No	54
Don’t know	10

Table 5. Examples of companies using DevOps (Kim, Behr, & Spafford, 2013)

Industry Type	**Companies**
Technology	Amazon, Twitter, LinkedIn, Etsy and Facebook
Banking	BNY Mellon, Bank of America, World Bank
High Education	University of British Columbia, Seton Hill University, Kansas State University
Government Agencies	uk.gov, Department of Homeland Security

These are only some examples of companies using DevOps, the entire list will not be detailed since its dynamic and it’s not the scope of this research to have such detail.

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 (Sharma & Coyne, 2015).

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 a rule of thumb of a DevOps team. The members of the team should also have common goals and incentives, and not only developers for delivering in time and with quality new features and operations personnel for having an uptime of excellence (e.g. if the PROD application is down it’s also a developer’s problem, as well the success of an uptime of 100% should be credited to the entire team and not only to the operations part of the team). This cultural shift is very important, or else no matter the number of highly efficient processes or tools that an organization has, if there’s a lack of people adherence it will cause impact in the adoption of a DevOps approach. So this means that building a DevOps culture is very important for the DevOps implementation inside an organization (Sharma & Coyne, 2015) (Hüttermann, 2012). From a functional point of view, all team members should have at least a elementary knowledge on the work that is being accomplished by all other colleagues so that a cooperative approach can be taken when solving problems or implementing new solutions (Hüttermann, 2012).

Processes: The DevOps process can be considered a business process due to the fact that aims to affect the entire lifecycle of an application as being a collection of activities or tasks that produce a specific result for customers. Some traditional approaches of change management tend to focus on defects or feature changes, this approach does not allow an integrated end to end management of the entire application lifecycle. When the DevOps approach is in place within an organization, all the involved parties since the business until the operations personnel should able to have the transparency and cooperate in the entire lifecycle of a change (Sharma & Coyne, 2015). Development and Operations should use the same delivery processes from an end to end perspective in order to quickly react to changes and defects and be able to respond faster to the customer needs. Also, one very important principle is that the process derive from Agile and Lean methodologies like SCRUM and Kanban this implies that the team focus should be on the deliverables and not on individual roles (Hüttermann, 2012).

Technology: Automation is an essential keystone of DevOps, it guarantees that the creation, testing and deployment of the software is always done on the same way, this avoids defects created by human error. Automation speeds up software delivery and increases quality by reducing human errors as well more frequent and earlier feedbacks. Another advantage is that the process being automatized is transparent and documented and is repetitive nature makes easy to measure and by consequence define improvement measures on the process (Hüttermann, 2012). Another advantage is to automate manual routine tasks, leaving people to more creative and valuable tasks. A set of automated tasks makes the process more efficient, fast, replicable and scalable. (Sharma & Coyne, 2015). The automation of build, implementation and testing is a keystone to enable short time to market of new features and defects correction. This means that any change can be done via an automated software delivery pipeline, which makes the all process faster and human error free.

2.2. The DevOps Reference Architecture

The reference architecture mentioned on (Sharma & Coyne, 2015) provides a template of a proven solution by using a set of preferred methods and capabilities. This architecture help practitioners access and use the guidelines, directives and other material that is needed in order to design and create a DevOps platform that accommodates people, processes and technology as describes in Figure 1.

Some similar practices are also described and pointed (Jabbari, bin Ali, Petersen, & Tanveer, 2016). However, the focus of this research is the DevOps Reference architecture detailed on Figure 1, this was the followed architecture to implement the DevOps model that is under the scope of this research.

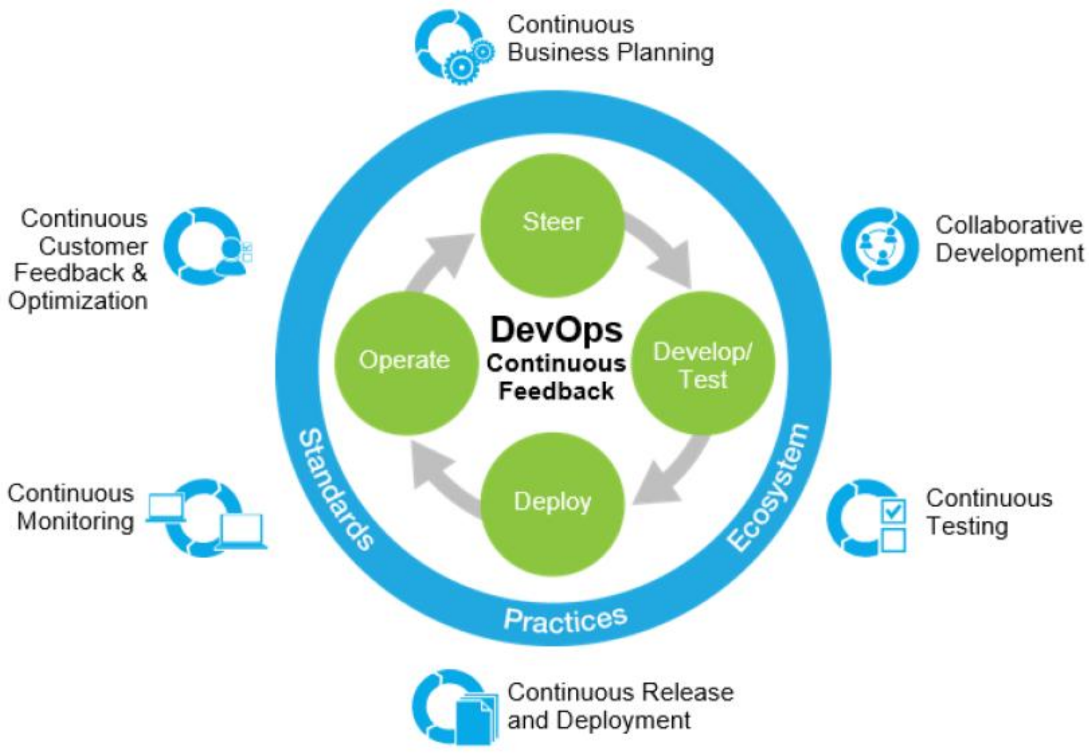

Continuous Business Planning
Continuous Customer Feedback & Optimization
Collaborative Development
Steer
DevOps Continuous Feedback
Operate
Develop/ Test
Deploy
Continuous Monitoring
Continuous Testing
Standards
Ecosystem
Practices
Continuous Release and Deployment

Figure 1 - The DevOps Reference Architecture (Sharma & Coyne, 2015).

The proposal of this reference architecture is to follow a constant workflow of steering, which consists in establishing business needs and adjust them takin in 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 user's feedback in order to get information for the business and if needed change the direction of the business strategy. These 4 elements before mentioned are the core activities of a DevOps platform: Steer, Develop/Test, Deploy and Operate. Completing this architecture and besides the core elements of DevOps we have 6 capabilities that compose the core elements, those capabilities will be explained below (Sharma & Coyne, 2015).

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. Some teams even consider this feedback as an intrusive measure on the planning of new products, when it should be the opposite, it's precisely this feedback that deliver value to the business (Sharma & Coyne, 2015).

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. Collaborative development enables these practitioners to work together by providing a common set of practices and a common platform they can use to create and deliver software. This capability is also identified as continuous integration by other authors; Figure 2 exemplifies how this capability can be seen.

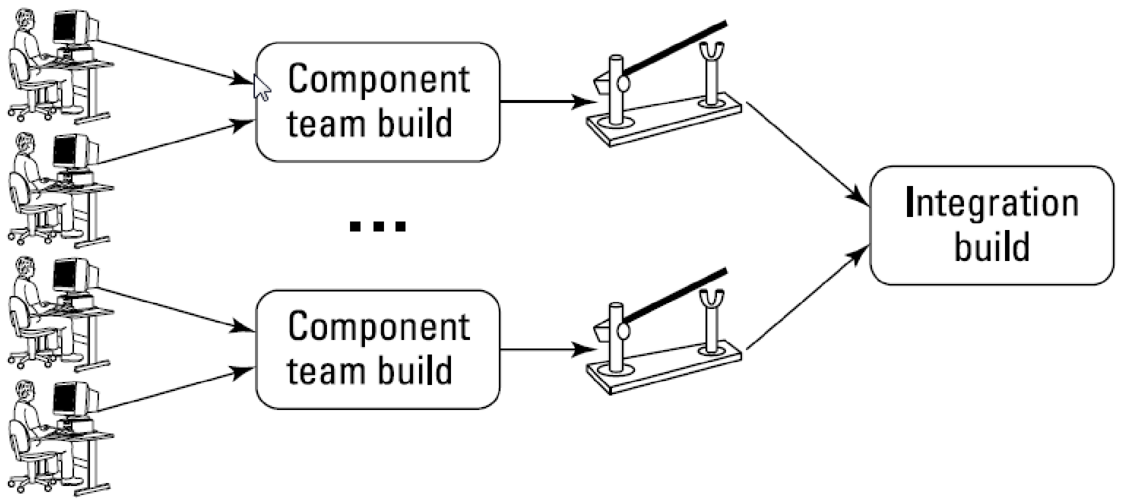


Figure 2 - Continuous integration example (Sharma & Coyne, 2015)

The idea is for developers to regularly integrate their work with the rest of the developers on their team and then test the integrated work. Regular integration of results leads to early discovery and exposure of integration risks. In complex systems, it also exposes known and unknown risks — both technical and schedule-related.

Humble & Farley conclude that the implementation of continuous integration is a change of paradigm to the development team and challenges the team to have more discipline to work on a collaborative way. (Humble & Farley, 2010). Another advantage is that this capability implies that two practices are required to be followed: a good configuration management and an automatized process of software build and testing.

Continuous Testing: Means to test as soon as possible and continuously during the development lifecycle, this leads to a cost and time reduction in costs as well to a better software quality. This practice is viable using techniques like test automation and virtualization to simulate the production environments for the tests to be executed in a scenario more similar to the reality as possible. (Sharma & Coyne, 2015) . According to (Roche, 2013) the growing use of DevOps has introduced the concept of release and tests simulation, automating the entire end to end process and testing several scenarios that can be found by the end user, and invest more effort on these features.

Humble & Molesky have stated that testing activity is a multifunctional activity which involves the entire team and it should be done since the beginning of the project, in an ideal situation the tests should start to be written even before the developers start working on the functionalities that will be tested (Humble & Molesky, 2011). The tests reflect the expected behaviour of the system, this means that when they are successfully conducted, the requested requirements where finished and implemented correctly.

Continuous Release and Deployment: The objective is continuous delivery is to allow that new functionalities are deployed as fast as possible. It was this practice that has 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. Most of the tooling and processes that make the core of DevOps technology exist to facilitate continuous integration, continuous release and continuous deployment. (Sharma & Coyne, 2015).

To Humble & Farley this practice allows that the development team can deliver software in production on a trustful way, predictable and with a low risk. This approach is reached thought feedback loops and cooperation between development team and operations team, in resume the goal is to reduce to a very minimum the time between identify a problem or a new feature, develop the code, test and deploy the solution in production, assuring that the problems are identified as soon as possible when they are easier and less costly to solve. The essential key factor to make the continuous delivery possible is automation, since it allows making the necessary changes between development test and release with the minimum human interaction. (Humble & Farley, 2010).

Humble & Molesky also state that in an advanced level of maturity, continuous delivery allows the teams to delivery software when is needed and with a very low technical risk, deployments can be done on a regular basis which leads to a much more stable work rhythm with less stress and less extra hours, since the IT is faster than the business and not the opposite which is the most common to non DevOps teams. At the end the business risk is low because the decisions are based is functional and trustful software, and this leads to a more integration between IT and the business (Humble & Molesky, 2011).

Continuous Monitoring: The continuous monitoring collects data and metrics that are coming from the different stages of the application lifecycle which allows the reaction of the all the parties involved can react fast to improve or modify the functionalities which are being used (Sharma & Coyne, 2015).

It’s essential to monitor high level business metrics, like revenues or end to end transactions in identified moments in time. From an operational point of view the ideal is define indicators that measure the effects of releases in the system to evaluate the efficiency of the release and deployment process. It’s important that the metric are available for all the involved parties in the software delivery so they can have a clear vision of the team’s evolution as well to identify possible bottlenecks in the process (Humble & Molesky, 2011).

Some more metrics are detailed by Humble & Farley like the % of coverage of automated tests, the number of defects, the daily number of commits as well the daily number of builds and its respective duration and the time for execution of automated tests (Humble & Farley, 2010).

Continuous Customer Feedback and Optimization: The new technologies provide the ability to monitor the customer behavior which allows that the business team or any other interested parties can take the necessary actions to improve the software, are those continuous feedbacks from the customer that allow the operations team to improve the system stability or the development team can improve the software functionalities or even that the business area can improve the products in order to improve customer satisfaction. This continuous feedback loop is an essential component of DevOps allowing business to be more agile and responsive to customer needs (Sharma & Coyne, 2015).

3. Related Work

In Section 2, it was shown that DevOps has not yet been properly studied in the scientific literature, few researches exist about DevOps. There is no specific definition of DevOps or process to be followed, which makes difficult to companies to adopt DevOps since they might not know which practices or process they should implement (Floris, Chintan, & Maya, 2014) (Riungu-Kalliosaari, Makinen, Lwakatare, Tiihonen, & Mannisto, 2016). Since DevOps is a recent framework, there is not much available literature about this subject. Nevertheless, it will be presented in Table 6 some case studies where DevOps was applied.

As one can infer from Table 6, there is no research that studies the end to end implementation of all 6 DevOps capabilities neither the productivity gains while merging teams to a DevOps approach. This will be then the objective of our research to make the holistic and complete study about the implementation and impact of the 6 DevOps capabilities when merging traditional Development and Operations teams.

Table 6 - Generic Information About DevOps Case Studies

CS ID	Title	Reference	Focused Capability
CS.1	Tool Support for Traceability Management of Software Artefacts with DevOps Practices	(Palihawadana, et al., 2017)	N/A
CS.2	Adopting DevOps Practices in Quality Assurance	(Roche, 2013)	Continuous testing
CS.3	End to End Automation On Cloud with Build Pipeline: The case for DevOps in Insurance Industry	(Soni, 2016)	Collaborative development Continuous testing Continuous release and deployment
CS.4	DevOps in Regulated Software Development: Case Medical Devices	(Laukkarinen, Kuusinen, & Mikkonen, 2017)	Continuous release and deployment Collaborative development
CS.5	DevOps for Dummies	(Sharma & Coyne, 2015)	N/A
CS.6	User Stories to User Reality: A DevOps Approach for the Cloud	(Punjabi & Bajaj, 2017)	N/A
CS.7	DevOps for IoT Applications using Cellular	(Karapantelakis, et al., 2016)	N/A
CS.8	Retail DevOps: Rebuilding an Engineering Culture Case	(Stoneham, et al., 2017)	N/A
CS.9	Technology Changes in Government Agencies (A Compilation of Cases): Lessons in Legacy and DevOps Case	(Stoneham, et al., 2017)	N/A
CS.10	Agile Implementation in a Large, Regulated Industry: DevOps and Accelerating Delivery Case	(Stoneham, et al., 2017)	Continuous release and deployment Continuous testing
CS.11	DevOps and Moving to Agile at a Large Consumer Website: Getting Faster Answers at Yahoo Answers Case	(Stoneham, et al., 2017)	Continuous release and deployment Continuous testing

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 (Yin, 2008). These general questions are meant to open up the door for further examination of the phenomenon observed as well as to start a study on a determined phenomenon observed where there isn't prior work (Yin, 2008) (Zainal, 2007). As mentioned in section 2.2, DevOps has not yet been deeply studied among the literature, in this scenario this research will use Case Study research methodology. This type of research is used to study determined phenomenon observed in areas with lack of research (Yin, 2008) as well as questions like "what" are from an exploratory nature since its objective is to develop propositions for future researches. Considering the above scenario, the used research methodology will be an exploratory Case Study which is compatible with the questions that are stated in this research (Zainal, 2007) (Yin, 2008)(Elliot, Sim, & Easterbrook, 2004). Plus, this research is classified as a single case study since the focus is a single team, which is a single unit of analysis as described by (Yin, 2008).

Triangulation means taking different angles towards the studied object and thus providing a broader picture (Runeson & Höst, 2008)(Modell, 2005). 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 author has performed interviews with a subset of team members to collect their analysis from a qualitative point of view. The author also had access to productivity reports and documentation. The methods used in this research for triangulation are:

1. **Team capacity planning analysis**: Analysis of the team capacity planning. This analysis was made using the official reports of annual team capacity planning.
2. **Observation and methodological triangulation**: combining different types of data collection methods. While observing this team it's intended to find how the work was done before, performing an analysis like "Before and After" and to find the difficulties on the adoption of the techniques.

3. **Interviews**: A subset of team members provided is feedback via interviews, the goal is to collect their analysis from a qualitative point of view.

As stated by (Runeson & Höst, 2008) it is important to use several data sources in a case study 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 in a single source.

This case study will be divided in four stages as suggested by Tellis (1997) and Runeson & Höst (2008) and detailed in Table 7. The remaining document is organized according the CS phases presented on the same table.

Table 7. Case Studies Stages adapted from (Tellis, 1997) (Runeson & Höst, 2008)

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 in order 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 author has followed the four validity tests proposed by Yin. Table 8 synthesizes this research validity under Yin's tests. This table shows that this Case Study has been successfully tested against all tests that should be applied to an exploratory Case Study.

Table 8. Validity tests adapted from (Yin, 2008)

Test	**Description**
Construct Validity	Multiple sources of evidences were used. The author has 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 author.
External Validity	The author has analyzed the literature (Section 3) and have not found any research about a complete DevOps implementation neither about productivity impacts when merging to a DevOps approach. Therefore, it proves the novelty of this research and the relevance of our findings for the body of knowledge. This research presents information about a full

	DevOps implementation and respective productivity gains. Therefore, one can consider it as pioneer on the subject.
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.

According to Thomas (Thomas, 2016) a CS should adhere to the following structure presented in Figure 3.

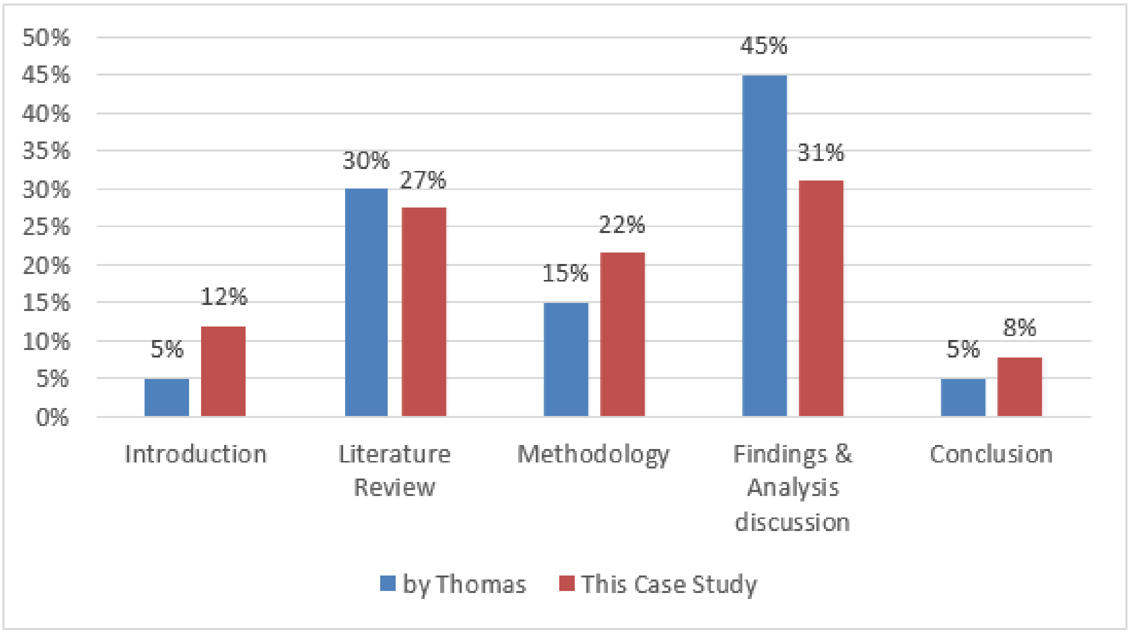


Figure 3 - Comparison between this case study and (Thomas, 2016) proposal

By analysing Figure 3, it is possible to conclude that this CS is following the CS structure proposed by (Thomas, 2016), with small differences between some CS stages.

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, distributed 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 and Development. The detail of each team can be seen in Table 8.

Table 9 - Teams before the DevOps merge

Team	**Team tasks**
Operations	With 4 members, the goal of this team is 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	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).

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 this way a DevOps team with 17 members. After the team's merge to a DevOps approach the implementation of the capabilities presented in Section 2.2 has started immediately. The implementation of this capabilities was divided in two phases named Phase1 and Phase2 and are described in Table 10.

Table 10 - Implemented Capabilities by Phase

Phase	Implemented Capabilities
1	Continuous business planning Collaborative development Continuous monitoring Continuous customer feedback and optimization
2	Continuous testing 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, this had several reason 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 by last technologically 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 (after Phase2 is completed). We called that KPI: Percentage of time spent by the team in each activity type

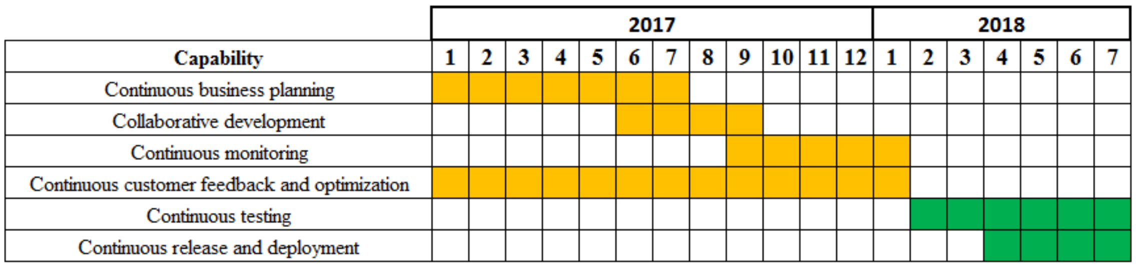
This KPI was defined by the company management as the strategic KPI to be measured and by consequence evaluate the success of this merge. According to the company management a positive result of this KPI is to lower the time spent on the activities: Operational Work and Defect solving and to increase 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 phase of the DevOps implementation detailed in Table 11.

From a timeline perspective Table 11 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 3. 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 11. 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)
2	Continuous release and deployment	Deploy	

The granularity of the timeline for the implementation of each capability is also listed and is presented in figure 4. 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.



	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 4 - Timeline implementation with a monthly granularity per capability

The data used in this research was collected from three 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 will allow us to infer all the hours planned for the team by task and provide input for the KPI calculation.
2. Interviews with 5 senior team members which had hands-on experience during the time of this merging process.
3. Observation on site, the observer has followed the team merge and have had access to the documentation mentioned in points 1 and 2.

6. Conduct the Case Study

In this stage it was gathered, compiled and worked all the data needed to analyze the KPI mentioned in section 5. The interviews conducted with the 5 senior members of the IT team had the purpose to get their own conclusions about this merge, further ahead in section 7 those testimonies will be crossed with the analytical analysis of the KPI evolution over the months.

To collect the team planning capacity, I 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 manager of the team regarding year 2017 and 2018 (until August) in excel format 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 it was asked the open questions presented in Table 12.

Table 12. 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 after implementation of Phase1
Q4	Comment the percentage of time spent on each activity after implementation of Phase1 and mention for each activity which capabilities you consider that have contributed the most.
Q5	What is your general opinion after implementation of Phase2
Q6	Comment the percentage of time spent on each activity after implementation of Phase2 and mention for each activity which capabilities you consider that have contributed the most.

Regarding the method for the interviews: 3 Interviews were done on-site and the other 2 were done via Skype. Table 13 details the age, years of experience and area (Development or Operations) of each interviewed.

Table 13 – Interviewees' details

Interviewee	**Age**	**Years of Experience**	**Area**	**Interview Duration (min)**
A	*39*	*14*	*Operations*	*38*
B	*37*	*12*	*Development/Architecture*	*32*
C	*29*	*5*	*Development*	*25*
D	*30*	*6*	*Development*	*28*
E	*36*	*10*	*Operations*	*33*
Average	*34,2*	*9,4*	*N/A*	*31,2*
Total	*171*	*47*	*N/A*	*156*

7. Analysis of Collected Data

On this stage it is supposed to analyze the data that was collected on the previous stage. The data that was collected from the team resource management documents and from the interviews with the team members. The analysis of the data will be divided in three different times of the implementation timeline:

- **Baseline Status**: Status about the time spent for each activity type before the merge.
- **Status after Phase1**: Status about the time spent for each activity type after the 4 capabilities are implemented.
- **Status after Phase2**: Status about the time spent for each activity type after the 6 capabilities are implemented.

The three next sub chapters will present the detailed analysis of the data in the above mentioned 3 different moments.

7.1. Baseline Status

Based on the interviews, before the teams were merged, operational work consumed most part of the time and important activities like: Architecture, Business Analyst and Test Management were not being performed by the team due to lack of capacity. This is based in the answers of the team members to question Q1 presented in Table 12: "*Due to the high amount of time invested in operations it was not possible to put more effort in activities considered crucial to increase the team service quality as well as evolve the team to more valuable activities then 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 must 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 14 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 presented in Table 12.

Table 14 – Interviewee's answers before implementation

Activity	**%**	**Team members Comments**
Meetings	*12%*	*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.*

7.2. Status after Phase 1

After Phase1 was implemented the author compared the differences of percentage of time spent per activity type in the baseline situation and after phase1, to have a more graphical overview about this situation a report was created and is showed on Figure 5.

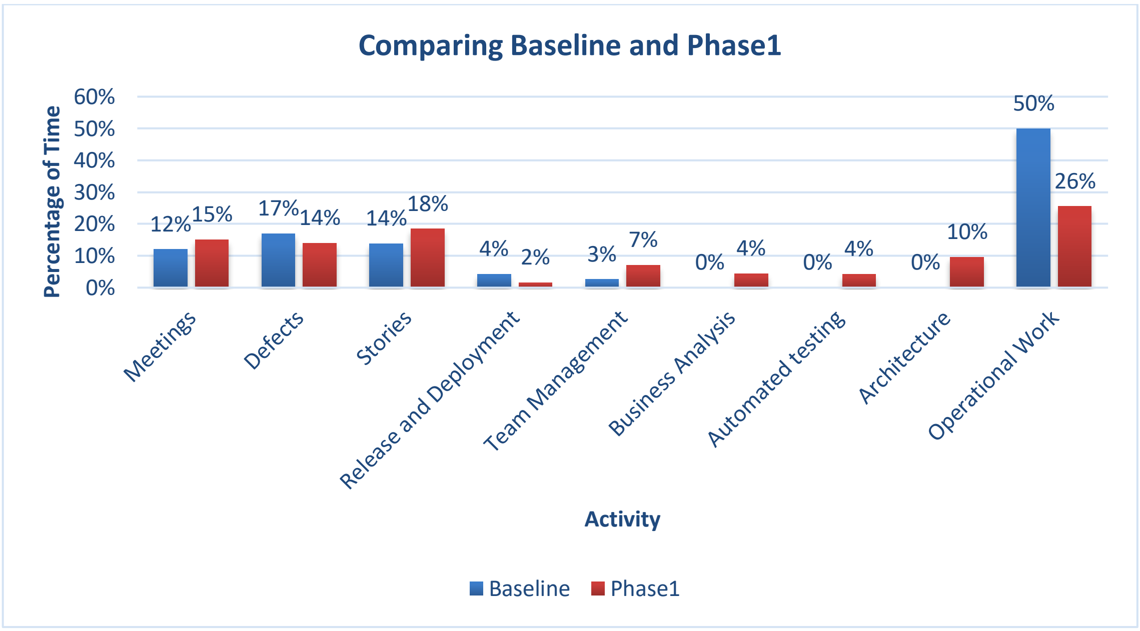


Figure 5 - Comparing Baseline and Phase1

After the implementation of Phase1 the operational work decreased from 50% of the overall time of the team to 26% due to the automation of manual tasks that was performed by the developers. This reduction was identified as the major success of this initiative and there was the recognition of the management for the good work developed, also the team felt more motivated and valued. The time saved on operational work gave the team the possibility to start doing another kind of activities, more valuable, 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.

A set of procedures based on the DevOps capabilities were implemented during phase 1 to implement these same capabilities. These procedures were the key to decrease the operational work the defects and the time spent doing the releases and to improve the number of developed stories. These procedures are described below in detail:

Daily stand up meetings were organized between the Product Owner and the DevOps team, this 15 minutes meetings had the purpose to align requirements and show the already implemented functionalities, this procedure helped decreasing the gap between the PO requirement and the DevOps team understanding, it was a key factor to decrease the defects, because in most part of the cases, defects were not really defects but yes deficient communication between PO and DevOps team

Around 10% of the development time was shifted to internal activities, the idea was to automate manual work, this was a crucial factor to decrease the time spent on releases, because almost all the manual work done by operational colleagues was automated.

A set of monitoring dashboards were created to detect in real time abnormalities in the productive system, this allows to react immediately when problems occurred which lead also to a decrease on the number of incidents because the resolution speed was higher than in the past. 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. It was also mentioned by the developers that the team spends more time then desirable in meetings, however they understand this fact as necessary due to the lack of experience of the team in a DevOps approach.

These improvements and negative consequences were highlighted by the team members in the answers to Q3 presented in Table 12. Those answers are compiled in Table 15.

To show a detailed analysis a delta between baseline and Phase1 is presented in Table 16, this table shows the percentage of time spent by the team for each activity type, in the baseline status and after Phase1 as well the most relevant comments of the team members. This table contains the answers to Q4 presented in Table 12.

Table 15 - Interviewee's answers after Phase1

Activity	%	Δ% to Baseline	Team members Comments
Meetings	*15%*	*+3%*	*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.*

Table 16 - Answers to Q3 by the team members

Question	**Interviewee**	**Comment**	
Q3	*A*	*After phase1 I have noticed a good improvement in the interaction among team members, because the management has established common goals the team works on a more jointly mindset and not in silos. Also, the story specification has improved, and we get much more help from the developers when an issue occurs in production.* *Still a lot to improve, the team seems inexperienced in this new mindset and sometimes roles and responsibilities are mixed.*	☺ ☹
	B	*Some positive achievements were made after phase1, especially in the mindset point of view, as one team we are now more focused on delivering the end 2 end value then only new features. The developers have automatized some manual tasks, which has released some effort to the operations colleagues. The interaction with the customer is also easier because we have more feedback loops. Even if we don't have in my opinion the two most important DevOps capabilities: Continuous Delivery and Continuous Testing we have made an interesting progress.*	☺
	C	*The synergies between the members of the team have improved, now as one single team and common goals it's obvious that we all have the same vision and objective. It was a promising first phase.*	☺
	D	*We could notice some changes after Phase1. Anyway, it's my expectation that the great benefit to the developers will be the implementation of automated testing and that will only come in Phase2.* *Some less positive things happened like the time that we took to implement 4 capabilities: 12 months. The time spent in meetings due to the lack of experience of them team in the DevOps model, however I can understand the need and it will improve when the team maturity also improves*	☺ ☹
	E	*The implementation of the 4 capabilities in Phase1 helped the operations, the developers have automated a huge number of manual tasks, which has reduced the error rate in releases and also daily tasks that were manually executed.* *Still room to improve, some team members are still having difficulties in adapting to this way of working.*	☺ ☹

7.3. Status after Phase 2

After Phase2 was implemented the author have compared the differences of percentage of time spent per activity type in three distinct occasions: baseline situation, after Phase1 and after Phase. To have a more graphical overview about this situation a report was created and is showed on Figure 6.

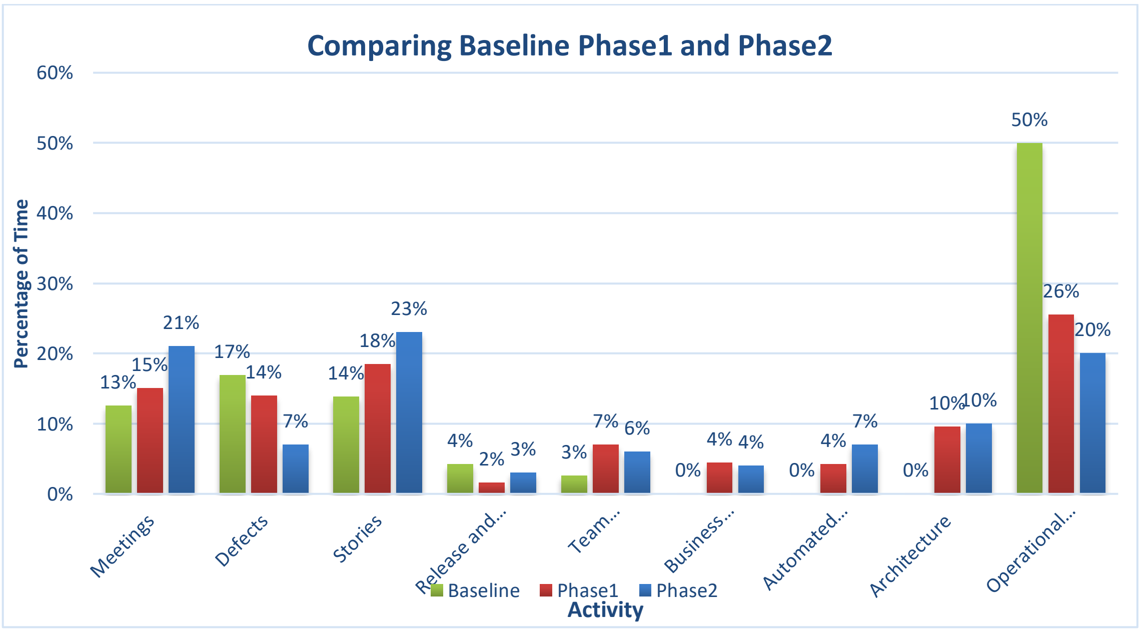


Figure 6 - Comparing Baseline, Phase1 and Phase2

After the implementation of Phase1 some interesting aspects have been observed. A decrease on defects (-10%) and increase on development capacity (+8% in stories) and this represents a very positive trend in shifting capacity from defect solving to development of new features, which increases delivery quality, increased customer satisfaction with a product with less bugs and more features. This has also positive impact in the team engagement because is always more desirable to create new features then correct bugs.

Another interesting result was the increase of management activities (+3%) and an increase in meetings (+10%), 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 a near future when the team is more mature on the DevOps model is expected a decrease on this value.

A set of procedures based on the DevOps capabilities were implemented during phase 2 to implement these same capabilities. These procedures were the key to decrease even more the operational work, the defects and the time spent in releases and to improve the number of developed stories. These procedures are described below in detail:

Change the release cycle to have weekly deployments instead of a 3 months cycle. This change had a strong impact in the time to market of new features and fixes, this led to the decrease of operational work because as soon a defect was detected it was fixed and deployed on a very short time frame, which reduced the incidents related with that defect.

An automated testing tool was bought and implemented. A rule in the process was implemented: The Product Owners when describing a story he should also describe the test case that is an acceptance criteria to that story, the DevOps team then configured the test automation tool with the described test case and when the implementation was done we run that test. This procedure did reduce considerably the number of defects because it prevented that incorrect code was deployed in prod, discovering errors upfront in staging environments.

These facts were noticed by the team members according to the answers to Q5 presented in Table 12. These answers are compiled in Table 17.

To provide a detailed analysis a delta between Baseline and Phase1 is presented in Table 17, this table shows the percentage of time spent by the team for each activity type, in the baseline status and after Phase2 as well the most relevant comments of the team members. This table contains the answers to Q6 presented in Table 12.

Table 17 - Answers to Q5 by the team members

Question	**Interviewee**	**Comment**	
Q5	*A*	*After Phase2 we have now much shorter release cycles and automated testing, these 2 capabilities have, in my opinion, improved a lot the team delivery quality. The defects have significantly decrease due to the automated testing and the customer is happier because we are now much faster in deploying in prod the demanded features by the business.* *Communication has improved however some team members have not yet adapted to this way of working, mindset is not totally aligned. We are getting better working as a team but still we can spot some immaturity*	☺ ☹
	B	*Phase 2 brought a very important capability for developers: Continuous Testing. With this capability the developers could test in a much broader extension the developments before making them available for the PO's. Also, the fact that we could deliver much faster (continuous delivery) has a positive effect on business side, we are no longer a delay. One thing that has improved is the people motivation, recognition for the good work developed and the fact that now more people have challenging and complex tasks (Architect, Business Analyst, etc.). It was possible to notice an increase on the meetings among team members, however I believe that this has a trend to decrease when the team gains more maturity*	☺
	C	*Phase2 brought something very desired by the developers, automated testing, this was a key factor to the decrease of defects and gaining speed on deployment. Also, we started to develop more and having less defects.* *The communication with PO's has slightly improved but is my feeling that they have severe difficulties adapting to this new way of working*	☺ ☹
	D	*Very positive outcome of Phase2, the automated testing was a real improvement on the way we work, defects were decreased and it's very useful. I can no longer imagine myself working in the old way. We took 18 months doing this shift but worth the time.*	☺
	E	*The main benefit of Phase2 was seen into the development process, with the implementation of automated testing the developers can deliver with high quality. From my point of view Phase1 was much better for the operations than development because it decreased a lot the operational work and Phase2 was better for the developers who saw the defects decrease and their development time increase.*	☺

Table 18 - Interviewee's answers after Phase2

Activity	**%**	**Δ% to Baseline**	**Team members Comments**
Meetings	*21%*	*+9%*	*The team spends a considerable amount of time communicating. Some elements still have doubts regarding their role. This is a necessary time to be spent however some people that already are align with the DevOps mindset are complaining that they are spending time in meeting which prejudices the work itself. This should be a point to improve in the team.*
Defects	*7%*	*-10%*	*This number was a great achievement. The number of defects is now reasonable, and this fact improves customer satisfaction and team motivation.*
Stories	*22%*	*+8%*	*The developers are now developing more features then solving defects. This is a very positive outcome for the team and increases the motivational level of developers.*
Release and Deployment	*3%*	*-1%*	*The decrease of time spent in the release deployment activities is interesting, even if we do more releases per year we spent less time with the releases. This is mainly due to the automation of manual tasks that were performed by the developers.*
Team Management	*6%*	*+3%*	*The need for steering is still high due to some inexperience of the team in the DevOps framework. There was the need to improve communication with all team members that was now only one joint team and that is the work of the manager to facilitate that communication.*
Business Analysis	*4%*	*+4%*	*Positive number. Capacity of the team was release from the operational work and that allowed the team to shift capacity to start performing business analysis.*
Automated testing	*7%*	*+7%*	*The fact that automated testing is in place is very positive for the entire team, this prevents a high number of defects being deployed in prod due to lack of proper testing, this has contributed to the decrease of operational work, specially incidents.*
Architecture	*10%*	*+10%*	*Positive achievement. The team provides now architecture services, this is seen by the management as a positive example of value added to the team.*
Operational Work	*20%*	*-30%*	*This number has highly impacted the personnel from operations. The workload is now much more reduced, and the tasks are more challenging. Manual work is very reduced, and the team members are more motivated, and their performance is getting better. From my point of view this was the major achievement in this DevOps marge, to reduce the operational work in more than half.*

8. Develop Conclusions

This research focusses a problem which is the lack of synergies between two teams that were providing support and development to the same application (in this case: Service Now). To solve this lack of alignment and to optimize the teams from a financial point of view, the two teams were merged into a single DevOps team. In this context and to develop this work, a research question was defined: *What are the main productivity gains when merging Development and Operations teams into a DevOps team*.

Since this research is exploratory in its nature, a Case Study methodology was used which is indicated when there is not so many information and previous researches about the topic. To perform this research, triangulation of sources of information was used: Team capacity planning (documentation), observation and interviews (5 senior team members).

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 traditional IT is recognized to do so.

This study adds to the body of knowledge concrete data extracted in a real-world scenario where it was observed the merge of two teams into a single DevOps team. The demonstrated impact of DevOps adoption and productivity improvements like the differences of % of time spent per activity type which can be a base for future researches about the productivity of merging two teams into a single DevOps team.

For the practitioners this study is an example of how to apply DevOps showing qualitative data which can be useful for future implementations. For the academic context is an investigation which brings information that did not existed in the body of knowledge and it’s a baseline for future investigations in this area.

For a better understanding of the outcome of this research, we have aggregated the conclusions in Table 19 which shows the impact of the DevOps transformation on each activity type.

Table 19 - DevOps impact in each activity type

Activity	Impact of DevOps transformation
Meetings	Meetings have increased. This increase was mainly due to two reasons: more need of alignment with the PO's and also more need of internal meetings due to lack of maturity of the team in a DevOps approach.
Defects	The defects have significantly decreased. The reason for this comes from a better alignment with the PO's but the biggest contribute comes from the automated testing tool which prevents a significant number of defects to be deployed in PROD.
Stories	The capacity for stories has increased, for this what contributed the most was the decrease on Defects, that time was totally shifted to stories.
Release and Deployment	At this moment we do more releases per year and we spent less time with the releases. This fact occurred due to the automation of manual.
Team Management	The management effort has increased. For this the factor that contributed the most is the lack of maturity in the team in a DevOps approach.
Business Analysis	The fact that we saved time in other areas, especially operational work, the team was able to start providing this role adding more value to the team and more challenging tasks
Automated testing	The fact that we saved time in other areas, especially operational work, the team was able to start providing this role adding more value to the team and more challenging tasks. This task was also crucial to implement automated testing in Phase2
Architecture	The fact that we saved time in other areas, especially operational work, the team was able to start providing this role adding more value to the team and more challenging tasks
Operational Work	The decrease on operational work was very positive. The team is more motivated and also freer to do other kind of interesting activities. Some stakeholders in the management side consider this the main achievement of this initiative

Regarding the limitations of this study we can point the fact that this research only focuses one team in a specific environment and we cannot infer this result to other teams. Also, the fact that we could not study in separate the impact of the implementation of each capability can be considered as a limitation.

Possible future work for this research can go into the direction of understanding why some percentages had a behaviour not so straightforward as expected, for example release and deployment has decreased on Phase1 and increased on Phase2. In this study due to the way that the company has made the merge it was not possible to study the impact of each single capability into the team planning, this study could also be a direction to follow.

Bibliography

Bass, L., Weber, I., & Zhu, L. (2015). *DevOps - A Software Architect's Perspective.* Addison-Wesley.

Callanan, M. S. (2016). DevOps: Making it easy to do the right thing. 53-59.

Cukier, D. (2013). DevOps patterns to scale web applications using cloud services. *Proc. of the 2013 Companion Publication for Conference on Systems, Programming, & Applications: Software for Humanity*, (pp. 143–152). NY.

Elliot, P. D., Sim, S. E., & Easterbrook, S. M. (2004). Case studies for software engineers. (pp. 736-738). Proceedings. 26th International Conference on Software Engineering.

Elliot, S. (2014). *DevOps and the Cost of Downtime: Fortune 1000 Best Practice Metrics Quantified.* International Data Corporation (IDC).

Floris, E., Chintan, A., & Maya, D. (2014). *Report: DevOps Literature Review.* Twente: University of Twente.

Humble, J., & Farley, D. (2010). Em *Continuous Delivery: Reliable Software Releases through Build, Test and Deployment Automation.* Addison-Wesley.

Humble, J., & Molesky, J. (2011). Why Enterprises Must Adopt Devops to Enable Continuous Delivery. *Cutter IT Journal*.

Hussaini, S. W. (2015). A Systemic Approach to Re-inforce Development and Operations. *Complex Adaptive Systems.* San Jose, California.

Hüttermann, M. (2012). *DevOps for Developers.* NY: Apress.

Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016). What is DevOps? A Systematic Mapping Study on Definitions and Practices. *Proceedings of the Scientific Workshop Proceedings of XP2016 on - XP '16 Workshops*, 1-11.

Karapantelakis, A., Liang, H., Wang, K., Vandikas, K., Inam, R., Fersman, E., . . . Giannokostas, V. (2016). DevOps for IoT applications using cellular networks and cloud. *Proceedings - 2016 IEEE 4th International Conference on Future Internet of Things and Cloud, FiCloud 2016*, 340-347.

Kim, G., Behr, K., & Spafford, G. (2013). *The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business win.* IT Revolution Press.

Langerman, J. J., & Kamuto, M. B. (2017). Factors Inhibiting the Adoption of DevOps in Large Organisations: South African Context. *2nd IEEE International Conference On Recent Trends In Electronics Information & Communication Technology*.

Laukkarinen, T., Kuusinen, K., & Mikkonen, T. (2017). DevOps in Regulated Software Development: Case Medical Devices. *2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER)*, 15-18.

Lwakatare, L., Karvonen, T., Sauvola, T., Kuvaja, P., Holmström Olsson, H., Bosch, J., & Oivo, M. (2016). Towards DevOps in the Embedded Systems Domain: Why is It so Hard? *49th Hawaii International Conference on System Sciences.*

Matej Arta, T. B. (2017). DevOps: Introducing Infrastructure-as-Code. *2017 IEEE/ACM 39th IEEE International Conference on Software Engineering Companion.*

Modell, S. (2005). Triangulation between Case Study and Survey Methods in Management Accounting Research: An Assessment of Validity Implications. *Management Accounting Research*, 231-254.

Neely, S. S. (2013). Continuous delivery? Easy! Just change everything (well, maybe it is not that easy). *Proceedings of the 2013 Agile Conference.*, (pp. 121-128). Washington, DC.

Palihawadana, S., Wijeweera, C. C., Sanjitha, M. G., Liyanage, V. K., Perera, I., & Meedeniya, D. A. (2017). Tool support for traceability management of software artefacts with DevOps practices. *3rd International Moratuwa Engineering Research Conference, MERCon 2017*, 129-134.

Punjabi, R., & Bajaj, R. (2017). User stories to user reality: A devops approach for the cloud. *2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings*, 658-662.

Riungu-Kalliosaari, L., Makinen, S., Lwakatare, L. E., Tiihonen, J., & Mannisto, T. (2016). DevOps Adoption Benefits and Challenges in Practice: A Case Study. *Product-Focused Software Process Improvement: 17th International Conference* (pp. 590-597). Trondheim, Norway: Springer International Publishing.

Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare, L. E., Tiihonen, J., & Männistö, T. (2016). DevOps Adoption Benefits and Challenges in Practice: A Case Study. *Product-Focused Software Process Improvement: 17th International Conference* (pp. 590-597). Trondheim, Norway: Springer International Publishing.

Roche, J. (2013). *Adopting DevOps practices in quality assurance.* New York: Magazine Communications of the ACM.

Runeson, P., & Höst, M. (2008). Guidelines for conducting and reporting case study.

Sharma, S., & Coyne, B. (2015). DevOps For Dummies®, 2nd IBM Limited Edition. John Wiley & Sons, Inc.

Soni, M. (2016). End to End Automation on Cloud with Build Pipeline: The Case for DevOps in Insurance Industry, Continuous Integration, Continuous Testing, and Continuous Delivery. *Proceedings - 2015 IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2015*, 85-89.

Stoneham, J., Thrasher, P., Potts, T., Mickman, H., DeArdo, C., & A. Limoncelli, T. (2017). *DevOps Case Studies - The Journey to Positive Business Outcomes.* IT Revolution.

Tellis, W. (1997). Information technology in a university: a case study. *Campus-Wide Information Systems*, 78-91.

Tessem, B. I. (2008). Cooperation between developers and operations in software engineering projects. *Proceedings of the 2008 Int. Workshop on Cooperative and Human Aspects of Software Engineering*, (pp. 105-108). NY.

Thomas, G. (2016). *How to Do Your Case Study* . Sage Publications.

Yin, R. (2008). *Case Study Research: Design and Methods (Applied Social Research Methods).* SAGE Publications.

Zainal, Z. (2007). Case study as a research method. *Journal Kemanusiaan*.

Appendices

Appendix A - Questionnaire about the inside view of a DevOps transition

Interview 1

Personnel and professional questions

Age: 39
Years of IT Experience: 14

Area:
Operations: X
Development:

1. Open Questions about the team merge

Q1: What is your general opinion about the baseline situation (before the DevOps merge)

In the baseline situation we clearly notice a lack of communication between development and operations. Several problems were identified in this relation, the operations team had a much higher workload then the development team and lack solidarity among both teams, they were both focused on each one's silos. Most part of the project budget was spent just keeping the lights on (Operations) what was frustrating scenario not only for the management but also for the team that did not had very challenging tasks.

Q2: Comment in the below table the percentage of time spent on each activity in the baseline situation

Activity	%	Comments

Meetings	12%	*Expected*
Defects	17%	*The high number of defects and time that we spent solving them is negative for the team motivation. And 17% of the time solving defects is wasted time that could be applied in new features instead of bug fixing.*
Stories	14%	*The team at this moment delivers less hours of development then expected, maybe because it too much work in other less valuable activities that is consuming time for innovation and new features.*
Release and Deployment	4%	*Expected*
Team Management	3%	*Expected, maybe a little less then desired.*
Business Analysis	0%	*Very important role that is not being done by the team due to lack of capacity*
Automated testing	0%	*Very important role that is not being done by the team due to lack of capacity*
Architecture	0%	*Very important role that is not being done by the team due to lack of capacity*
Operational Work	50%	*In my opinion the biggest problem in the team, the repetitive manual tasks and unnecessary problems that are caused by the heavy processes. Also operational work is demotivating to the team and from a financial point of view is negative*

Q3: What is your general opinion after implementation of Phase1

After phase1 I have noticed a good improvement in the interaction among team members, because the management has established common goals the team works on a more jointly mindset and not in silos. Also, the story specification has improved, and we get much more help from the developers when an issue occurs in production. Still a lot to improve, the team seems inexperienced in this new mindset and sometimes roles and responsibilities are mixed.

Q4: Comment the percentage of time spent on each activity after implementation of Phase1, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	15%	+3%	*With more people in the team more meeting were necessary to clarify how things works. From my point of view the 2% rise is still a low number for the big change that we faced, I was excepting a higher rise on this indicator.*
Defects	14%	-3%	*This was a good result, the team could notice some decrease in the number of defects which is always a good scenario, however we still have room to improve in here, 14% is still a high number.*
Stories	18%	+4%	*I'm not so close to the development side, but this is positive, the team has now more capacity to develop new features.*
Release and Deployment	2%	-2%	*The developers have automatized a lot of manual tasks that were being done during the release, that's why we could decrease the release time and reduce the errors as well because with manual tasks there is always the human error involved.*

Team Management	7% +4%	*Expected, the management had to spend some time coaching the team for this change and it was not an easy task to make people change their mindset*
Business Analysis	4% +4%	*The fact that we were able to start doing business analysis was very positive to the team, we can do more with the same amount of people and those tasks are more interesting then manual operational work. However, I was expecting that we could allocate more time to this task, but maybe with Phase2 that can be a reality*
Automated testing	4% +4%	*This helped a lot my area (Operations). Development is finally properly tested which avoids a considerable number of incidents in the days after releases.*
Architecture	10% +10%	*This was definitely a needed activity, the architects start to control the more strange ideas of the Product Owners which lead to a more controlled development of new features. However, I see that the architects sometimes are being too strict and are blocking some new features that in my point of view make sense.*
Operational Work	26% -24%	*This has impacted directly my area of work and it's really an improvement. A considerable amount of work was automated by the developers, this caused a great decrease of the workload in the operational side and raised motivation, because the colleagues started to do more challenging tasks.*

Q5: What is your general opinion after implementation of Phase2

After Phase2 we have now much shorter release cycles and automated testing, these 2 capabilities have, in my opinion, improved a lot the team delivery quality. The defects have significantly decrease due to the automated testing and the customer is happier because we are now much faster in deploying in prod the demanded features by the business. Communication has improved however some team members have not yet adapted to this way of working, mindset is not totally aligned. We are getting better working as a team but still we can spot some immaturity

Q6: Comment the percentage of time spent on each activity after implementation of Phase2, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	21%	+9%	*The team spends a considerable amount of time communicating, some elements still have a lot of doubts regarding this new DevOps approach, some still don't know clearly their responsibilities. I believe this is a necessary time to be spent however some people that already are align with the DevOps mindset are complaining that they are spending time in meeting which prejudices the work itself. This should be a point to improve in the team.*
Defects	7%	-10%	*This number was a great achievement, finally we have a reasonable number of Defects and this fact improves customer satisfaction and team motivation.*
Stories	22%	+8%	*The developers are now developing more features then solving defects, even if it's not my area I can understand that this is a very positive outcome for the team and motivating for the developers.*
Release and Deployment	3%	-1%	*The decrease on the release deployment is interesting, even if we do more releases per year we spent less time with the*

			releases. This is mainly due to the automation of manual tasks that were performed during a release.
Team Management	6%	+3%	*The need for steering and organization of the team increased because of some inexperience of the team in DevOps. Coaching was needed by the team manager. I also felt that there was the need to improve communication with all team members that was now only one joint team and that is the work of the manager to facilitate that communication.*
Business Analysis	4%	+4%	*Positive number. Capacity of the team was release from the operational work and that allowed the team to shift capacity to start performing business analysis.*
Automated testing	7%	+7%	*The fact that automated testing is in place is very positive for the entire team, this prevents a high number of defects being deployed in prod due to lack of proper testing, this has contributed to the decrease of operational work, specially incidents.*
Architecture	10%	+10%	*Positive result. The team provides now architecture services which was not possible in the past, this is seen by the management as a positive example of value added.*
Operational Work	20%	-30%	*This number has highly impacted the personnel from operations. The workload is now much more reduced, and the tasks are more challenging. Manual work is very reduced, and the team members are more motivated and their performance is getting better. From my point of view this was the major achievement in this DevOps marge, to reduce the operational work in more then half.*

Interview 2

Personnel and professional questions

Age: 37
Years of IT Experience: 12

Area:
Operations:
Development: X

1. Open Questions about the team merge

Q1: What is your general opinion about the baseline situation (before the DevOps merge)

In the baseline situation we could frequently assist to a clash between developers and operations. The fact that the dev team works in a Scrum approach makes us much faster than the operations team which still works in a traditional way. It's also complicated to interact with customers from another cultural environment which have a very conservative ways of seeing software development

Q2: Comment in the below table the percentage of time spent on each activity in the baseline situation

Activity	%	Comments
Meetings	12%	*In my opinion we spent too much time in meeting, this is manly due to the lack of capacity of the product owners to define what they needed, this means that in the end the developers have to make part of their work.*
Defects	17%	*High number of defects. Most part of the mentioned defects are spec changes, however to make into production they are wrongly reported*

		as defects. It’s crucial to reduce this number, at this moment we use more time to work on defect then on new stories
Stories	14%	*Reduced number, we as a development team cannot have only 14% of the time developing, the value that we add to the project developing so less is not good. It’s very important to improve this number*
Release and Deployment	4%	*I’m not so aware of this number because this task is done by the ops team.*
Team Management	3%	*Expected result.*
Business Analysis	0%	*Unfortunately, not done by team*
Automated testing	0%	*Unfortunately, not done by team*
Architecture	0%	*Unfortunately, not done by team*
Operational Work	50%	*This is a very high number to deal with. Sometimes we have developers solving tricky incidents due to the lack of knowledge and documentation of the Operations team, Ops and Devs should cooperate more in here to reduce this number*

Q3: What is your general opinion after implementation of Phase1

Some positive achievements were made after phase1, specially in the mindset point of view, as one team we are now more focused on delivering the end 2 end value then only new features. The developers have automatized some manual tasks, which has released some effort to the operations colleagues. The interaction with the customer is also easier because we have more feedback loops. Even if we don’t have in my opinion the two most important DevOps capabilities: Continuous Delivery and Continuous Testing we have made an interesting progress.

Q4: Comment the percentage of time spent on each activity after implementation of Phase1, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	15%	+3%	*It was not so positive to increase the time with meeting, however is obvious that while doing the change to a new way of working the communication is key and for that some extra meetings had to be conducted. Could be a point to improve reduce the number of meetings when we are more mature in this way of work.*
Defects	14%	-3%	*Some improvement could be noticed in this area, is positive however we still have room to improve. I have the perceptions that after phase2 when continuous testing is in place this number will be more reduced.*
Stories	18%	+4%	*Interesting result, we can notice that we already spent more time developing, in my opinion this is a direct consequence of having less defects to solve.*
Release and Deployment	2%	-2%	*Good result, a lot of manual tasks were automated which decreased the release time.*
Team Management	7%	+4%	*The coaching to a DevOps approach was mandatory due to the lack of experience of the team in this way of working. This coaching had to be done by the team manager. This result makes sense.*

Business Analysis	4%	+4%	*Positive result, we as a team are now able to provide this role to the customer, this lead to some savings because we could dismiss the external consultant who did this role, and improved team motivation because the tasks are more interesting*
Automated testing	4%	+4%	*Positive result, we as a team are now able to provide this role to the customer, this lead to some savings because we could dismiss the external consultant who did this role, and improved team motivation because the tasks are more interesting*
Architecture	10%	+10%	*Positive result, we as a team are now able to provide this role to the customer, this lead to some savings because we could dismiss the external consultant who did this role, and improved team motivation because the tasks are more interesting*
Operational Work	26%	-24%	*This was in my opinion the biggest achievement, the quantity of manual tasks that were automated contributed to this result and the decrease on incidents and email exchange. Very good indicator.*

Q5: What is your general opinion after implementation of Phase2

Phase 2 brought a very important capability for developers: Continuous Testing. With this capability the developers could test in a much broader extension the developments before making them available for the PO's. Also, the fact that we could deliver much faster (continuous delivery) has a positive effect on business side, we are no longer a delay. One thing that has improved is the people motivation, recognition for the good work developed and the fact that now more people have challenging and complex tasks (Architect, Business Analyst, etc). It was possible to notice an increase on the meetings among team members, however I believe that this has a trend to decrease when the team gains more maturity

Q6: Comment the percentage of time spent on each activity after implementation of Phase2 and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**∆% to Baseline**	**Comments**
Meetings	21%	+9%	*This result is interesting, the internal meetings among the team have increased the purpose is to align the way we work in this new approach. I don't have the evidences, but I think that the time spent in meetings with customers have been maintained, as a Developer I don't like to have such time spent in meetings, however I understand the need and I'm expecting a decrease of this metric when we have more maturity in a DevOps model.*
Defects	7%	-10%	*Very positive result, the automation tool for continuous testing has increased the detection rate of defects before they are released to prod*
Stories	22%	+8%	*Positive result, we are able to produce more features, and this is noticed by the customer, the satisfaction increases. This is also positive for the developers who spent more time doing what they enjoy.*
Release and Deployment	3%	-1%	*The time spent in releases is almost the same, with a small decrease, however we release more times than in the past, so it might not seem, but the number is very positive, because with less time we release more frequently, the productivity on this topic has increased significantly*

Team Management	6%	+3%	*The work in managing the team has increased due to the need of coaching into a DevOps approach, this number will probably decrease when the team is more mature and stable.*
Business Analysis	4%	+4%	*No changes from Phase1.*
Automated testing	7%	+7%	*Increased from Phase1 which is normal because we have invested in automate the testing.*
Architecture	10%	+10%	*No changes from Phase1*
Operational Work	20%	-30%	*This was in my opinion the biggest achievement, the quantity of manual tasks that were automated contributed to this result and the decrease on incidents and email exchange. Very good indicator. This result even got better from phase1 to phase2, this was in my opinion our best achievement of our DevOps merge.*

Interview 3

Personnel and professional questions

Age: 29
Years of IT Experience: 5

Area:
Operations:
Development: X

2. Open Questions about the team merge

Q1: What is your general opinion about the baseline situation (before the DevOps merge)

The sprit in the Dev team is very positive however I can feel some attrition with the Operations team, sometimes they come to us asking for help but without any pre analysis done earlier. The synergies are not the best. Also, the developers lose a lot of time in operational tasks, this situation is demotivating because developers typically are not keen in doing operational tasks. I also feel a cultural bump between PO and Developers, it’s difficult to work with conservative mindsets.

Q2: Comment in the below table the percentage of time spent on each activity in the baseline situation

Activity	%	Comments
Meetings	12%	*Too many times spent in meetings specially with PO’s. As a developer I don’t feel comfortable with so many meetings, mainly because I consider them not so productive.*
Defects	17%	*The system is not so stable, so we have some defects, something that we need to improve.*

Stories	14%	*We as developers would like to spend more time developing, 14% is a low number, we expect improvements with the DevOps merge.*
Release and Deployment	4%	*As a developer I don’t have a vision on this number.*
Team Management	3%	*Seems a reasonable number but I don’t have such a detailed vision on this score.*
Business Analysis	0%	*Task that we cannot perform due to lack of time, which is not positive because the task is interesting, and it would add value to the team to be able to perform this task.*
Automated testing	0%	*Task that we cannot perform due to lack of time, which is not positive because the task is interesting, and it would add value to the team to be able to perform this task.*
Architecture	0%	*Task that we cannot perform due to lack of time, which is not positive because the task is interesting, and it would add value to the team to be able to perform this task.*
Operational Work	50%	*This is really a burden, a lot of operational work is done by the developers and the operational personnel is heavily loaded with this boring manual tasks.*

Q3: What is your general opinion after implementation of Phase1

The synergies between the members of the team have improved, now as one single team and common goals it’s obvious that we all have the same vision and objective. It was a promising first phase.

Q4: Comment the percentage of time spent on each activity after implementation of Phase1, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	15%	+3%	*The increase of time was related to internal meetings to align in this new way of working. They were necessary.*
Defects	14%	-3%	*Positive indicator, the system is getting more stable.*
Stories	18%	+4%	*A positive indicator, developers are happy to have more time to develop. The decrease of defects had a direct impact in this indicator.*
Release and Deployment	2%	-2%	*The developers have automated some manual tasks related to the release process. This has helped in doing the release faster and with less human errors.*
Team Management	7%	+4%	*The increase on the team management effort makes sense because the merge had a great impact in the way the team worked, and coaching had to be done by the management.*
Business Analysis	4%	+4%	*Very positive indicator, we are now finally able to provide this service to the customer this means value added to the team and more interesting tasks to be done*
Automated testing	4%	+4%	*Very positive indicator, we are now finally able to provide this service to the customer this means value added to the team and more interesting tasks to be done*

Architecture	10%	+10%	*Very positive indicator, we are now finally able to provide this service to the customer this means value added to the team and more interesting tasks to be done*
Operational Work	26%	-24%	*The most interesting indicator, a lot of operational work was reduced, this was because the system is more stable, and developers have automated most part of the manual tasks*

Q5: What is your general opinion after implementation of Phase2

Phase2 brought something very desired by the developers, automated testing, this was a key factor to the decrease of defects and gaining speed on deployment. Also, we started to develop more and having less defects. The communication with PO's has slightly improved but is my feeling that they have severe difficulties adapting to this new way of working

Q6: Comment the percentage of time spent on each activity after implementation of Phase2, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	21%	+9%	*The time we spent in meeting is increasing but I can understand the need, it's essential that we learn how to work together in this new approach.*
Defects	7%	-10%	*Very positive indicator, we decrease the defects almost in half, this has impact in the team mood and customer satisfaction and we also have more time to develop.*

Stories	22%	+8%	*Good indicator, the shift from time spent in defects to time spent developing new features is positive to the team, we can now develop more, the system is more stable and we feel that the output of our work is recognized.*
Release and Deployment	3%	-1%	*I cannot judge due to that fact that I'm not so close to the release process.*
Team Management	6%	+3%	*Now more time is needed to coach and align the team in the common goals, makes sense from my point of view this increase.*
Business Analysis	4%	+4%	*This result remained unchanged since Phase1, we could have invested more time in this role, however this is management decision.*
Automated testing	7%	+7%	*Increase in the testing activities lead to the rise of this number, is expected.*
Architecture	10%	+10%	*No change since Phase1, for now I believe we don't need to invest more time in architecture.*
Operational Work	20%	-30%	*We could reduce the operational work from Phase1, it's a real good indicator. I can see that the work is almost all optimized and we spent now a reasonable amount of time in operational tasks.*

<u>Interview 4</u>

Personnel and professional questions

Age: 30
Years of IT Experience: 6

Area:
Operations:
Development: X

3. Open Questions about the team merge

Q1: What is your general opinion about the baseline situation (before the DevOps merge)

The situation s not very positive, the team has a good environment but when it comes to interact with PO things are not smooth. The team should have a more strategic mindset and not so siloed.

Q2: Comment in the below table the percentage of time spent on each activity in the baseline situation

Activity	**%**	**Comments**
Meetings	12%	*Too much time spent in meetings, negative mark.*
Defects	17%	*Too much time spent in defect solving, it's demotivating.*
Stories	14%	*This should be the main focus of our work and it's not, we need to develop more.*
Release and Deployment	4%	*Don't have so much vision, its done by operations team.*

Team Management	3%	*Seems to me reasonable, but also don't have a clear idea.*
Business Analysis	0%	*We don't perform this activity, but we should and the goal to move to DevOps must help us doing this activity*
Automated testing	0%	*We don't perform this activity, but we should and the goal to move to DevOps must help us doing this activity*
Architecture	0%	*We don't perform this activity, but we should and the goal to move to DevOps must help us doing this activity*
Operational Work	50%	*This is our worst mark. It does not seem sustainable spending so much time in operational work.*

Q3: What is your general opinion after implementation of Phase1

We could notice some changes after Phase1. Anyway, it's my expectation that the great benefit to the developers will be the implementation of automated testing and that will only come in Phase2. Some less positive things happened like the time that we took to implement 4 capabilities: 12 months. The time spent in meetings due to the lack of experience of them team in the DevOps model, however I can understand the need and it will improve when the team maturity also improves.

Q4: Comment the percentage of time spent on each activity after implementation of Phase1, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**

Meetings	15%	+3%	*Even if I understand the need for team meetings to learn how to work in a DevOps approach this is not a positive number. Needs to improve.*
Defects	14%	-3%	*Good mark, I was expecting slightly better but already a positive achievement, I believe it will improve in Phase2*
Stories	18%	+4%	*Good mark, finally the Dev team starts to spend more time developing than solving defects. We still have margin to improve specially when Phase2 comes.*
Release and Deployment	2%	-2%	*Dev team has automated some manual tasks in the release process, this has saved time to the operations colleagues.*
Team Management	7%	+4%	*This mark was unavoidable, the team had no experience in DevOps and it needed help from the management to learn how to work in this new model.*
Business Analysis	4%	+4%	*Good mark, finally the team can provide this service.*
Automated testing	4%	+4%	*Good mark, finally the team can provide this service.*
Architecture	10%	+10%	*Good mark, finally the team can provide this service.*
Operational Work	26%	-24%	*Very good mark, the operational work was reduced almost in 50%, this is very motivational and a very good message for the other projects.*

Q5: What is your general opinion after implementation of Phase2

Very positive outcome of Phase2, the automated testing was a real improvement on the way we work, defects were decreased and it's very useful. I can no longer imagine myself working in the old way. We took 18 months doing this shift but worth the time.

Q6: Comment the percentage of time spent on each activity after implementation of Phase2, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	21%	+9%	*We could spent less time in meetings but it's understandable while we still have doubts in this new way of working*
Defects	7%	-10%	*Very positive number, the automated testing helped the developers to increase quality and avoid mistakes.*
Stories	22%	+8%	*Very positive number, we have now time to develop more.*
Release and Deployment	3%	-1%	*Im not so close to release deployment so I cannot judge this value.*
Team Management	6%	+3%	*Seems reasonable.*
Business Analysis	4%	+4%	*Good mark, finally the team can provide this service.*
Automated testing	7%	+7%	*Good mark, finally the team can provide this service. This has improved a lot the team performance, I think this was the best achievement in the last 18 months.*
Architecture	10%	+10%	*Good mark, finally the team can provide this service.*
Operational Work	20%	-30%	*Impressive number, I was not expecting such a decrease on the operational work, this is highly motivating and also from a financial point of view is very positive.*

Interview 5

Personnel and professional questions

Age: 36
Years of IT Experience: 10

Area:
Operations: X
Development:

4. Open Questions about the team merge

Q1: What is your general opinion about the baseline situation (before the DevOps merge)

In the Operational area we have a considerable number of manual tasks. We still work based on emails which is very inefficient, and we fell the need to automate tasks. We are much more reactive to problems than proactive. Even if we work hard we fell that we have no progress. However the team spirit is very good which helps do the best we can.

Q2: Comment in the below table the percentage of time spent on each activity in the baseline situation

Activity	%	Comments
Meetings	12%	*The time spent in meetings is necessary, to align daily how to proceed.*

Defects	17%	*This number is significant, the most part of defects occurs from the fact that the system is not stable and misunderstood between PO and Dev's*
Stories	14%	*Few time for development, makes no sense from my point of view, that a Dev team spent only 14% of his time developing.*
Release and Deployment	4%	*This is done by the operations team, significant number of manual tasks needs to be done every release, I see here room for automate tasks.*
Team Management	3%	*I don't have a clear vision about this number.*
Business Analysis	0%	*We don't have this role in the team by lack of capacity and not skills.*
Automated testing	0%	*We don't have this role in the team by lack of capacity and not skills.*
Architecture	0%	*We don't have this role in the team by lack of capacity and not skills.*
Operational Work	50%	*Most part of time is doing operational work, manual tasks, incidents, firefighting problems. It's demotivating such a way of working.*

Q3: What is your general opinion after implementation of Phase1

The implementation of the 4 capabilities in Phase1 helped the operations, the developers have automated a huge number of manual tasks, which has reduced the error rate in releases and also daily tasks that were manually executed. Still room to improve, some team members are still having difficulties in adapting to this way of working.

Q4: Comment the percentage of time spent on each activity after implementation of Phase1 and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	15%	+3%	*The time spent in meeting is useful and it helps improving the way we work*
Defects	14%	-3%	*Good news, defects are finally starting to decrease, I expected a bigger decrease but still a good number.*
Stories	18%	+4%	*Those are good numbers, the Dev team is being more productive.*
Release and Deployment	2%	-2%	*The release time has decreased to half, the developers have automated release tasks and this has decreased the release time and the errors.*
Team Management	7%	+4%	*A huge effort was made by the management side to put the team working in a DevOps approach, so is normal that this value normal is higher than in the baseline*
Business Analysis	4%	+4%	*With the reduction in the operational work we can provide this role to the customer, this fact is very positive to the team and the colleagues like to perform this more challenging roles.*
Automated testing	4%	+4%	*With the reduction in the operational work we can provide this role to the customer, this fact is very positive to the team and the colleagues like to perform this more challenging roles.*

Architecture	10%	+10%	*With the reduction in the operational work we can provide this role to the customer, this fact is very positive to the team and the colleagues like to perform this more challenging roles.*
Operational Work	26%	-24%	

Q5: What is your general opinion after implementation of Phase2

The main benefit of Phase2 was seen into the development process, with the implementation of automated testing the developers can deliver with high quality. From my point of view Phase1 was much better for the operations than development because it decreased a lot the operational work and Phase2 was better for the developers who saw the defects decrease and their development time increase.

Q6: Comment the percentage of time spent on each activity after implementation of Phase2, and mention for each activity which capabilities you consider that have contributed the most.

Activity	**%**	**Δ% to Baseline**	**Comments**
Meetings	21%	+9%	*Seems to me higher than desirable, we have room for improve here.*
Defects	7%	-10%	*Very good result, time spent on Defect solving has decrease mainly due to automated testing.*
Stories	22%	+8%	*Good result, developers have now more capacity to deliver new features in a shorter period of time*

Release and Deployment	3%	-1%	*Interesting result, we deploy much more frequently and we don’t spent much time in the release, the process is highly automated.*
Team Management	6%	+3%	*Expected*
Business Analysis	4%	+4%	*Result already achieved in phase1.*
Automated testing	7%	+7%	*With more investment in automated testing is normal that this value has increased*
Architecture	10%	+10%	*We already had achieved this result in Phase1*
Operational Work	20%	-30%	*Very good achievement, the team is now free of al the boring manual tasks and firefighting.*