INSTITUTO
UNIVERSITÁRIO
DE LISBOA

DevOps Dashboard

Francisco João Lúcio Bruno

Master’s in Computer Science and Business

Supervisor:
PhD Ruben Filipe de Sousa Pereira Pereira, Assistant Professor
ISCTE-IUL

Co-Supervisor :
Rafael Saraiva Almeida

November, 2021

iscte

TECNOLOGIAS
E ARQUITETURA

DevOps Dashboard

Francisco João Lúcio Bruno

Master’s in Computer Science and Business

Supervisor:
PhD Ruben Filipe de Sousa Pereira Pereira, Assistant Professor
ISCTE-IUL

Co-Supervisor :
Rafael Saraiva Almeida

November, 2021

Resumo

DevOps é uma cultura que combina desenvolvimento e operação e que tem como principais objectivos reduzir o tempo de chegada ao mercado, fazer mudanças incrementais em resposta à mudança das condições, e construir um processo de desenvolvimento mais racionalizado.
DevOps é adotado em todo o mundo, e com a adoção em massa, vêm as diferentes implementações e padronizações.
Contudo o software responsável por agregar métricas não é de fácil implementação a nível de negócio e tem sido um problema para várias organizações.
Com o intuito de medir e monitorizar software, existe a premissa de utilizar um painel de maneira a simplificar a forma como o DevOps pode interagir com as métricas.
Esta tese centra-se no desenvolvimento de um painel de DevOps focado nas boas praticas de visualização com o objetivo principal de apoiar as equipas DevOps na tomada de decisões.
A fim de continuar a desenvolver o painel, foi escolhida a metodologia Design Science Research (DSR) com o objectivo de construir um artefacto e o avaliar.
Foi identificado que os dashboards utilizados na comunidade DevOps carecem de uma perspectiva mais ampla de todo o ecossistema de forma ajudar as partes interessadas na tomada de decisões.
A contribuição desta investigação é o painel de DevOps que monitoriza um sistema de DevOps, que segue as melhores praticas de visualização, utilizando categorias de métricas de maneira a mais facilmente navegar e interpretar os dados, a fim de melhorar a experiência do utilizador e tomada de decisão.

Keywords: Métricas de DevOps; DevOps KPI; Painel de DevOps .

Abstract

DevOps stands for development and operations. DevOps is a culture that empowers both development and operations teams to reduce time to market, make incremental improvements in response to changing conditions, and create a more efficient development process.

Software development and delivery is a very complex practice, and managing it is even harder. Any kind of system or process needs to gather data and metrics to understand how it is performing.

Understandably, measuring is essential in creating valuable software. However, measuring software is not easy and has been a problem for several organizations. There is the notion of utilizing a dashboard to ease the way DevOps teams interact and respond to data collected from systems to aid stakholders measure and monitor.

The Design Science Research (DSR) methodology was chosen to build an artefact – the DevOps dashboard - and evaluate its value to the DevOps community. Several versions of the artifact were developed as part of an improvement process, with each iteration validated through interviews.

It was identified that the dashboards generally accessible in the DevOps community are extremely specialized and lack a broader perspective of the entire ecosystem to help stakeholders in decision-making.

The contribution of this research is the developed dashboard that allows more significant monitoring of a DevOps system employing metric categories that follow visualization best practices to improve user experience and impact the user decision process positively.

Keywords: DevOps metrics; DevOps KPI; DevOps dashboard.

Index

Resumo **i**
Abstract **iii**
Index **v**
Index of tables **vii**
Index of figures **ix**
Listo of Abbreviations and Acronyms **11**
Chapter 1 - Introduction **1**
Chapter 2 – State of the art **3**
2.1. DevOps and its importance 3
2.2. Metrics in a Development Environment 3
2.3. Dashboard 4
2.4. Related Work 6
2.4.1. Outlining Systematic Literature Review 7
2.4.2. Conducting a Systematic Literature Review 8
2.4.3. Reporting the SLR 10
2.5. Literature Review Synthesis 17
Chapter 3 – Research Methodology **19**
Chapter 4 - Proposal and Evaluation **23**
4.1 . First DSR Iteration 23
4.1.1. Proposed Dashboard 23
4.1.2. Research and Analysis 23
4.1.3. Dashboard elaboration 24
4.1.4. Guidelines 24
4.1.5. Demonstration 25
4.1.6. Evaluation 27
4.2. Second DSR Iteration 29
4.2.1. Proposal 29
4.2.3. Demonstration 29
4.2.4. Evaluation 30
4.3. Third DSR Iteration 31
4.3.1. Proposal 31
4.3.2. Demonstration 31
4.3.3. Evaluation 34
4.4. DSR Synthesis 35
Chapter 5- Conclusion **37**
5.1. Contributions 38
5.2. Limitations 38
5.3. Future Work 39
References **41**
Appendices **45**
Appendix A – Questionaire - 1st Iteration 45
Appendix B – Questionaire - 2nd Iteration 46
Appendix C – Questionaire - 3rd Iteration 47

Index of tables

Table 1- *Stephen Few Color Rules* [28] 5

Table 2- *Gestalt principles* [19] 6

Table 3- *Outlining Systematic Literature Review* 7

Table 4- *Filters used in the SLR* 7

Table 5- *Article percentage per year* 8

Table 6- *SLR filters application and results* 10

Table 7- *Metrics per article and categories* 12

Table 8- *Metrics and respective description* 13

Table 9- *Articles vs Dashboard Criteria* 14

Table 10- *Articles with dashboards vs Stephen Few color rules* 15

Table 11- *Articles with dashboards vs Gestalt Principles* 16

Table 12- *Design science research principles* 20

Table 13- *Design Science Research Guidelines* 20

Table 14- *Key Evaluating Questions* 21

Table 15- *Evaluation prototype structure* 21

Table 16-*Improvements prototype structure* 21

Table 17- *Interviewee Data* 23

Table 18 - *Evaluation of the prototype - 3rd Iteration* 34

Table 19- *Proposed improvements by Iterations* 35

Index of figures

Figure 1- *State of the Art selection diagram* 6

Figure 2- *Articles focus keyword* 9

Figure 3- *Articles that are related to dashboards* 9

Figure 4- *Dashboard criteria by percentage* 15

Figure 5- *DRSM Model followed in the research* 19

Figure 6- *Productivity Dashboard* 26

Figure 7- *Performance Dashboard* 26

Figure 8 - *Quality Dashboard* 27

Figure 9- *Dashboard homepage* 29

Figure 10- *Quality vs Performance Dashboard* 32

Figure 11-*Quality vs Productivity Dashboard* 33

Figure 12- *Performance vs Productivity* 33

List of abbreviations and acronyms

DSR	– Design Science Research
IT	– Information Tecnology
DORA	– Devops Research and Development
SLR	– Systematic Literature Review
UI	– User Interface

Chapter 1 - Introduction

Organizations have been using technology for competitive advantage, create value for their customers and reach them in an improved way [1]. Information Technology (IT) driven enterprises like Apple, Facebook and Oracle have shown that by investing in developing their software technologies, companies can build a stronger lead across several market domains [2]. However, it can be difficult for a corporation to keep up with the rate of how technology changes nowadays [3].

According to a report published by Tricentis, software failures caused more than $1.7 trillion in financial losses in 2017 [4]. Therefore, having the ability to deliver reliable software in a safe manner quickly is core to a successful technology transformation [5], [6]. As a result, there is an increasing need to manage software quality (SQ), and companies should strive to avoid software failures and aim at progressively improving software [7]. However, most companies still have trouble orchestrating their developers and operations teams, which leads to several software problems [6]. To decrease these problems, Debois [8] enforced a closer relationship between Development and Operations, which is now designated as DevOps. Since then, numerous corporations have started to shift into a DevOps approach [9].

According to Hemon [10], improving the collaboration between different technology branches is a prerequisite for survival. For instance, companies such as Netflix, Amazon and Facebook have adopted DevOps within their business organizations. One of the substantial benefits of DevOps is the commitment to deliver software at a quicker pace, provided by the faster deployment rate, which means a faster time to market [11]. Complementary to the increase in the software deployment speed, the quality of the software is also targeted. This methodology includes de automation and constant monitoring of all stages of the build process [12].

The increasing adoption of DevOps [13] is driving value into organizations across all industries quicker and in a more sustainable manner than ever before. However, the ability to evaluate and measure the impact of DevOps within corporations has been one of the problems at hand. After all, you cannot manage what you cannot measure [11]. Many companies want to embrace DevOps, however owing to IT system complexity already implemented, gathering metrics that offer insight across the pipeline is difficult. Diagnosing a problem requires inspecting all the information collected, including log statements, memory consumption, system metrics [14], all of which can be very lengthy, consuming a significant portion of the developer time [15].

DevOps is being more and more adopted worldwide as its benefits spread in practical and scientific forums [16]. However, measuring and monitoring DevOps adoption is not clear and deserves further investigation. Some institutions, like Devops Research and Assessment (DORA), are now focused on that specific goal. Considering that monitorization helps determine the main problems with a system [18], it can be asserted that teams need an integrated view of metrics to respond to problems quicker. One criticism commonly seen in the many dashboards used in the literature is the amount of irrelevant data and visualizations utilized without a practical purpose and without taking the end-user into account. [17].

Based on the preceding assertions, the purpose of this research is to propose the creation of a DevOps dashboard that adheres to best practices in data visualization to assist stakeholders in monitoring their systems and making decisions.

This research is structured in the following way: The next chapter introduces the State of the Art, which is mainly focused on introducing the theoretical background and emphasizing the related work and how DevOps is being researched, followed by the third chapter where the focus is the Research Methodology that was applied during the research.

The fourth chapter is dedicated to the presentation of the proposal, which includes the results of the developed iterations.

Finally, the conclusion in Chapter five serves to highlight the major findings of the research, research limits, lessons learned, and future work.

Chapter 2 – State of the art

This chapter will present a review of the literature.

2.1. DevOps and its importance

DevOps was introduced to address the bottleneck created by Operation teams that were setting back Development teams from delivering in a more agile manner [6]. DevOps allows the software community to be cross-disciplinary [3], allowing them to build more resilient software and also be able to deliver and evolve in a faster way. This paradigm can be condensed in the phrase "You build it, you run it" [5].

For instance, developers need to understand the production environment where their code is released. In the same way, Operations need to work compatibly with the code developed [3]. There is a rising necessity for code changes to be delivered rapidly to satisfy the always increasing requirements in such a way as to stay relevant in such a competitive market [20]. Nowadays, companies and teams are faced with constant changes and new requirements. Teams need to work in an agile manner to reduce the whole development cycle by orchestrating both development and the operations team [18].

DevOps seeks to reduce the time between changing a software and deploying it into production environments while keeping quality standards[19]. Being able to profit from DevOps has been an evident differentiator in vying for end-user awareness and financing. [20]

It is important to understand that DevOps heavily rely on automation and tool support. The ability to ease the deployment process and the whole way teams are working together is essential to develop further their technical savviness and their culture [21].

2.2. Metrics in a Development Environment

A very important target in DevOps is to improve productivity to generate more value for their customers. It is a consensus that productivity is essential. Productive employees work more efficiently, allowing them to spend their working hours not just developing but also properly documenting, refactoring or even delivering additional features [5].

According to Forsgren [3], productivity is "the capacity to perform difficult, time-consuming jobs with minimum distractions and interruptions, It is not as simple to evaluate productivity as it is to examine metrics such as lines of code, tickets finished, or problems fixed. This puts the team's macro-goals in jeopardy. [22]

Organizations nowadays are keener on adhering to technology transformations with major goals of improving their quality as a service or product. Measuring quality is difficult because there is always a need to understand the context, and it can change from company to company [23].

For a user, quality is one of the most important factors regarding its satisfaction with the product or service, which is clearly related to business growth. One of the ways to measure software quality is using software quality characteristics. These characteristics are a set of attributes that a software product can be described and evaluated. Software quality is defined as "the totality of characteristics of a software product that satisfy stated or implied needs" [24]

It is key to leverage software quality in order to increase customer satisfaction. Corporations need to leverage DevOps to improve their overall product quality, and continuous quality monitoring is directly related to continuous development or continuous deployment [25].

The performance of the software can be characterized by numerous system properties that have access to the system timeliness and the use of related resources [26]. Performance can be measured by looking at the throughput of code and the stability of the system itself. Some performance metrics are response time, resource utilization, lead time to state a few.

2.3. Dashboard

The volume of data available to corporations and teams is growing exponentially, with some experts believing its growing by 60% every year [19]

There is a clear need to improve a company communication effectiveness the more a company grows. Therefore, corporations have to resort to tools to monitor their systems and products. Dashboards are a very common tool to improve the efficiency that the data outputs presented to the end-user [19]

Dashboards are able to aggregate in a single screen the data that the end-user has the purpose of monitoring. Dashboards are often heavy on graphics and charts, and their aim is not only to be visually appealing, but also because people grasp and consolidate information more quickly from visuals than from words. [17].

A dashboard can be used as a tool to assist organizations in making decisions, such as understanding and measuring a product sustainability and resource use levels [18]. However, even though it is obvious that dashboards provide added value to those who use them, many have failed to realize the full potential of a well-developed and designed dashboard; a common mistake is not focusing on the way the data is displayed and not maximizing the visual keys the way they are meant to pass on to the users [19].

It is clear that there is a significant gap between knowing the effectiveness of a dashboard and how it is utilized in the field, which has frequently been employed as a strategy to simply appeal to new potential clients [30].

In addition to the content of the dashboard, the manner in which it was created is a crucial aspect in its efficacy and general use. It is also important to understand if the dashboards follow industry best practices and rules. The evidence suggests, even the greatest dashboard software will not be effective and will not reach its full potential if the visual element and design are not taken into account [19].

According to [19], dashboards should adhere to a set of colour guidelines in order to convey the objective of the dashboard as clearly as feasible. The colour guidelines and their explanations may be found in Table 1. Furthermore, in order to standardize the way dashboards are created, the Gestalt-Principles in Table 2 might aid in the provision of recommendations for dashboard implementation.

Gestalt theory has the purpose of explaining how humans interpret visual elements and aggregate them into groups and how individuals distinguish visual patterns. Previous authors argued that the Gestalt principles could provide a framework with the capability of categorizing visual guidelines [31].

Table 1- *Stephen Few Color Rules* [28]

Rule	Description
Rule 1	If you want different objects of the same color in a table or graph to look the same, make sure that the background-the color that surrounds them-is consistent
Rule 2	If you want objects in a table to be easily seen, use a background color that contrasts
Rule 3	Use color only when needed to serve a specific communication goal.
Rule 4	Use different colors only when they correspond to differences of meaning in thedata.
Rule 5	Use soft, natural colors to display most information and bright and/or dark colorsto highlight information that requires greater attention.
Rule 6	When using color to encode a sequential range of quantitative values, stick witha single hue (or a small set of closely related hues) and vary intensity from palecolors for lower values to increasingly darker and brighter colors for higher values.
Rule 7	Non-data components of tables and graphs should be displayed just visibly enough to perform their role
Rule 8	To guarantee that most people who are colorblind can distinguish groups of data that are color coded, avoid using a combination of red and green in the samedisplay.
Rule 9	Avoid using visual effects in graphs.

Table 2- *Gestalt principles* [19]

Principle	Description
Proximity	Objects that are physically close together are perceived as a group
Similarity	Objects thar are of similar color, shape, size, or orientation are perceived as related or belonging to part of a group
Enclosure	Objects that are enclosed create the idea of belonging to the same group
Closure	We tend to complete the form of objects that are open, incomplete or outside the norm
Continuity	We tend to group objects that are aligned or form some sort of continuity
Connection	Objects that are interconnected create an idea of a group

2.4. Related Work

The present body of completed and documented work generated by academics, scholars, and practitioners was synthesized using a systematic literature review (SLR) [29].

This SLR was based on the author Kitchenham's [29] criteria, and the procedures taken to perform the review are depicted in Figure 1.

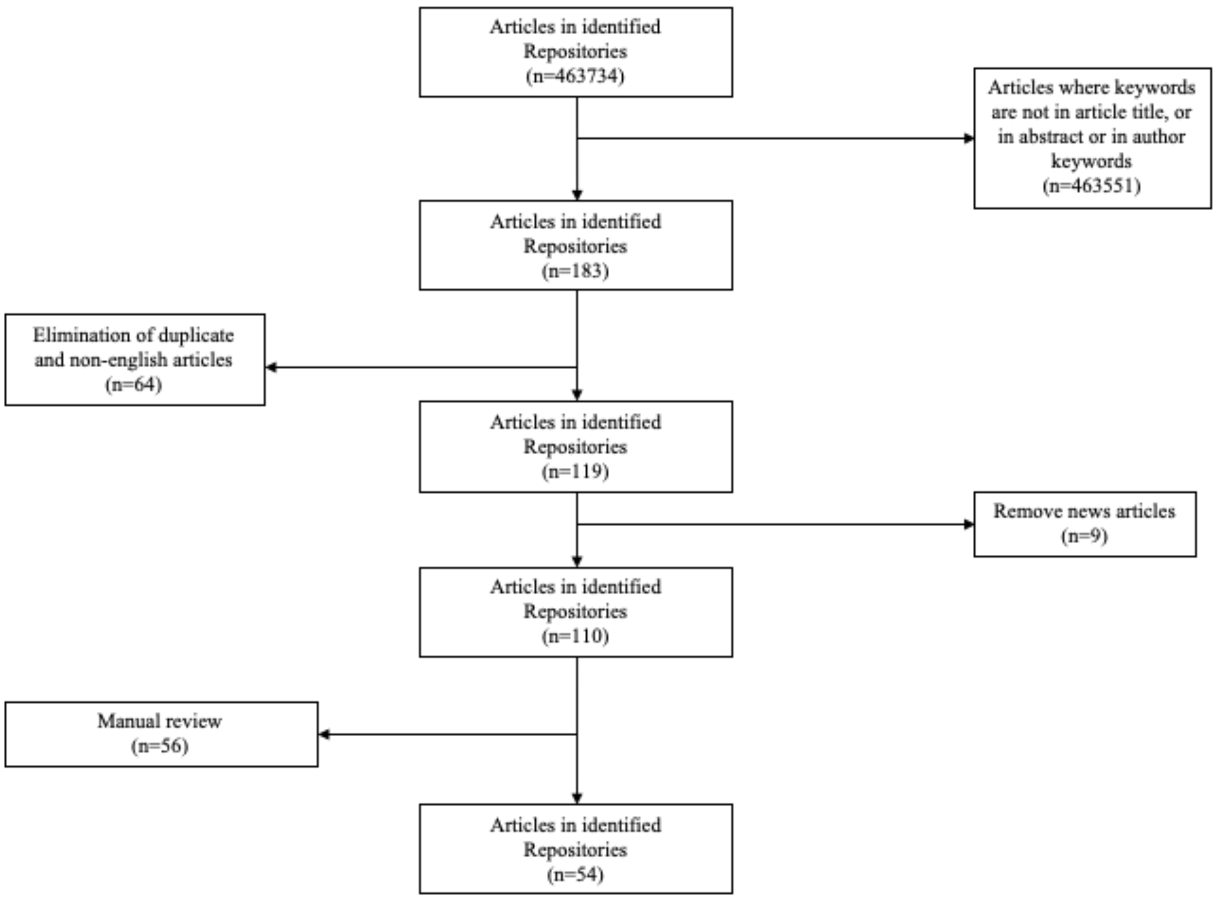

Articles in identified Repositories (n=463734)
Articles where keywords are not in article title, or in abstract or in author keywords (n=463551)
Articles in identified Repositories (n=183)
Elimination of duplicate and non-english articles (n=64)
Articles in identified Repositories (n=119)
Remove news articles (n=9)
Articles in identified Repositories (n=110)
Manual review (n=56)
Articles in identified Repositories (n=54)

Figure 1- *State of the Art selection diagram*

2.4.1. Outlining Systematic Literature Review

Since DevOps as a culture has been striving in recent years, the quantity of research being conducted is also being developed concurrently, which leads to a sizable number of articles and therefore the search had to be integrated with the keywords “Metrics, “KPI”, “Indicators” and “Dashboard”, which allowed the search to be more detailed. As previously stated, a SLR was done with the goal of addressing the objective of this research. The actions required to carry out this review may be seen in Table 3.

Table 3- *Outlining Systematic Literature Review*

Outlining Systematic Literature Review	Conducting Systematic Literature Review	Reporting the Review
Identification of the need for a review: Lack of standardization in DevOps metrics and monitoring practices	Applying filters and get final articles: 54 articles	Report the findings: Discussion related to how is the DevOps community measuring their systems and teams and how they monitor those metrics
Objective of the review: Research how DevOps implementations are being measured	Perform data extraction and analyse the sample: Extract information about DevOps metrics Analysis of the sample	
Review Protocol: Search String, filters, repositories, and inclusion criteria		

In the end, the following search string was applied with the Boolean OR and AND: (DevOps" AND ("Metrics" OR "KPI" OR "Indicators" OR "Dashboard")).

The first query for the selected keywords was conducted without any filter and returned a total of 463734 articles, which was too much to be examined. From this point moving forward, there was a need to start using filters depicted in Table 4 to get a smaller range of articles to study.

Table 4- *Filters used in the SLR*

Filter Name	Criteria
1st Filter	Keywords only in the article keywords, in the abstract or title
2nd Filter	Discard duplicated articles and non-English
3rd Filter	Remove news articles
4th Filter	Manual review

The 1st filter that was applied is the search for the keywords only in the article keywords, in the abstract or title; The 2nd filter was to discard article duplicates and non-English articles;

On the 3rd filter, all the news articles were removed; And finally, the 4th filter was the manual review of the data set removing the articles where the thematic was not related to the research.

It had to be considered that all the repositories have their own search approach, and therefore the search string had to be adapted accordingly.

To pursue the research, the following repositories were selected

- Google Scholar https://scholar.google.com
- IEEE https://ieeexplore.ieee.org/
- ACM https://dl.acm.org
- SpringerLink https://link.springer.com
- Web of Science https://apps.webofknowledge.com/
- EbcCohost https://search.ebscohost.com/
- DORA https://www.devops-research.com/research.html

2.4.2. Conducting a Systematic Literature Review

After applying all the filters in the chosen repositories, in Table 5 and Table 6, we can see that the final set of articles ended up being 54.

There is the need to mention DORA a research centre focused on DevOps that publishes a report every year regarding the best practices developed and analysed that year, which is very specific to this research.

After applying the filters, an individual article analysis was done to identify how DevOps was being implemented and practiced with or without dashboards. The year of the article, as well as whether or not it was linked to the dashboard, were retrieved for further analysis for each piece. As mentioned before, no date filters were applied. However, it is important to understand how recent the data set is. Most of the articles are from 2016 onwards as it can be seen in Table 5.

Table 5- *Article percentage per year*

Years	Percentage	Total
2011	7%	4
2013	4%	2
2014	6%	3
2015	4%	2
2016	13%	7
2017	9%	5
2018	15%	10
2019	15%	8
2020	22%	12
2021	4%	2
		54

It is certain that DevOps is still growing as a culture. According to [30], DevOps teams increased 16% in 2014, 19% in 2015, 22% in 2016 and 27% in 2017. It is clear that research has also been increasingly developed alongside the increasing number of DevOps teams that are implementing the philosophy.

According to Figure 2, it is possible to see the distribution of the papers by keyword. There is a clear discrepancy in the articles with most of the articles being related to either DevOps Metrics or Dashboards. Has it can be seen in Figure 3, 28% of the articles mentioned the use of a dashboard. By observing the data set, it can be found that dashboards applied to DevOps are not researched as intensively as DevOps in their entirety

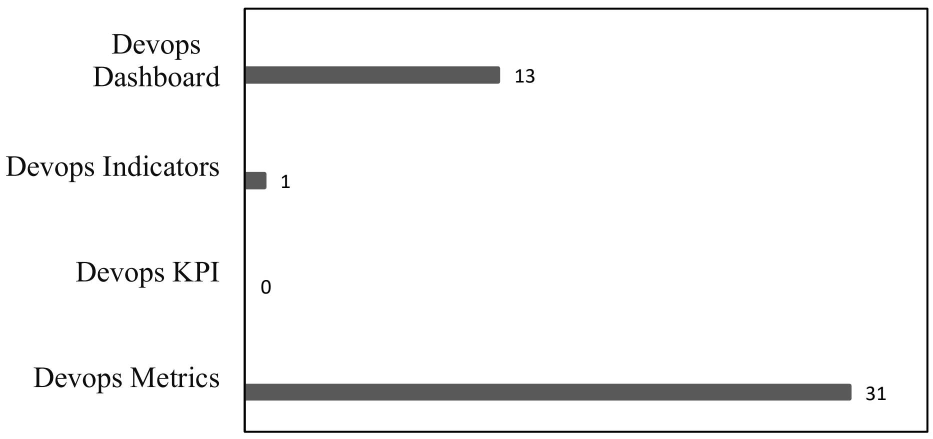


Figure 2- *Articles focus keyword*

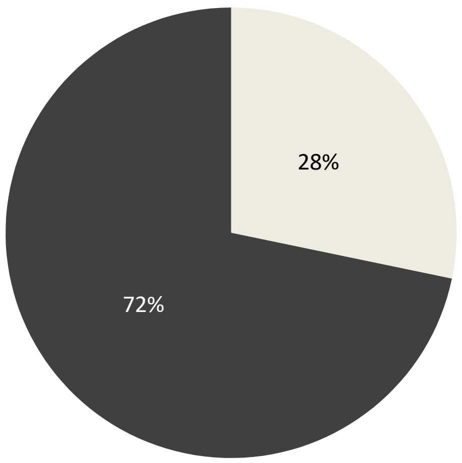


Figure 3- *Articles that are related to dashboards*

.

Table 6- *SLR filters application and results*

Source	Keywords	No filter	1 Filter	2 Filter	3 Filter	4 Filter
Google Scholar	DevOps Metrics	7130	12	3	4	4
	DevOps KPI	827	0	0	0	0
	DevOps Indicators	2680	0	0	0	0
	DevOps Dashboard	2810	2	0	0	0
IEEE	DevOps Metrics	19	12	5	5	5
	DevOps KPI	0	0	0	0	0
	DevOps Indicators	2	2	0	0	0
	DevOps Dashboard	4	1	1	1	1
ACM	DevOps Metrics	142242	10	5	5	7
	DevOps KPI	1936	0	0	0	0
	DevOps Indicators	297923	0	0	0	0
	DevOps Dashboard	6091	1	1	1	1
SpringerLink	DevOps Metrics	724	43	43	43	7
	DevOps KPI	71	0	0	0	0
	DevOps Indicators	315	1	1	1	1
	DevOps Dashboard	351	6	6	6	6
Web of Science	DevOps Metrics	28	19	19	19	9
	DevOps KPI	0	0	0	0	0
	DevOps Indicators	3	2	1	1	0
	DevOps Dashboard	4	2	1	1	1
Scopus	DevOps Metrics	424	33	9	9	2
	DevOps KPI	14	1	0	0	0
	DevOps Indicators	70	5	0	0	0
	DevOps Dashboard	40	9	5	5	4
EBSCO	DevOps Metrics	9	9	6	0	0
	DevOps KPI	5	3	3	0	0
	DevOps Indicators	1	0	0		0
	DevOps Dashboard	**5**	4	4	0	0
DORA	DevOps Metrics	6	6	6	6	6
	DevOps KPI	0	0	0	0	0
	DevOps Indicators	0	0	0	0	0
	DevOps Dashboard	0	0	0	0	0
	TOTAL	**463734**	**183**	**119**	**106**	54

2.4.3. Reporting the SLR

In this section, this research will focus on several types of dashboards and how metrics were aborded in the related articles.

The objective of this chapter is to interpret and understand if the methodologies applied in the selected articles are of aid to the DevOps teams. To analyze the articles, it was necessary to determine what metrics were used in order to understand what is now standard practice and whether there is any type of pattern or distribution. According to [36], a metric is a “standard of measurement that defines the conditions and the rules for performing the measurement and for understanding the result of a measurement".

Harvesting measurements from the software pipeline that is of any use is difficult. The data in question needs to be put through a process of quality assurance in order to be further utilized as a way to drive business value [37].

Therefore, if corporations are willing to apply a system based in metrics it is relevant to aggregate the metrics in sub-groups to further understand how metrics are being utilized and if there is a pattern in the metrics corporations are mostly following [1], in Table 7 the metrics were aggregated by Productivity, Performance and Quality and then understand what is the focus of the applied metrics. However it is possible to understand that most of the times the scope of the system is not well defined which makes it hard to track the needed metrics.

In Table 8 it is possible to verify what metrics were scanned from the literature analysis. Furthermore, within software engineering, dashboards are utilized to provide teams with the required information and give access to selected metrics on the product or the team itself, having the bonus of being able to display information and support quicker decisions [38].

According to [2], they have identified four key metrics that should be an industry standard, allowing corporations to drive their organizational performance and promote technology transformation.

These four metrics are able to capture the efficacy of the development process and also the delivery process, being able to measure the delivery process the main used metric is the Lead Time of code changes, which is measured from the moment a feature is checked in to the point of its release into production, in combination with Deployment Frequency [2]. After measuring Software Development there is a need to validate the deployment which can be measured by using Change Fail Rate and Time to Restore that can also be implied in measuring the overall quality of not only the process but also the system that is in place [24].

According to Table 9, there are certain metrics that are found more often in the literature. It is still feasible to confirm that dashboard drill-down is not commonly used, additionally, it can be reported that there is still a lack of dashboards that are evaluated. To elaborate, a dashboard that implements the use of drill-down will allow the user to explore a data set at a more granular level [39]. The usage of correctly integrated filters that allow for simple and effective drill down is not only a value-added to the dashboard but also a technique to engage with the dashboard in a more effective manner [13].

Furthermore, as dashboards are always developing and being updated, having an evaluation criterion based on its current relevance, efficiency, and usefulness is critical. [38].

Table 7- *Metrics per article and categories*

Metrics categories		Productivity						Quality												Performance					
Year	Articles	Nº of requests	Nº of issues	Nº of developers	Nº of pushes	Sprint duration	Nº of developer per teams	Lines of Code chang	Deployment Frequency	Project duration	Nº of deploys	Availability	Nº of defects	Code Coverage	Mean-time	Lead time	Nº of faile	Activity coverage	Lines of Duplicated Code	Latency	Nº of crashes	% of CPU utilization	Change Fail rate	Nº of containers	Packet loss
2021	[31				●			●																	
	[32																			●		●			
2020	[33									●		●			●	●									
	[34				●			●						●								●			
	[11		●		●			●				●	●				●						●		
	[35	●										●			●							●			
	[36	●																		●		●		●	●
	[37		●					●				●		●					●						
	[38												●	●			●								
	[2]												●	●											
2019	[39	●									●	●										●		●	
	[40		●		●			●										●			●				
	[41		●		●			●					●												
	[10			●			●	●		●															
	[42	●																				●			
	[43											●										●			
	[3]							●																	
2018	[44		●	●	●			●					●	●					●						
	[45	●						●				●	●				●	●			●				
	[22								●			●				●							●		
	[1]					●			●			●	●		●										
	[46														●					●		●			
	[47												●	●									●		
	[48	●																		●					●
	[49			●				●				●													
	[50													●											
	[51							●																	
2017	[52			●		●				●	●	●	●			●									
	[30								●			●				●							●		
	[53	●						●														●			
	[54												●												
2016	[55						●		●						●	●							●		
	[56																			●		●			
	[57											●													
201	[58	●	●									●			●							●			●
	Tot	8	6	4	6	2	2	13	4	3	2	14	10	7	6	5	3	2	2	5	2	11	5	2	2

Table 8- *Metrics and respective description*

References	Metrics	Description
[10][55]	Nº of developer teams	Number of Developer teams in a Project
[10][44][49][52]	Nº of developers	Number of developers in a Project
[11][37][40][41][44][58]	Nº of issues	Number of issues that the teams have resolved or is solving
[35][36][39][42][45][48][53][58]	Nº of requests	Number of server requests made in a set period of time
[1][52]	Sprint duration	The set duration for the sprint
[11][40][41][44]	Nº of pushes	The number of pushes made to the code base
[11][37][40][41][10][3][44] [45][49][51][53]	Lines of Code Changed	Lines of code changed in the repository per commit
[22][1][30][54]	Deployment Frequency	The frequency that there is a deployment of a system/application
[10][52]	Project duration	Number of hours a project is going to take or has taken
[39][52]	Nº of Deploys	Number of Deploys for an amount of time
[37][44]	Lines of Duplicated Code	Lines of code blocks found in the repository
[40][45]	Activity coverage	Participation in as many activities of DevOps as possible, particularly implementation and verification
[37][38][2][44]55][50]57]	Code Coverage	While the automated tests are running, code coverage is a calculation of how many lines/blocks/arcs of the code are executed. Code coverage is determined by using a specialized tool to instrument binaries with tracing calls and then running a comprehensive series of automated tests against the instrumented product.
[11]52][2] [41][44] [45][1][47][52]51]	Nº of defects	Number of defects found in a code source
[35][1][54][42][33]	Mean-time to repair	Measures the average time required to troubleshoot and be able to repair a failed system and is calculated by dividing total maintenance time by the number of repairs
[11][38][45][22]	Nº of failed tests	Number of failed tests that occurred in the system
[22][52][30][55][33]	Lead time	Is the elapsed time between the identification of a requirement and the actual fulfilment of a feature
[11][35][37][55][43][45][22][1] [49][52][30][57][58][33]	Availability	The percentage of time your service or configuration item is available is referred to as availability. It provides information about the past and forecasts the future of service. It indicates how well a service is performed over a specified time span. A service's availability indicators can also predict how well it will perform in the future.
[40][45]	Nº of crashes detected	Number of crashes detected in a certain system or application
[36][46][48][56]	Latency	Latency is the time between a user's action and a web application's response to that action; it's also known as the cumulative round trip time a data packet takes to move in networking terms.
[35][36][39][42][43][46][53][56] [58][34][32]	CPU utilization %	The total amount of work handled by a Central Processing Unit is known as CPU utilization. It's also used to predict how well a device would do. Since certain tasks need a lot of CPU time while others need less, CPU utilization can vary depending on the type and amount of computing task.
[36][39]	Nº of containers	Nº of containers utilized in the project/system
[36][48][58]	Packet loss	When one or more packets of data travelling through a computer network fail to reach their destination, packet loss occurs. Packet loss is caused by either data transmission errors, which are common in wireless networks, or network congestion.
[11][22][47][30][55]	Change fail rate	The number of deployments that result in a product failure. It's time to re-establish service. How long it takes for a company to rebound from a production loss.

Table 9- *Articles vs Dashboard Criteria*

Article	Displays Dashboard	DrillDown	Data Visualization	Identified metrics	Scope	Was the dashboard evaluated?	What graphics do the dashboards use?			
							Line Graph	Pie Chart	Bar graph	Speedometer
[49]			✓	✓						
[44]			✓				✓			
[63]	✓	✓		✓	✓					
[11]	✓									
[60]	✓		✓	✓			✓	✓		
[15]	✓		✓				✓			
[13]			✓		✓	✓				✓
[64]	✓		✓		✓				✓	
[38]	✓	✓	✓	✓	✓		✓	✓	✓	
[46]	✓		✓	✓			✓			✓
[62]	✓	✓	✓		✓			✓	✓	
[49]				✓	✓					
[44]			✓	✓					✓	

In Figure 4, we can further verify by using percentages what criteria is mostly utilized in the selected articles.

	Yes	No
	Nº (%)	Nº (%)
Line chart	5 (38,4%)	8 (61,6%)
Pie chart	3 (23%)	10 (77%)
Bar Chart	4 (30,7%)	9 (69,3%)
Speedometer	2 (15,3%)	11 (84,7%)
Was the dashboard evaluated?	1 (7,7%)	12 (92,3%)

	Yes	No
	Nº (%)	Nº (%)
Displays Dashboard	8 (61,6%)	5 (38,4%)
DrillDown	3 (23%)	10 (77%)
Data Visualization	10 (78,3%)	3 (21,4%)
Identified metrics	7 (53,9%)	6 (46,1%)
Scope	6 (46,1%)	7 (53,9%)

Figure 4- *Dashboard criteria by percentage*

In Figure 4, we can analyse how charts are being used in the dashboard, being the line charts the most adopted. Furthermore, with the intent to analyse the considered dashboards, it was considered if they were compliant with the Stephen Few rules (Table 6) for applying colour in charts, to analyse the dashboard, the set of articles used was smaller since some of the articles in Table 5, did not display a dashboard making it impossible to verify their implementation, therefore not being able to verify their compliance

In Table 10, we can see how color is used in the dashboards under consideration. With a few exceptions, it is obvious that the majority of the regulations were obeyed.It is relevant to state that colour should be used sparingly, and the dashboard target must bevery clear because colour has a different interpretation depending on cultural aspects.

Table 10- *Articles with dashboards vs Stephen Few color rules*

Article	RULE1	RULE2	RULE3	RULE4	RULE5	RULE6	RULE7	RULE8	RULE9
[63]	✓	✓							✓
[60]	✓		✓	✓		✓	✓		✓
[15]	✓	✓	✓		✓				✓
[13]	✓	✓	✓	✓	✓		✓	✓	✓
[64]	✓	✓	✓		✓		✓	✓	✓
[62]	✓	✓	✓	✓			✓	✓	✓
[49]	✓		✓				✓		✓
[44]	✓	✓	✓	✓	✓		✓		✓

After analysing the colours there was a need to verify if the Gestalt principles were being applied in the dashboard design as it can be noted in Table 11.

However, when comparing the Gestalt principle, it is apparent that most of the studied articles did not implement the dashboard with regards to the principles. The Gestalt principles are commonly used to tie data together, separate data or make data stand out in the dashboard.

Table 11- *Articles with dashboards vs Gestalt Principles*

Article	Proximity	Similarity	Enclosure	Closure	Continuity	Connection
[63]	✓					
[60]	✓	✓		✓		
[15]	✓	✓	✓	✓		
[13]	✓	✓	✓			
[64]					✓	
[38]	✓		✓		✓	
[2]		✓			✓	✓
[66]	✓	✓		✓		✓

According to [63] design-wise goes against some of Stephen Few rules, such as to have the dashboard spread into several screens, which can increase the difficulty of usage to the end-user, it is crucial that the dashboard can be seen from a single screen making it simple and accessible to the final user. However, the way they brought innovation into the whole DevOps environment with the integration of a user interface (UI) heatmap is a good way to understand that dashboard development and DevOps metrics still is a growing area filled with innovation.

The article [64] has a more generic approach on the design of the dashboard where they use Splunk, to generate the dashboard after capturing the metrics through an ElasticSearch stack. ElasticSearch is an open-source search and analytics engine with the capability of ingesting data from several sources concurrently.

Utilizing ElasticSearch has the advantage of accelerating the development process, with the major drawback of having the design constrained to the tools available.

Commonly, in [15] also resorts to ElasticSearch however instead of utilizing Splunk for the dashboard implementation resorts to the ElasticSearch native tool Kibana, which has the particular ability to generate Dashboards with a very standardized designwhich is not just overall a pleasing implementation regarding its colour schemes, but it also uses most of the Stephen Few Rules of colour, and the Gestalt Principles are also present in mostof its implementation.

As a result, in terms of open-source software, ElasticSearch has the capacity of building dashboards while also supporting the whole data monitoring process, making it a frequent implementation in data-intensive systems with the goal of monitoring their data. After analyzing all of the articles, it can be stated that none of the data set articles go into depth on how to design and implement the dashboard because it is mostly a feature of their study and not its main focus, so there is a noticeable gap in the literature on how to properly design a DevOps dashboard in order to retain the maximum value from its implementation.

2.5. Literature Review Synthesis

The work in this SLR had the purpose of understanding how the DevOps community is leveraging dashboards in their work and trying to understand if there is any common practice being utilized by the community. In Table 7, there is not a standardized group of metrics being used across DevOps teams and that they are simply measuring key aspects of their system or project, not considering what has been studied and targeted as the key metrics to follow within the DevOps community. Although there should be a standardized range of metrics, it is also important to understand each use case to not measure something irrelevant or pointless [67].

According to Figure 4 one can also perceive that not everyone is implementing a dashboard to aid in visualizing software development or any monitoring. However, a big portion of those that use dashboards are following design methodologies and best practices to aid their organization and team.

The findings of this study are expected to help understand how to Monitor a Devops environment, as well as a fundamental understanding that there is much space for improvement, whether in terms of decision-making and understanding of what needs to be measured, whether the emphasis is on efficiency, quality, or productivity of the systems in question.

The primary difference between what has already been researched and what will be investigated in this study is how the best visualization approaches may assist in monitoring a DevOps environment and how metrics can be divided into categories, allowing the monitored data to be accessible and analyzed independently.

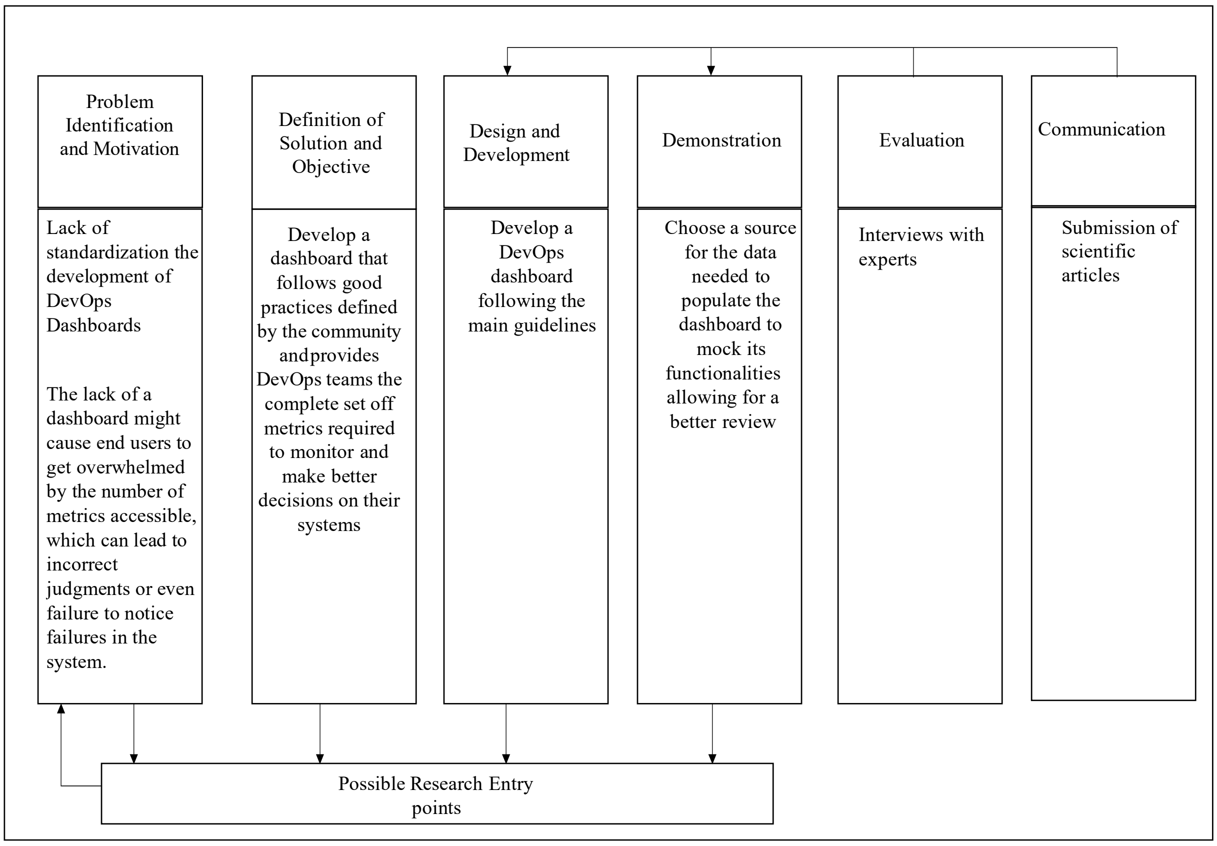
Chapter 3 – Research Methodology

Given the nature of this investigation, the Design Science Research (DSR) is the select research methodology to achieve this research goal.

Design Science Research Methodology (DSRM) is defined as the goal of developing and functionally researching an artifact or instruction [59], being regarded as a comprehensive process aimed at developing artifacts with the goal of issue resolution [60].

Additionally, DSR is considered to be a new technique that enables the development of research in several areas, being able to be utilized as a way to produce academic and organizational knowledge [59].

The use of DSR has the capability of reducing the gap between theory and practice by not only allowing to solve organizational and academic problems but also by producing knowledge with the capability of improving theories [59]. Figure 5 depicts how the DSR approach was implemented.



Problem Identification and Motivation
Definition of Solution and Objective
Design and Development
Demonstration
Evaluation
Communication
Lack of standardization the development of DevOps Dashboards
The lack of a dashboard might cause end users to get overwhelmed by the number of metrics accessible, which can lead to incorrect judgments or even failure to notice failures in the system.
Develop a dashboard that follows good practices defined by the community andprovides DevOps teams the complete set off metrics required to monitor and make better decisions on their systems
Develop a DevOps dashboard following the main guidelines
Choose a source for the data needed to populate the dashboard to mock its functionalities allowing for a better review
Interviews with experts
Submission of scientific articles
Possible Research Entry points

Figure 5- *DRSM Model followed in the research*

The principles of DSR are based on the engineering of artificial things, and information systems (IS) are a perfect example of an artificial system when the goal is to increase the organization's efficiency. Because of this reason, in Table 12, one can see how the principles were adopted. Because the principles alone are insufficient to justify the added value and consequent applicability that is useful in design science, we also adopted Hevner's DSR Guidelines[61]. Table 13 shows how this research matches the seven DSR Guidelines

Table 12- *Design science research principles*

DSR Principles	Explanation
Abstraction	The research entails the creation of dashboards to monitor the information acquired from a DevOps environment, with the goal of assisting the many stakeholders in the decision-making process. As a result, the author followed industry best practices and then investigated widely utilized techniques to further verify the concept.
Originality	Dashboards are commonly used to monitor DevOps
Justification	The dashboard's explanation is based on a solid technique for its creation and being further assessed by industry professionals.
Benefit	This dashboard aggregates various scopes of DevOps in a single instance, allowing stakeholders to make faster and better decisions.

Table 13- *Design Science Research Guidelines*

Guideline 1: Design as an Artefact
The research artifact suggested is a DevOps dashboard
Guideline 2: Problem Relevance
There is a need to create an effective dashboard that enables the monitoring of productivity, performance, and quality in a DevOps environment.
Guideline 3: Design Evaluation
Semi-structured interviews are conducted. Interviewees who must make decisions using dashboards are assessed.
Guideline 4: Research Contributions
A novel artifact that does not exist in the body of knowledge.
Guideline 5: Research Rigor
The fundamental concepts, practices, and methods of DSR were used to improve the credibility of the artifact and, as a result, the research's contribution. Stephen Few provides practical Guidelines for using color in charts. Gestalt theory and visual perception formation principles
Guideline 6: Design as a Search Process
The resulting outcome is the departure from the unknown. Combination of effective visualization methods and other relevant prototype creation criteria.
Guideline 7: Communication of Research
Furthermore, the work is intended to be submitted to a journal/conference with a high reputation and regard in the scientific community.

A questionnaire was to be used throughout the evaluation phase of the proposal. The questions that were used to aid the questionnaire are present in Table 14, and with them, it was possible to acquire the positive features, the negative aspects, and the recommended changes for further improving the dashboard. The structure of the table is shown in Tables 15 and 16. In Table 16, the advantages, cons, and suggested improvements parts are divided into four columns and three rows. The first column specifies the three above-mentioned parts. The second column is the identifier for the item. The third column is a short synthesis of what the interviewee described. The fourth column is the whole interviewee opinion from what was extracted during the interview.

Table 17 is where the suggested improvements are dissected to further improve the dashboard. It was decided to divide this table into five categories as follows, the suggested improvement, the type of improvement as it can be an improvement of information/context or visualization, next there is a column where there is a verification if the improvement was implemented, and who suggested the improvement and finally the figure where the improvement was implemented

Table 14- *Key Evaluating Questions*

Questions
What are the positive aspects of the proposed dashboard?
What are the negative aspects of the proposed dashboard?
What improvements would you suggest for the suggested dashboard?

Table 15- *Evaluation prototype structure*

	ID	Interviewee synthesis	Interviewee Opinions
Pros			
Cons			
Suggested Improvements			

Table 16-*Improvements prototype structure*

Iteration					
P1	Suggested Improvement	Type of improvement	Was it implemented	Who suggested	Figure

Chapter 4 - Proposal and Evaluation

The designed dashboard was the focus of an iterative DSR process that included three semi-structured interviews with IT professionals. Each interview produced a DSR iteration that assisted with validating, consolidating, and improving the dashboard. Table 17 lists the interviewees who participated in this procedure. The interviewees' professional expertise varies from interview to interview. Their experience, however, is mostly in the IT industry, working in a day-to-day DevOps setting.

4.1. First DSR Iteration

The prototype was improved over three iterations using the DSR technique. This was completed using mock data to build a relevant dashboard.

In this section, there is a focus on the three phases of each iteration: the proposal, the demonstration, and the evaluation.. To ensure that the mock data was not inappropriate, the interviewers were provided a context with the goal of contextualizing the dashboard and its scope with the main purpose of improving their input.

4.1.1. Proposed Dashboard

To create the dashboard, numerous processes were required, which are divided into three parts. The initial step consisted of research and analysis, followed by the process of producing the mock data, however in a production environment, a whole process of extraction, load, and transformation (ETL) would be required and ultimately enable the use of the dashboard.

Table 17- *Interviewee Data*

DSR Iteration	Age	Role	Years of experience in IT	Graduation
1	40	Solutions Architect	20	Computer Engineering
2	35	Software Lead	10	N/A
3	29	Software Developer	6	Computer Engineering

4.1.2. Research and Analysis

The research and analysis began with an evaluation of the previously implemented dashboards in a DevOps environment within the scientific community to identify a gap in the studies so that it could be addressed. And the result was that the dashboards did not adhere to the best methodology or standards for data visualization and that the metrics applied to each dashboard were very context-specific, with no discernible pattern. Prior to building the dashboard, it was necessary to conduct more research into what metrics are typically utilized in a DevOps context.

Understanding how DevOps is measured across the community would allow for a better approach to what metrics should be used in the dashboard implementation.

This metrics research led to the production of an artifact that enables understanding of which metrics are more often used and whether there is any link to the best standards for evaluating a DevOps environment. Furthermore, the metrics were categorized in order to organize how they are connected and interpreted. According to the articles analyzed, the categories chosen were productivity, quality, and performance, and this grouping has the ability to improve the way metrics are utilized in a dashboard. To move on with the dashboard implementation, it was necessary to mock the data so that it could not be tied to any specific system or product. As a consequence, any issues linked to data privacy or any other issue related to the method data is extracted are eliminated.

4.1.3. Dashboard elaboration

The dashboard's creation process began with dividing the entire collection of metrics into the three discovered metric groupings. Then after selecting which metrics were to be used in each group, there was a need to further choose the charts that were going to be used for each metric by following the guidelines used to build effective and powerful dashboards. With three distinct dashboards, there is a need to address how navigation is done between the dashboards. Therefore a homepage was built to further simplify navigation between the dashboards. And with that, the dashboard was finished.

4.1.4. Guidelines

The Gestalt principles should always be followed to improve data visualization so that the reader has an easier time interpreting the charts and making sense of the material visually [62]. Gestalt's visual perception principles aid in understanding which aspects of training are critical to information transfer and which elements are contaminants and/or enhancers. Several writers approach these ideas, which can be examined in further depth in one of the publications listed below [28].

In order to communicate through a dashboard, the use of color can be advantageous. However, there is a need to be aware of how the color is perceived because there are underlying issues in how colors can be used to communicate with an audience and with the purpose of aiding in color usage [17].

There was a need to provide some type of context for the initial iteration so that the interviewees could link the metrics presented with something they saw on a regular basis. As a result, the notion of teams was created, and the dashboard is now connected with a project where many teams operate within a DevOps environment, each with their own color. It helps interviewees to become more familiar with the data sooner and assess the usefulness of the suggested dashboards more quickly.

4.1.5. Demonstration

To show that the artefact implemented can be used in a real-world DevOps environment, the data was dressed up to make it more perceptible and not just random data, so that individuals who completed the survey could provide accurate and valuable comments.. The goal of the dashboard demonstration is to show how to navigate between dashboards and how metrics are shown in the form of charts and graphs, as well as how a simple drill-down allows for an interactive approach on dashboards.

The dashboards are divided into three groups, each of which is distinguished by one of the three metrics categories. The teams may be selected by using the filters above or by tapping on the charts to choose a team to drilldown into the appropriate team; there is also the option of filtering between time intervals to acquire a specific period for a more in-depth study. The colors are chosen on purpose to distinguish amongst the working teams so that they can be distinguished. The Gestalt Principles of Proximity, Similarity, Closing, and Connection were attempted to be included in the dashboards.

The three dashboards in Figures 6, 7, and 8 were designed with the goal of allowing users to quickly identify difficulties or problems that can be identified and comprehended in the charts and graphs, as well as pinpointing where the problem exists so that remediation may be implemented as soon as feasible.

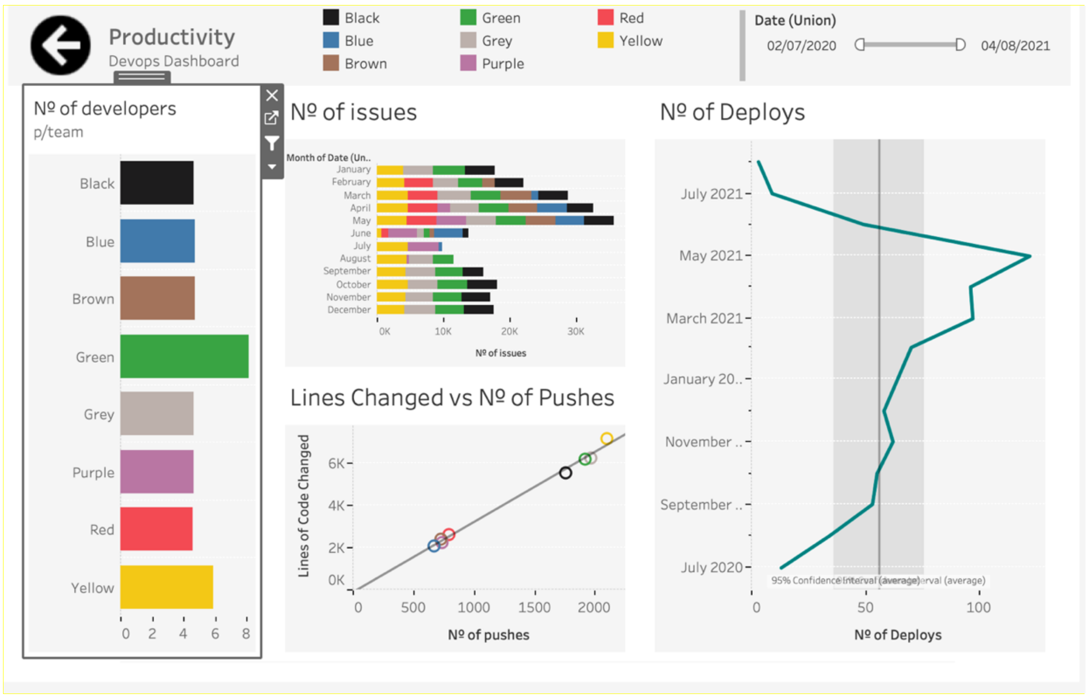


Figure 6- *Productivity Dashboard*

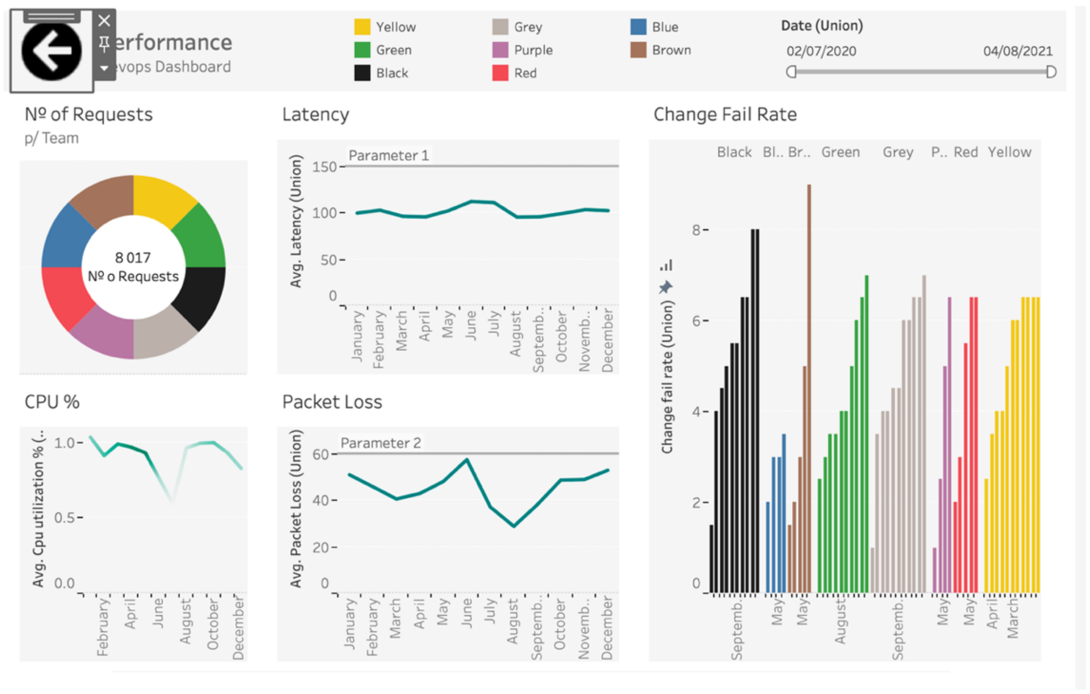


Figure 7- *Performance Dashboard*

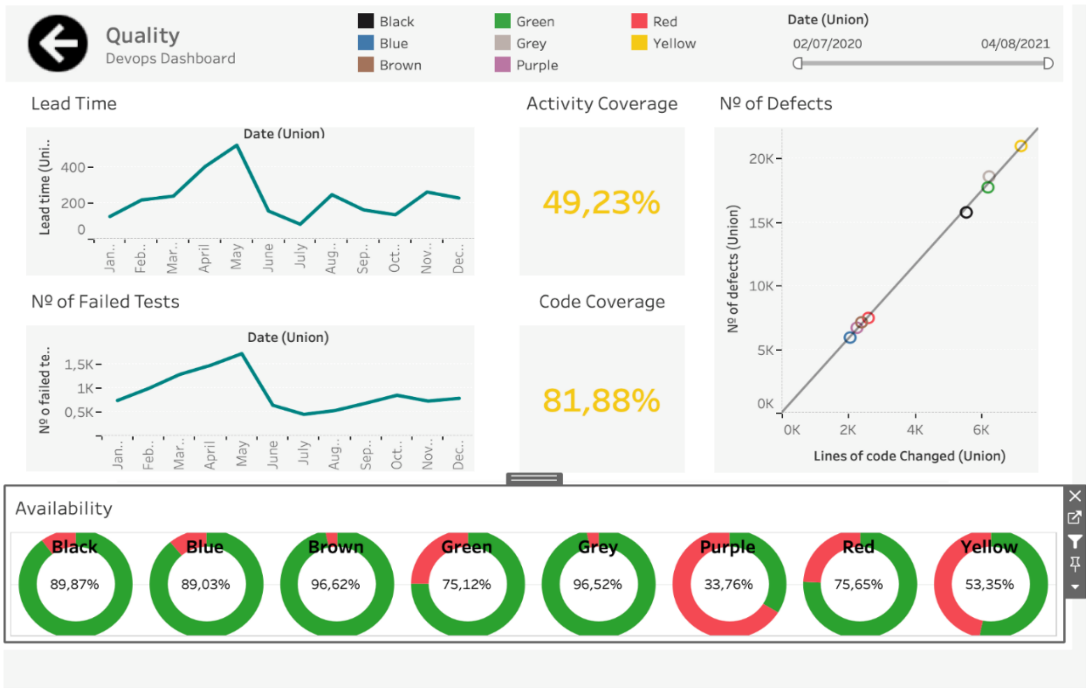


Figure 8 - *Quality Dashboard*

4.1.6. Evaluation

With the first interview, input was collected to further iterate the DevOps dashboard, the feedback gained, and it can be further analysed in Table 18 contained four positive aspects and three negative ones, resulting in three improvement recommendations, only two of which were adopted in the second iteration.

The good reviews are mostly connected to how it allows stakeholders to quickly assess DevOps and allow for more technical delivery since most deliveries do not include a comprehensive technical delivery.

Table 18- Evaluation of the prototype 1st Iteration

	ID	Stakeholder synthesis	Interviewee Opinions
Pros	P1.1	"Without monitoring DevOps, there is no clear way of validating any type of DevOps implementation."	To determine if the adoption of various technologies is meaningful, there must be quantitative evidence. Validating our pipelines requires constant monitoring of DevOps. Measuring is required. To get the most out of DevOps, follow these steps.
	P1.2	"The use of modern technologies and the simple drill down is really beneficial for the user experience."	The usage of modern technology such as Tableau or even Power BI allows for a highly current and relevant implementation. The use of drilldown provides a very user-friendly experience and interaction.
	P1.3	"I found it captivating to divide the dashboard into metric groups."	It was well thought out to use groupings to better show the metrics.
	P1.4	"This dashboard enables technical validation of team deliverables, allowing for technical delivery, which validates deliverables on a technical level."	These dashboards will allow the quality/quantity of the deliverables at a technical level since the deliverables output is generally business related.
Cons	C1.1	"There is a need to give more context in order to further relations with the dashboard"	The dashboard display lacks context, such as the construction of a story. It lacks context, which makes it difficult to comprehend the graphics.
	C1.2	"Some of the measures employed have an apparent lack of coherence."	Some indicators have no relationship with one another and so become meaningless in the absence of a connection.
	C1.3	"There isn't a need to have the filter in the top bar since you can drill down"	With the added ability to drill down there isn't a need to implement a filter to select what team to choose
Suggested Improvements	PI1.1	"Creation of a narrative to provide more perspective for the dashboard presentation."	Create a story that allows stakeholders to connect the experience of monitoring with the metrics so that they are more than simply some visuals with data.
	PI1.2	"Implement the metrics in such a way that they are more coherent."	Some indicators have no relationship with one another and so lose significance in the absence of a link.
	PI1.3	"Implementation of a homepage for the several dashboards"	Create a dashboard that provides stakeholders with a easy to use a landing page that allows selecting what team you want to drill down into.

4.2. Second DSR Iteration

After completing the first interview and analysing the comments received, all the advantages and disadvantages were used to create a better dashboard iteration.

4.2.1. Proposal

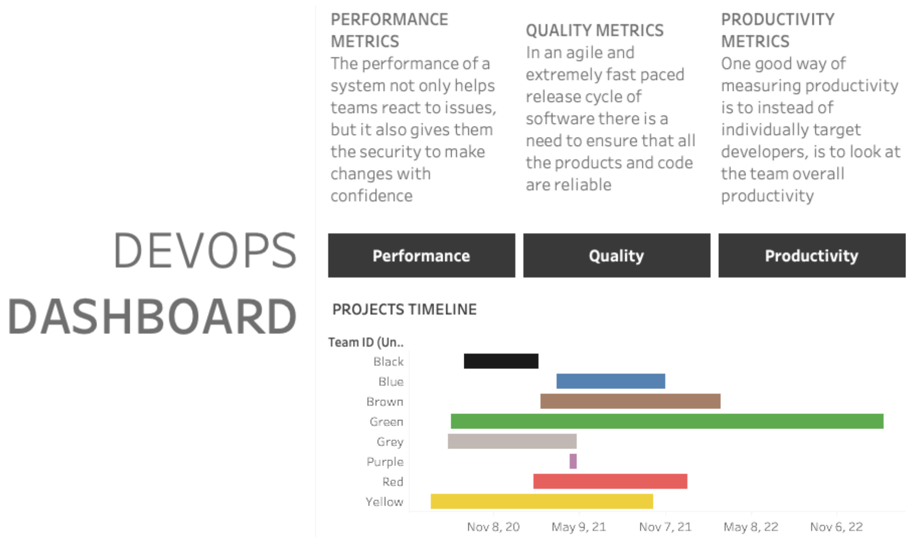
The dashboard was updated with the first interviewee's enhancement suggestions to further improve the second iteration. Table 19 presents the improvement ideas, which aids in having a consolidated perspective of the improvements. As it is feasible to validate the two enhancement recommendations that were applied during 4.2.2 Demonstration.

Table 19- Proposed improvements 1st iteration table

Iteration					
P1	**Suggested Improvement**	**Type of improvement**	**Was it implemented**	**Who suggested**	**Figure**
PI 1.1	Creation of a story so that the dashboard presentation has more context	Information/ Context	Yes	Interviewee	6 7 8
PI 1.2	Metric adjustment	Visualization	No	Interviewee	
PI 1.3	Homepage implementation	Visualization	Yes	Author	9

4.2.3. Demonstration

The primary feature in the second iteration was the construction of the main menu, from which it is possible to browse in-between dashboard implementations and have an overview of the time of all current running projects, as shown in Figure 9. The primary goal of this solution was to simplify navigation and make it extremely clear how to go from dashboard to dashboard.



PERFORMANCE METRICS
The performance of a system not only helps teams react to issues, but it also gives them the security to make changes with confidence

QUALITY METRICS
In an agile and extremely fast paced release cycle of software there is a need to ensure that all the products and code are reliable

PRODUCTIVITY METRICS
One good way of measuring productivity is to instead of individually target developers, is to look at the team overall productivity

Figure 9- *Dashboard homepage*

There was also a need to improve how the data presented on the dashboard is linked to aid the presentation so that the statistics are consistent and the interviewer narrative telling abilites are improved.

4.2.4. Evaluation

According to Table 20, the results of the second interview were two good aspects, two negative aspects, and two suggestions for improvement.

The second interview positive aspects focused on how clean and effective the dashboard can be. Still, there is also an opportunity to improve on how the metrics are used across the dashboard. There should be a focus on connecting the way the data is shown and allowing for cross-category analysis.

Table 20- Evaluation of the prototype 2nd Iteration

	ID	Stakeholder synthesis	Interviewee Opinions
Pros	P2.1	The dashboard displays basic quite core and important technical metrics	The dashboard has the benefit of presenting data from the different metric groups, which is highly useful for analysing information and making decisions.
	P2.2	This dashboard is extremely beneficial not only for people interested in technology but also for those interested in the business side of things.	It seems very relevant from a business perspective the category differentiation with a main emphasis in the productivity category
Cons	C2.1	The number of features and metrics are lacking	There is an obvious need to improve how metrics are shown on the dashboard or to try to obtain a larger picture on the dashboard.
	C2.2	It's just and overviews, could be clearly more detailed	There are methods to enhance the dashboard's experience by emphasizing the data that is being displayed.
Suggested Improvements	PI2.1	Allow for a cross analysis in between the categories	Allow for a cross-analysis of the categories. There are undoubtedly additional significant categories, but the details and cross-analysis metrics on the existing categories are the most important.
	PI2.2	It's just and overview, could be clearly more detailed	Correlate the indicators more effectively and avoid using basic numbers over time.
	PI2.3	There could be more KPIs show in the dashboard	Present more key performance indicators in a way that allows the user to quickly understand the state of the system in case

4.3. Third DSR Iteration

After completing the second interview and analysing the feedback received, all the advantages and disadvantages were used to create a better dashboard iteration.

4.3.1. Proposal

The third iteration was created with the goal of increasing the dashboard's resilience to give a more solid solution for DevOps dashboards. This third version of the concept includes the two recommended modifications. Table 21 summarizes the implemented proposals, with a complete discussion in section 4.3.2 Evaluation.

Table 21- Proposed improvements 2nd iteration

	Iteration				
PI2	**Suggested Improvement**	**Type of improvement**	**Was it implemented**	**Who suggested**	**Figure**
PI2.1	Allow for a cross-analysis in between the categories	Information /context	Yes	Interviewee	10 11 12
PI 2.2	Correlate the metrics in a better and avoid using simple numbers across time	Information / context	Yes	Interviewee	10 11 12
PI 2.3	Implement charts in a way that allows the user to read KPIs in a fast manner	Information / context	Yes	Interviewee	12

4.3.2. Demonstration

To continue with the implementation, three more dashboards were required to be able to cross-analyse the metric categories, as recommended in the PI2.1. With the inclusion of these three dashboards, the artifact became obviously more robust in its targeting of the metric categories, allowing for a more in-depth examination of the system being monitored.

Firstly, the Quality and Performance dashboard was created in Figure 10, with a strong focus on helping the end-user to rapidly identify which teams have a better overall outcome in their development by understanding how the change fail rate connects with the lead time for each team—and then being able to read how the system CPU % corresponds to the system's availability. Secondly, the Quality vs Productivity dashboard implemented Figure 11, with an emphasis on how the number of deploys in each team affects the mean time to repair, and understand which team has a problem with the software that is deployed and if it increases the time required to repair a defect. In addition, this cross-analysis dashboard allows users to watch how the number of issues corresponds to the lead time per team and check if the teams can handle the number of issues given without growing the lead time exponentially.

Finally, the Performance vs Productivity Dashboard presented in the Figure 12 was created with the goal of being able to quickly understand if the systems and teams being monitored are in a healthy state overall.

cross-analysis dashboards have a focus on providing more useful information to the end-user.

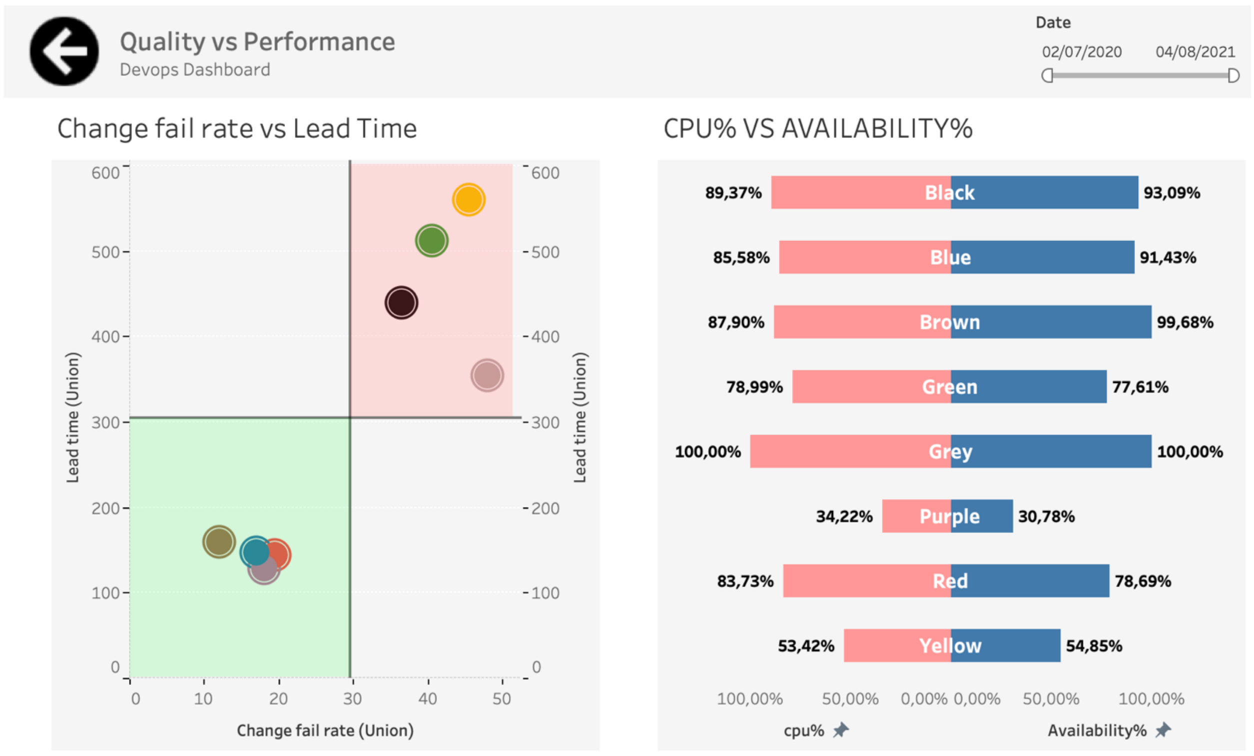


Figure 10- *Quality vs Performance Dashboard*

To achieve that quick perception, both the metrics number of crashes and the deployment frequency were approached in two different ways by utilizing a heatmap to get a perception.

Also, by utilizing Tableau calculation fields, it is possible to get a comprehensive view of how those two measures are performing over a period. In this example, you can see how the number of crashes and deploys has changed from a month or even a year ago, or you can set a target for each of these metrics. All these scopes may be selected in the Performance and Productivity left-side column.

With a focus on addressing PI2.2 and PI2.3, it can be seen in these newly implemented dashboards that there was a major focus on increasing the depth of how the metrics were being used to monitor the system and teams. By focusing less on quantity and more on detail, all these

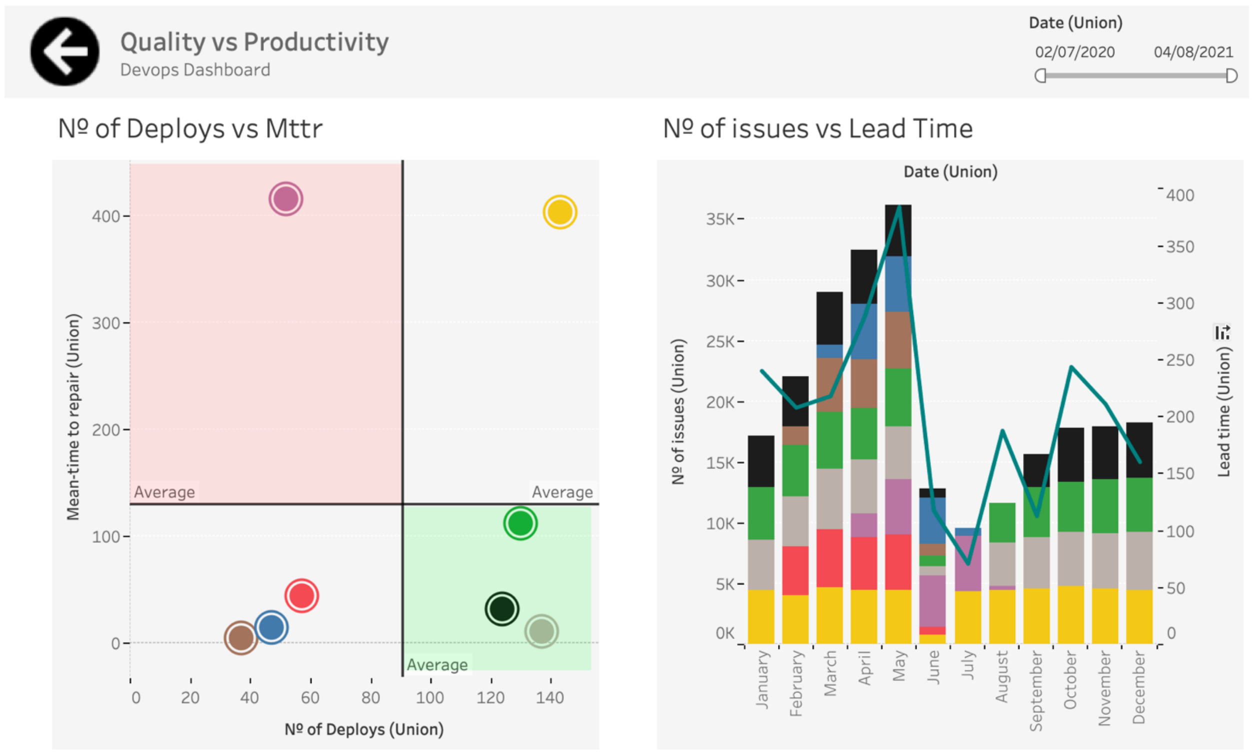
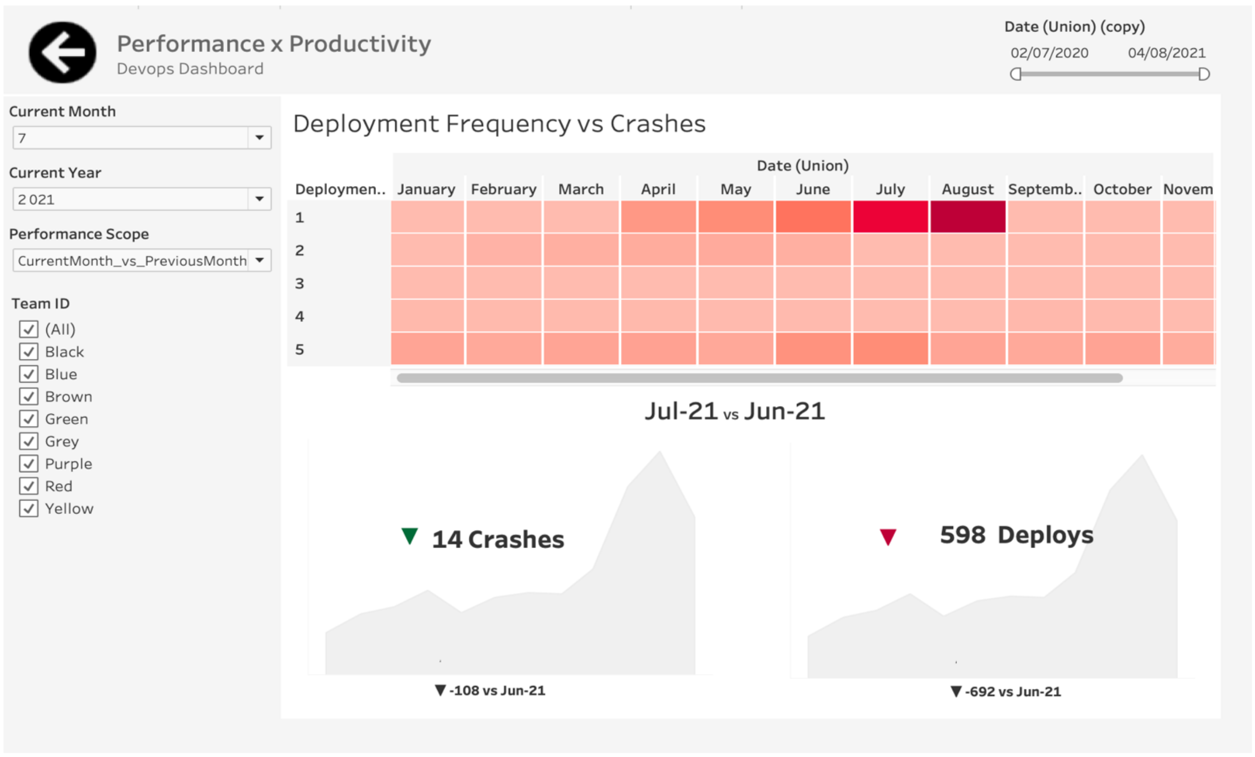


Figure 11-*Quality vs Productivity Dashboard*



Current Month: 7

Current Year: 2021

Performance Scope: CurrentMonth_vs_PreviousMonth

Team ID
- ✓ (All)
- ✓ Black
- ✓ Blue
- ✓ Brown
- ✓ Green
- ✓ Grey
- ✓ Purple
- ✓ Red
- ✓ Yellow

Figure 12- *Performance vs Productivity Dashboard*

4.3.3. Evaluation

According to Table 22, the results of the third interview were two good aspects, one bad aspect, and one suggestion for improvement.

The positive elements are mostly focused on how the dashboard has the potential to offer a more analytic approach to software development while also assisting DevOps and business teams in identifying important components of what is being delivered and how to enhance them further. And how categorizing DevOps metrics allows for easier and faster analysis.

One negative is the absence of further information about the labels, which would allow for a more in-depth explanation of the measurements employed.

Table 22 - *Evaluation of the prototype - 3rd Iteration*

	ID	Stakeholder synthesis	Interviewee Opinions
Pros	P3.1	The ability to easily infer data and make decisions based on the metrics categories	Because the data can tell you a lot about the team's performance and best practices, as well as the frequency with which they release versions and deploy their systems.
	P3.2	Simple to use and effective, the way the charts interact allows for a better degree of drill down	The user interface is straightforward to use and comprehend, and you can take a lot of conclusions from what you acquire.
	P3.3	Allow teams to see a broader picture of the software being built. Providing a more analytical picture of the product and the value of the team	It enables teams to learn from the systems, reflect internally on the systems, and have a feeling of the product that we are generating. It is indeed feasible, both as a developer and as a project manager, to measure the projects that are currently in progress.
Cons	C3.1	Some of the charts might have more specific measuring units.	Some of the units should be clarified to speed up the way metrics are displayed to the main target.
Suggested Improvements	PI3.1	Improve the way the metrics are labelled with their respective unit	There is a need to improve the way metrics and KPI are labelled

4.4. DSR Synthesis

We derived a set of improvement recommendations from the interviewees' responses, which are given in Table 23. The interviewees made 6 of the 7 improvement suggestions, while the author offered one. Five of the seven form improvement recommendations were adopted, with the remaining two being filed for future enhancements.

Table 23- *Proposed improvements by Iterations*

Iteration 1					
P1	**Suggested Improvement**	**Type of improvement**	**Was it implemented**	**Who suggested**	**Figure**
PI 1.1	Creation of a story so that the dashboard presentation has more context	Information/Context	Yes	Interviewee	6 7 8
PI 1.2	Metric adjustment	Visualization	No	Interviewee	10
PI 1.3	Homepage implementation	Visualization	Yes	Author	9
Iteration 2					
PI2.1	Allow for a cross-analysis in between the categories	Information /context	Yes	Interviewee	10 11 12
PI 2.2	Correlate the metrics in a better and avoid using simple numbers across time	Information / context	Yes	Interviewee	10 11 12
PI 2.3	Implement charts in a way that allows the user to read kpis in a fast manner	Information / context	Yes	Interviewee	12
Iteration 3					
PI 3.1	Improve the way the metrics are labelled with their respective unit	Information / context	**No**	Interviewee	

Chapter 5- Conclusion

The proposed DevOps Dashboard went through numerous revisions with the purpose of becoming a useful and resilient artifact, but it is apparent that there are other aspects that could have been targeted, such as a financial approach throughout the dashboard.

The positive aspects of the proposed artifact are that it is very intuitive and user-friendly, allowing users to analyse and easily make decisions based on the data provided in the dashboard. The ability to drill down via the team name and filters allows the end-user to be very incisive in the sort of metrics that they are looking for.

Secondly, utilizing metric groups to aggregate the way data is shown in DevOps was a method of distinguishing itself from the rest of the existing data measurement tools for DevOps. Thirdly, the Cross Analysis tabs allowed for a more in-depth comparison of how the systems are produced, providing further insight

The negative aspect is the lack of real-world data, which would allow for a more in-depth understanding and examination of how DevOps is performed in the field.

Almost all the respondents noted additional tools that supported DevOps monitoring, but they are generally overly focused on one of the categories that we identified and seldom connected to the other types of metrics that are available in a DevOps setup. They are either not particularly interactive or user friendly, necessitating the end-user to be a technical member of the team, leaving the business side in the shadows.

It is possible to infer that the suggested artifact adds value to the current collection of accessible DevOps measurement tools by allowing users to have a better user experience and presenting more than one group of relevant data. Furthermore, quantitative evidence is required to determine whether the adoption of certain technologies is meaningful. Monitoring is critical for getting the most out of DevOps.

The findings of the interviews suggest that implementing dashboards on productivity, quality, and performance is beneficial to DevOps teams and organizations. Furthermore, another key finding from the study is that the aid colour standards and Gestalt's visual principles can help to increase the value of dashboards for stakeholders.

It is also worth noting that the prototype was adapted to the interviewees' project reality where the presence of several teams working together with the purpose of improving each other in a DevOps environment, which allowed stakeholders to get more comfortable with the data and develop greater clarity about the information they want from dashboards. The deployment of the dashboards meets all the requirements for success, as evidenced by the input of the stakeholders polled.

Furthermore, the dashboard enables teams to satisfy a demand by delivering output that is normally business-related in a more technical context. These dashboards will allow the quality of the deliverables to be assessed at a technical level, enabling technical delivery, which certifies deliverables on a technological level. To conclude, the proposed dashboard would provide a macro perspective of performance, quality, and productivity. It focuses on assisting teams in understanding how their systems are doing as well as improving the end-user experience.

So far, DevOps monitoring solutions have been highly focused on very particular elements of the systems, such as deployment or code quality of the created product. The developed artifact provides teams with a tool that focuses on a larger setup while keeping to the finest standards of visualization techniques and drill-down.

5.1. Contributions

The findings of this study provide a better understanding of how DevOps implementations use metrics and dashboards, as well as a fundamental understanding that there is still much room for improvement, whether in the decision of which metrics to use, whether performance, productivity, or quality-related, or in how the dashboard itself is visually designed for the end user. Nowadays, teams must first establish their purpose and then determine what metrics they want to measure [34]. Then, focus on how they want to display and monitor those metrics in a dashboard, always with the focus being the target of the dashboard, has end-user experience is one of the major focuses of any dashboard software implementation.

The produced artifact enables stakeholders to get important information on productivity, quality and performance, as well as to give a more quantitative analysis of the systems developed by the organization, resulting in a more in-depth understanding of the product status. The dashboard took into consideration good visualization practices and drill-down techniques to aid decision making in a DevOps setting.

5.2. Limitations

The research has certain limitations. DevOps dashboards is a topic that is frequently addressed in a closed corporate setting, making it hard to really appreciate the entire community's awareness of this issue. As a result, over the course of this research, only the scientific basis was investigated. Furthermore, no real data was included in the dashboard in order to avoid the dashboard being tied to a single project and being too specific to be useful to a larger audience.

Furthermore, only three interviews were conducted; more interviews would have allowed for the collection of further dashboard-related improvements. However, these three interviews were conducted with experts that work in a DevOps setting where this sort of artifact might have a direct impact. Furthermore, owing to time constraints, it was not feasible to implement all of the suggested changes.

5.3. Future Work

As a proposal for future work, It would be interesting to implement the created dashboard in a real DevOps scenario with access to real-time data. This would allow the dashboard to be evaluated in real-world scenarios, giving the opportunity to develop the artifact further with the insights taken with the real data.

Another future work suggestion would be to provide the dashboard with a financial overview of the three metric categories, providing for a clearer understanding of how a more comprehensive system assists a team in minimizing resources used.

References

[1] N. Forsgren and M. Kersten, ‘DevOps metrics’, *Commun. ACM*, vol. 61, no. 4, pp. 44–48, Mar. 2018, doi: 10.1145/3159169.

[2] G. Casale *et al.*, ‘RADON: rational decomposition and orchestration for serverless computing’, *SICS Softw.-Intensive Cyber-Phys. Syst.*, vol. 35, no. 1–2, pp. 77–87, Aug. 2020, doi: 10.1007/s00450-019-00413-w.

[3] N. Forsgren, D. Smith, J. Humble, and J. Frazelle, ‘2019, State of DevOps’, 2019.

[4] Tricentis, ‘Software fail watch: 5th edition’, 2018.

[5] N. Forsgren, J. Humble, and G. Kim, *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*. 2018.

[6] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*, 1st ed. Addison-Wesley Professional, 2010.

[7] P. Behnamghader, R. Alfayez, K. Srisopha, and B. Boehm, ‘Towards better understanding of software quality evolution through commit-impact analysis’, Aug. 2017, pp. 251–262. doi: 10.1109/QRS.2017.36.

[8] P. Debois, ‘Agile Infrastructure and Operations: How Infra-gile are You?’, presented at the Agile 2008 Conference, 2008. doi: 10.1109/Agile.2008.42.

[9] H. H. Olsson, H. Alahyari, and J. Bosch, ‘Climbing the “Stairway to Heaven” -- A Mulitiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software’, presented at the 2012 38th Euromicro Conference on Software Engineering and Advanced Applications, Sep. 2012. doi: 10.1109/SEAA.2012.54.

[10] A. Hemon, B. Lyonnet, F. Rowe, and B. Fitzgerald, ‘From Agile to DevOps: Smart Skills and Collaborations’, 2019, doi: 10.1007/s10796-019-09905-1.

[11] A. Capizzi, S. Distefano, L. J. P. Araújo, M. Mazzara, M. Ahmad, and E. Bobrov, *Anomaly Detection in DevOps Toolchain*, vol. 12055 LNCS. 2020, p. 51. doi: 10.1007/978-3-030-39306-9_3.

[12] A. Balalaie, A. Heydarnoori, and P. Jamshidi, ‘Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture’, *IEEE Softw.*, vol. 33, no. 3, pp. 42–52, May 2016, doi: 10.1109/MS.2016.64.

[13] G. Kim, P. Debois, J. Willis, and J. Humble, *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press, 2016.

[14] J. Cito, P. Leitner, T. Fritz, and H. C. Gall, ‘The making of cloud applications: an empirical study on software development for the cloud’, 2015, pp. 393–403. doi: 10.1145/2786805.2786826.

[15] M. Traverso, M. Finkbeiner, A. Jørgensen, and L. Schneider, ‘Life Cycle Sustainability Dashboard’, *J. Ind. Ecol.*, vol. 16, no. 5, Oct. 2012, doi: 10.1111/j.1530-9290.2012.00497.x.

[16] W. Luz, G. Pinto, and R. Bonifacio, ‘Adopting DevOps in the Real World: A Theory, a Model, and a Case Study’, *J. Syst. Softw.*, vol. 157, Jul. 2019, doi: 10.1016/j.jss.2019.07.083.

[17] S. Few, *Now You See It: Simple Visualization Techniques for Quantitative Analysis*, 1st ed. Oakland, CA, USA: Analytics Press, 2009.

[18] M. Artac, T. Borovssak, E. Di Nitto, M. Guerriero, and D. A. Tamburri, ‘DevOps: Introducing Infrastructure-as-Code’, presented at the 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), May 2017. doi: 10.1109/ICSE-C.2017.162.

[19] L. Bass, I. Weber, and L. Zhu, *DevOps: A Software Architect’s Perspective*, 1st ed. Addison-Wesley Professional, 2015.

[20] A. Ravichandran, K. Taylor, and P. Waterhouse, *DevOps and Real World ROI*. Berkeley, CA: Apress, 2016, p. 150. doi: 10.1007/978-1-4842-1842-6_9.

[21] L. Bass, ‘The Software Architect and DevOps’, *IEEE Softw.*, vol. 35, no. 1, pp. 8–10, Jan. 2018, doi: 10.1109/MS.2017.4541051.

[22] ‘State of DevOps report 2018’.

[23] W. Gerald, *Quality Software Management: Systems Thinking*. 1991.

[24] ISO/IEC 9126, ‘Software Engineering—Software Product Quality. Part 1: Quality Model, ISO/IEC 9126-1, Geneva, Switzerland’, 2001.

[25] P. Perera, R. Silva, and I. Perera, ‘Improve software quality through practicing DevOps’, presented at the 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer), Sep. 2017. doi: 10.1109/ICTER.2017.8257807.

[26] R. Y. Al-Jaar, ‘Book review: The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling by Raj Jain (John Wiley & Sons 1991)’, *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 19, no. 2, Sep. 1991, doi: 10.1145/122564.1045495.
[27] A. Brunnert *et al.*, ‘Performance-oriented DevOps: A Research Agenda’, *ArXiv*, vol. abs/1508.04752, 2015.
[28] S. Few, ‘Common pitfalls in dashboard design’, 2006.
[29] B. Kitchenham, ‘Procedures for Performing Systematic Reviews’, 2004.
[30] ‘State of DevOps report 2017’.
[31] C. Castellanos, C. A. Varela, and D. Correal, ‘ACCORDANT: A domain specific-model and DevOps approach for big data analytics architectures’, *J. Syst. Softw.*, vol. 172, 2021, doi: 10.1016/j.jss.2020.110869.
[32] S. Dalla Palma, D. Di Nucci, F. Palomba, and D. A. Tamburri, ‘Within-Project Defect Prediction of Infrastructure-as-Code Using Product and Process Metrics’, *IEEE Trans. Softw. Eng.*, pp. 1–1, 2021, doi: 10.1109/TSE.2021.3051492.
[33] K. Maroukian and S. R. Gulliver, ‘The Link Between Transformational and Servant Leadership in DevOps-Oriented Organizations’, in *Proceedings of the 2020 European Symposium on Software Engineering*, Rome Italy, Nov. 2020, pp. 21–29. doi: 10.1145/3393822.3432340.
[34] Z. Ding, J. Chen, and W. Shang, ‘Towards the Use of the Readily Available Tests from the Release Pipeline as Performance Tests. Are We There Yet?’, in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, Oct. 2020, pp. 1435–1446.
[35] M. A. López-Peña, J. Díaz, J. E. Pérez, and H. Humanes, ‘DevOps for IoT Systems: Fast and Continuous Monitoring Feedback of System Availability’, *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10695–10707, Oct. 2020, doi: 10.1109/JIOT.2020.3012763.
[36] E. Kristiani, C.-T. Yang, C.-Y. Huang, Y.-T. Wang, and P.-C. Ko, ‘The Implementation of a Cloud-Edge Computing Architecture Using OpenStack and Kubernetes for Air Quality Monitoring Application’, *Mob. Netw. Appl.*, 2020, doi: 10.1007/s11036-020-01620-5.
[37] M. Oriol *et al.*, ‘Data-driven and tool-supported elicitation of quality requirements in agile companies’, *Softw. Qual. J.*, vol. 28, pp. 931–963, 2020, doi: 10.1007/s11219-020-09509-y.
[38] T. J. Wagner and T. C. Ford, ‘Metrics to Meet Security & Privacy Requirements with Agile Software Development Methods in a Regulated Environment’, Feb. 2020, pp. 17–23. doi: 10.1109/ICNC47757.2020.9049681.
[39] T. Zheng *et al.*, ‘SmartVM: a SLA-aware microservice deployment framework’, vol. 22, pp. 275–293, 2019, doi: 10.1007/s11280-018-0562-5.
[40] L. Diogo Couto, · Peter, W. V. Tran-Jørgensen, · René, S. Nilsson, and G. Larsen, ‘Enabling continuous integration in a formal methods setting’, *Int. J. Softw. Tools Technol. Transf.*, 2019, doi: 10.1007/s10009-019-00546-y.
[41] Storey Margaret-Anne and C. Treude, ‘Software Engineering Dashboards: Types, Risks, and Future’, in *Rethinking Productivity in Software Engineering*, T. Sadowski Caitlin and Zimmermann, Ed. Berkeley, CA: Apress, 2019, pp. 179–190. doi: 10.1007/978-1-4842-4221-6_16.
[42] M. Barisits *et al.*, ‘Rucio: Scientific Data Management’, *Comput. Softw. Big Sci.*, vol. 3, no. 1, pp. 11–11, Dec. 2019, doi: 10.1007/s41781-019-0026-3.
[43] I. Dr Dragan *et al.*, ‘A Scalable Platform for Monitoring Data Intensive Applications’, *J Grid Comput.*, vol. 17, pp. 503–528, 2019, doi: 10.1007/s10723-019-09483-1.
[44] F. Trautsch, S. Herbold, P. Makedonski, and J. Grabowski, ‘Addressing problems with replicability and validity of repository mining studies through a smart data platform’, *Empir. Softw. Eng.*, vol. 23, no. 2, pp. 1036–1083, Apr. 2018, doi: 10.1007/s10664-017-9537-x.
[45] N. Alshahwan *et al.*, ‘Deploying search based software engineering with sapienz at facebook’, Sep. 2018, vol. 11036 LNCS, pp. 3–45. doi: 10.1007/978-3-319-99241-9_1.
[46] N. Herbst, ‘Quantifying Cloud Performance and Dependability’, *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 3, no. 4, pp. 1–36, Sep. 2018, doi: 10.1145/3236332.
[47] B. Snyder and B. Curtis, ‘Using Analytics to Guide Improvement during an Agile–DevOps Transformation’, *IEEE Softw.*, vol. 35, no. 1, pp. 78–83, Jan. 2018, doi: 10.1109/MS.2017.4541032.
[48] N. Asha and P. Mani, ‘Knowledge-based acceptance test driven agile approach for quality software development’, *Int. J. Recent Technol. Eng.*, vol. 7, no. 4, pp. 196–202, 2018.

[49] A. Rahman, ‘Characteristics of defective infrastructure as code scripts in DevOps’, May 2018, pp. 476–479. doi: 10.1145/3183440.3183452.
[50] D. A. Tamburri, M. M. Bersani, R. Mirandola, and G. Pea, ‘DevOps Service Observability By-Design: Experimenting with Model-View-Controller’, 2018, pp. 49–64. doi: 10.1007/978-3-319-99819-0_4.
[51] A. Rahman, J. Stallings, and L. Williams, ‘Defect prediction metrics for infrastructure as code scripts in DevOps’, May 2018, pp. 414–415. doi: 10.1145/3183440.3195034.
[52] H. Huijgens, R. Lamping, D. Stevens, H. Rothengatter, G. Gousios, and D. Romano, ‘Strong agile metrics: mining log data to determine predictive power of software metrics for continuous delivery teams’, 2017, pp. 866–871. doi: 10.1145/3106237.3117779.
[53] J. Cito, F. Oliveira, P. Leitner, P. Nagpurkar, and H. C. Gall, ‘Context-based analytics - Establishing explicit links between runtime traces and source code’, 2017, pp. 193–202. doi: 10.1109/ICSE-SEIP.2017.1.
[54] M. G. Jaatun, D. S. Cruzes, and J. Luna, ‘DevOps for Better Software Security in the Cloud Invited Paper’, Aug. 2017, pp. 1–6. doi: 10.1145/3098954.3103172.
[55] Puppet and Dora, ‘State of DevOps report 2016’.
[56] D. Sun, M. Fu, L. Zhu, G. Li, and Q. Lu, ‘Non-Intrusive Anomaly Detection With Streaming Performance Metrics and Logs for DevOps in Public Clouds: A Case Study in AWS’, *IEEE Trans. Emerg. Top. Comput.*, vol. 4, no. 2, pp. 278–289, Apr. 2016, doi: 10.1109/TETC.2016.2520883.
[57] R. Meissner and K. Junghanns, ‘Using DevOps principles to continuously monitor RDF data quality’, 2016, vol. 13-14-Sept, pp. 189–192. doi: 10.1145/2993318.2993351.
[58] E. Birngruber, P. Forai, and A. Zauner, ‘Total recall: Holistic metrics for broad systems performance and user experience visibility in a data-intensive computing environment’, presented at the Proceedings of HUST 2015: 2nd International Workshop on HPC User Support Tools - Held in conjunction with SC 2015: The International Conference for High Performance Computing, Networking, Storage and Analysis, 2015. doi: 10.1145/2834996.2835001.
[59] A. G. L. Romme, ‘Making a Difference: Organization as Design’, *Organ. Sci.*, vol. 14, no. 5, Oct. 2003, doi: 10.1287/orsc.14.5.558.16769.
[60] March and Storey, ‘Design Science in the Information Systems Discipline: An Introduction to the Special Issue on Design Science Research’, *MIS Q.*, vol. 32, no. 4, 2008, doi: 10.2307/25148869.
[61] A. R. Hevner, S. T. March, J. Park, and S. Ram, ‘Design Science in Information Systems Research’, *MIS Q.*, vol. 28, no. 1, pp. 75–105, 2004, doi: 10.2307/25148625.
[62] D. Chang and K. V. Nesbitt, ‘Developing Gestalt-Based Design Guidelines for Multi-Sensory Displays’, AUS, 2006, pp. 9–16.

Appendices

Appendix A – Questionaire - 1st Iteration

Question 1	Do you consider this dashboard relevant?
	Yes
Question 2	Question If you answered yes, why you consider the dashboard important in this matter?
	To determine if the adoption of various technologies is meaningful, there must be quantitative evidence. Validating our pipelines requires constant monitoring of Devops. Measuring is required. To get the most out of DevOps.
Question 3	Is the devops dashboard complete (Yes/No)?
	No
Question 4	If you answered no, what do you think is missing?
	The dashboard display lacks context, such as the construction of a story. It lacks context, which makes it difficult to comprehend the graphics.
Question 5	Identify the positive aspects of the dashboard?
	The usage of modern technology such as tableau or even power bi allows for a highly current and relevant implementation, and the use of drilldown provides for a very user-friendly experience and interaction.
Question 6	Identify the negative aspects of the dashboard?
	Some indicators have no relationship with one another and so become meaningless in the absence of a connection.
Question 7	Compared to other used devops dashboard/monitoring tabs, what are the positive aspects?
	It was well thought out to use groupings to better show the metrics.
Question 8	Compared to other used devops dashboard/monitoring tabs, what are the Negative aspects?
	None
Question 9	Do you think the implementation of this dashboard is an added value?
	Yes
Question 10	If so, could you justify?
	Yes, since these dashboards are what will keep the quality/quantity of the deliverables at a technical level since the deliverables' output is generally business related. Allowing for technical delivery, which validates deliverables on a technical level

Appendix B – Questionaire - 2nd Iteration

Question 1	Do you consider this dashboard relevant?
	Yes
Question 2	Question If you answered yes, why you consider the dashboard important in this matter?
	Because it displays basic quite core and important technical metrics
Question 3	Is the devops dashboard complete (Yes/No)?
	As an overview dashboard yes, but it could also be more detailed
Question 4	If you answered no, what do you think is missing?
	There certainly are other important categories but most importantly details and cross analysis metrics on the current categories
Question 5	Identify the positive aspects of the dashboard?
	Clean, straight to the point
Question 6	Identify the negative aspects of the dashboard?
	It’s just and overview, could be clearly more detailed
Question 7	Compared to other used devops dashboard/monitoring tabs, what are the positive aspects?
	It seems more relevant from a business perspective the productivity category
Question 8	Compared to other used devops dashboard/monitoring tabs, what are the Negative aspects?
	The number of features/metrics are lacking
Question 9	Do you think the implementation of this dashboard is an added value?
	It depends on the cost of the implementation.
Question 10	If so, could you justify?
	From a business perspective I would have to always analyse the competitors and understand how it would pane out with the competition

Appendix C – Questionaire - 3rd Iteration

Question 1	Do you consider this dashboard relevant?
	Yes
Question 2	Question If you answered yes, why you consider the dashboard important in this matter?
	Because the data can tell you a lot about the team's performance / best practices, as well as the frequency with which they release versions/deploys.
Question 3	Is the devops dashboard complete (Yes/No)?
	Yes
Question 4	If you answered no, what do you think is missing?
	The KPI data has to be more explicit
Question 5	Identify the positive aspects of the dashboard?
	It's easy to interact with the interface, and it's straightforward to understand, so it's possible to remove a lot of the ilaciones we're dealing with.
Question 6	Identify the negative aspects of the dashboard?
	Measurements (Units) should be made easier in order to enhance reading speed.
Question 7	Compared to other used devops dashboard/monitoring tabs, what are the positive aspects?
	It is very intuitive and easy to navigate and acesss
Question 8	Compared to other used devops dashboard/monitoring tabs, what are the Negative aspects?
	N/A
Question 9	Do you think the implementation of this dashboard is an added value?
	Yes
Question 10	If so, could you justify?
	It enables teams to learn from the systems, reflect internally on the systems, and have a feeling of the product that we are generating. It is indeed feasible, both as a developer and as a project manager, to measure the projects that are currently in progress.