Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Máster Universitario en Innovación digital
Master's Programme in Digital Innovation

Trabajo Fin de Máster
Master Thesis

Adopción de Metodologías DevOps para la Inteligencia de Negocio

Business Intelligence Adoption of DevOps Methodologies

Autor / Author: Guillermo Antoñanzas Martínez

Madrid, Marzo 2022

Este Trabajo Fin de Máster se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid.

This Master Thesis has been deposited in ETSI Informáticos de la Universidad Politécnica de Madrid.

Trabajo Fin de Máster
Master Thesis
Máster Universitario en Innovación digital
Master's Programme in Digital Innovation

Título: Adopción de Metodologías DevOps para la Inteligencia de Negocio

Title: Business Intelligence Adoption of DevOps Methodologies

March 2022

Autor / Author: Guillermo Antoñanzas Martinez

Supervisor:
prof. dr. Jurgen Vinju, j.j.vinju@tue.nl
Mathematics and Computer Science, TUe

Co-supervisor:
Lina Ochoa Venegas, l.m.ochoa.venegas@tue.n
Mathematics and Computer Science, TUe

Abstract

For this Master Thesis, we want to know what is the state of the Business Intelligence (BI) tools in the Development and Operation (DevOps) maturity level scale. To the best of our knowledge, no analysis like this exists in the state of the art. Moreover, the communication between the DevOps teams is fundamental for any project and it is even more relevant to those that keep growing, like how BI projects tend to be.

To this aim, we will take an in-depth look into the most relevant and most used BI tools and the features they have, with reference to the DevOps models and its maturity levels, as well as some additional features that are of interest to the community (e.g. license, activity, maintainability). We will analyze their characteristics in the DevOps categories and obtain the theoretical maximum maturity levels for each tool to show the gaps that the BI tools currently have. Multiple DevOps Maturity Models will be used to widen the scope of the analysis and have a better representation of how tools implement DevOps.

Additionally, we will analyze real cases of BI projects from the internship company, NTTData, to see if the theoretical maximum maturity levels are reached and propose changes to improve those levels.

Acknowledgements

This Master Project has been the result of many months of hard work and effort and the last part of my Master's program in the EIT. Despite taking longer than anticipated, it has been an exciting journey and has been a very interesting project where I have learned a lot about many different topics. It has been difficult personally due to multiple reasons, mainly due to mental health issues, so I would like to take this moment to emphasize how important it is to take care of yourself, especially in this aspect. But we cannot forget that we are not alone and that it is essential to talk to other people and ask for help if necessary. That said, there are many people with whom I could not have continued if it had not been for them.

First and foremost, my thesis tutor, Lina Ochoa. She helped me and gave me feedback for the project, and supported me throughout this whole journey. Also, my thesis supervisor, Prof.dr. Jurgen Vinju, who guided me with the direction to take for this project. I would also like to thank all the members and colleagues from the TUe and EIT who have been there, helping me in many ways, and being by my side, giving me the encouragement I needed to finish this project.

Finally, I would like to thank my family and friends for the support and love they have given me all the time, especially through the worse moments of these last months.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Summary

In the last decades, we have seen an increase in the popularity of Development and Operation (DevOps) methodologies as a new take on agile for software development projects and Business Intelligence (BI) as a critical approach to many businesses for optimization and improvement of their operations. However, we have not seen any studies or articles carried out in academia that relate both of them.

This lack of interest exists because DevOps adoption has not been popularized in BI contexts, as DevOps models are designed around code-oriented projects. The improvements and ideas behind this philosophy have the most potential. These projects commonly present extensive development cycles and an ever-increasing complexity (in terms of data volume, lines of code, and internal processes) that tends to cause problems in the long run. Unless properly addressed, these issues can lead to inefficient projects, and the use of DevOps methodologies is one solution. It provides a framework to start improving upon what is already built, where speed and reliability are the spotlights, and where the maturity level can get measured to obtain how well the adoption process is going.

However, BI projects do not have any specific programming language, instead consisting of proprietary scripting code for the front-end visualizations and data modeling, with query-based languages and sometimes Python or R scripts for the data extraction and transformation. They are complex systems with different components for each task with no set structure. When more than one of these components is handled by the same software, that combination is considered a BI tool.

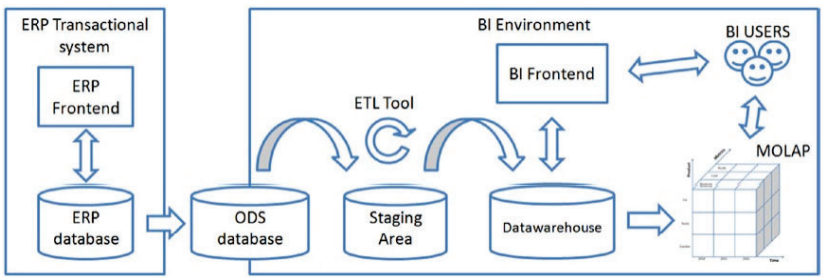


Figure 1.1: BI System components diagram from [1]

The changes proposed by many of the DevOps models publicly available and in agile development, in general, are flexible to the nature of the project and the technologies being. However, it assumes that the project carries out specific tasks not always present for most BI projects. In our experience, it is a challenge to adapt the proposed guidelines developed for the previously mentioned "code-oriented" projects. The software and the infrastructure it is supported on are inherently different, as shown in Figures 1.1 and 1.2 We have observed that, in comparison to other non-BI projects, BI software lacks some if not many of the features their counterparts have that DevOps prioritizes to improve the project's performance.

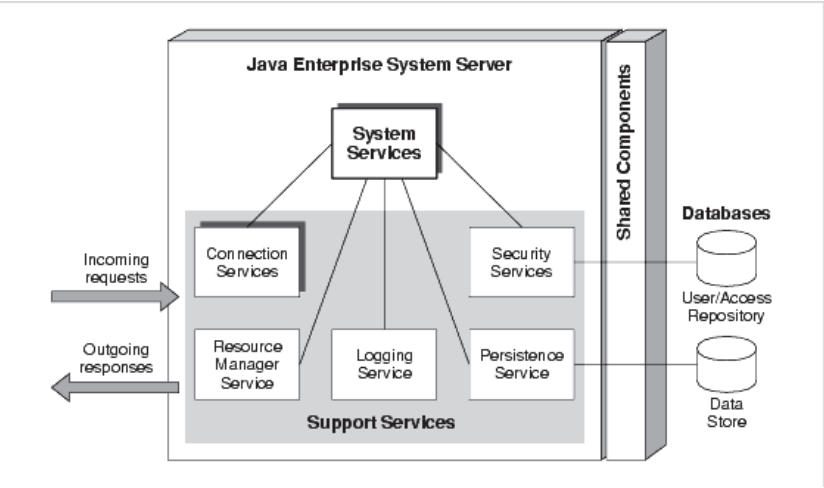


Figure 1.2: Java System diagram.

We would assume that the difference in BI projects' development nature is why agile and DevOps are not as widely adopted, hence the lack of academic publications. Still, we would need to prove that theory is true first. Are this sector's programs and tools prepared to implement DevOps's changes? Or are they dragging behind in comparison to other markets? And, have they been keeping the features up to date and adapted to meet those standards with time?

1.2 Problem Statement

With BI growing more and more each year and already existing projects increasing in size to the point where its speed and reliability start to decrease, adopting measures to tackle this issue becomes mandatory. Here, DevOps is one solution that has already been adopted and shown positive results for other types of technologies and developments. It has resulted in better performance than before, where even continuous integration and deployment cycles occur. However, DevOps models have not been designed for BI projects, so their implementation is complex. From what we have investigated, barely any publications showing the adoption process or discussing the results obtained when applying these models to existing BI projects.

Considering this lack of publications for the combination of BI and DevOps, an excellent research opportunity arises. This study will delve into the adoption and preparation of the current BI software solution market for these kinds of agile methodologies, more specifically

for DevOps Maturity Models (DOMMs). Now that the objective is clear, we can start with the central question for this thesis:

RQ *How are BI projects currently implementing DevOps?*

Since DevOps models are designed with maturity models as frameworks, we use these DOMMs as the base for comparison for preparing the BI tools for its implementation. For this purpose, we are creating theoretical representations of various DOMMs that take into account only the aspects that can be informed or measured at the theoretical level. These representations are explained in more detail in Chapter 4, where we present some of the issues and design decisions faced at this point.

For the theoretical representation of the adoption of DevOps methodologies by BI projects, we are using a selection of BI tools representative of the market. Since the same software can be used in many companies and projects, we focus on that aspect of the project's maturity. More specifically, we focus on the technical characteristics of the software as parameters for comparison with the maturity models. With both, we can make a theoretical representation of the current state, allowing us to identify possible areas of improvement and gaps in the tools. The tool analysis and the results are presented in Chapter 5, where we carry out a Systematic Tool Review and explain the process for selecting the BI tools used for the study to represent the current market.

However, as this is a theoretical study, it is necessary to validate these results. We have proposed a validation procedure in which we represent the maturity of real BI projects and observe if there is the same level of preparation in real cases compared to the theoretical results. To perform this representation, a dashboard is developed, where all the previous information is modeled and visualized. This procedure is detailed in Chapter 6.

All the software and output files obtained in this master project will be available to download in this Zenodo repository [2].

Each of the following sub-questions is answered in one chapter:

SQ1 Which are the state-of-the-art DevOps Maturity Models? [Chapter 4]

SQ2 Which BI tools are available in the market and how they fit in DOMMs? [Chapter 5]

SQ3 How can we model implementation of DevOps methodologies in BI projects? [Chapter 6]

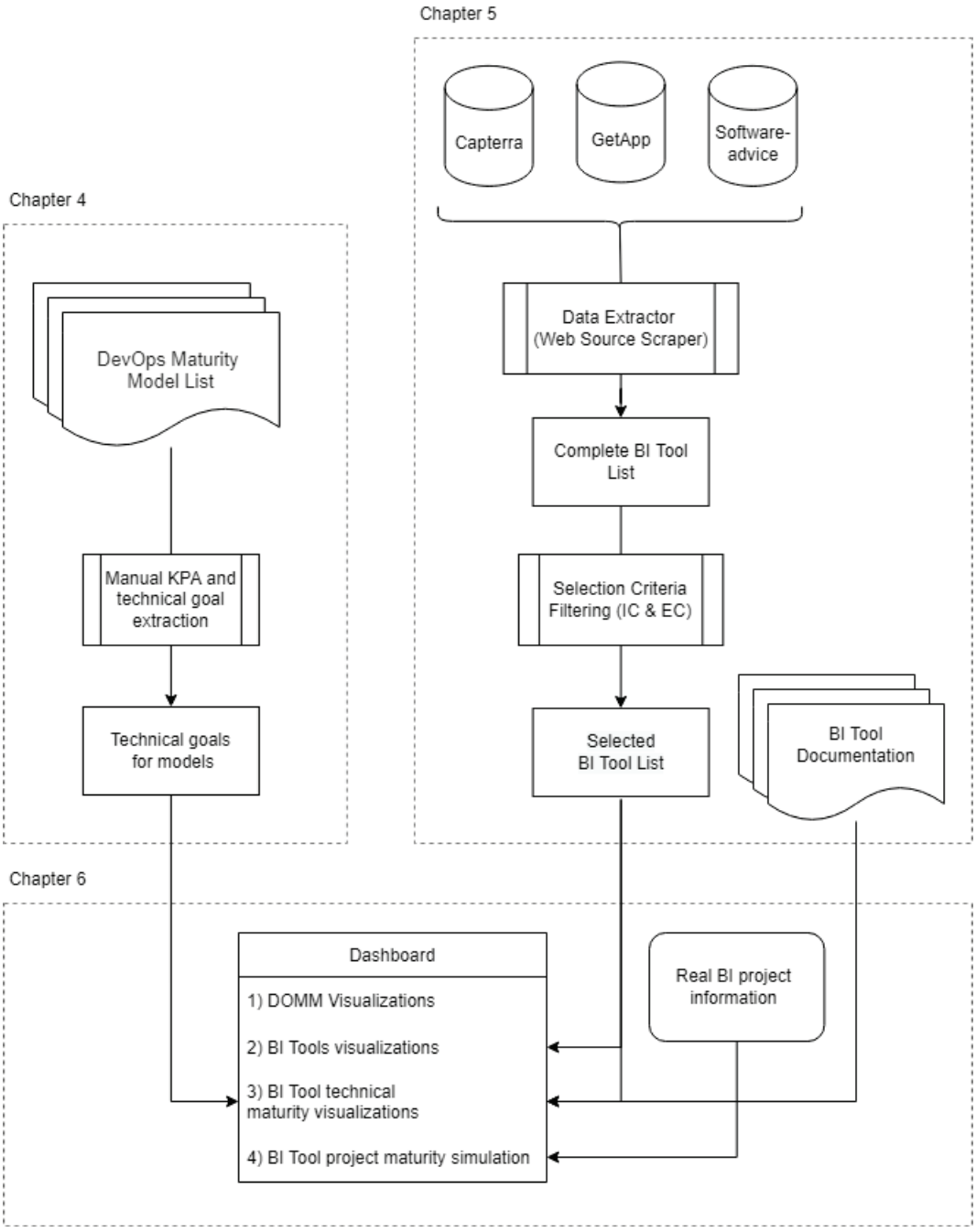


Figure 1.3: Overview of the chapters and information flow.

1.3 Thesis Structure

This master thesis will be structured in the following chapters:

- Chapter 2 delves into the BI and DevOps topics to provide the reader with a quick view of both and lay down some standard terms used in the industry and research communities.
- Chapter 3 talks about related work to these topics.
- Chapter 4 begins the research by looking at the different existing DevOps models and extracting their objectives for the analysis.
- Chapter 5 shows the tool review for the current BI tool market and the results obtained when analyzing them in the context of the maturity model's goals from the previous chapter.
- Chapter 6 details the task carried out to validate the formal theory and presents the dashboard created in this project.
- Chapter 7 concludes with the report and the main takeaways and presents the future work that needs to be done.

Chapter 2

Background

This chapter briefly introduces the two main concepts this master thesis revolves around. The first part will be a quick look at what DevOps is, what maturity models are, and; how DevOps models are used to represent the maturity of the projects, and what areas should be improved—which are the main subjects of Chapter 4. The second part will explain what Business Intelligence is, its components, and its structures.

2.1 DevOps

In the last decade, DevOps has gained popularity in the software engineering community [3] as one of the options for agile methodologies. But, what is it exactly?

2.1.1 What is DevOps?

DevOps is a methodology formed by the combination of cultural philosophies, practices, and tools that focus on improving an organization's processes to increase the speed, quality, and capacity of teams and their projects. [4]

From our research, we have not found a specific definition, yet a simplified view of the core idea would be the union of development and operations teams to achieve the previously mentioned goal. The development team consists of software developers that focus on creating the functional aspects of the project through software. The operations team comprises experts who maintain various software elements in a production environment, such as database administrators and network specialists [5].

Since no formal definition nor baseline exists with a standard set of actions or implementations and organizations' needs and characteristics are different, carrying out a DevOps transformation from scratch is complicated. This lack of a formal concept is considered one of the biggest challenges in the industry [6].

In turn, various DevOps models exist, each defining a specific set of goals and cultural changes that drive the teams adopting said model to focus on the aspects to be improved. These models also have differences in their structure and implementation. Still, they usually focus on four distinct areas of interest that are always present in an organization: *technology*, *processes*, *people*, and *culture*. In Figure 2.1 we can see a simplified representation of these models. The ideas proposed in DevOps, as a methodology, are what constitutes the basic concepts of what this model will try to achieve. The maturity model is used as the framework

to represent those ideas, but as we will see in the next section, different design characteristics exist specific to each model. Other unique components to the model will be the combination of individual practices, tools, and cultural philosophies proposed by each instance and the level they can apply it.

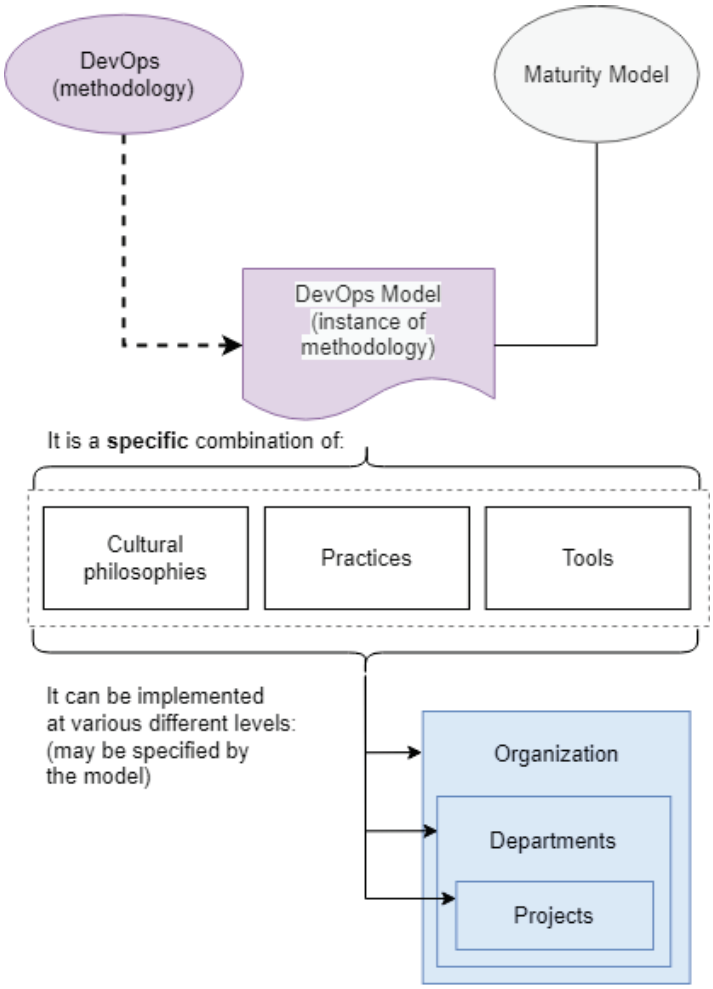


Figure 2.1: Simplified DOMM representation

2.1.2 What is a Maturity Model?

For several decades now, maturity models have been successful means for improving software-intensive organizations [7]. Different models have been developed, changing and improving in certain areas, such as Business Process Management [8] or Cloud Computing [9]. Their purpose is "to give guidance through this evolutionary process by incorporating formality into the improvement activities" [10].

If we look at Figure 2.2, we can start differentiating maturity models by two types: "focus area" and "fixed level" models. The former lists a number of focus areas that need to be developed to reach maturity in a functional domain. A series of progressively mature capabilities is provided for each focus area. Maturity level scales are individual to each focus area, opposite to fixed level maturity models. These models often follow the maturity levels as

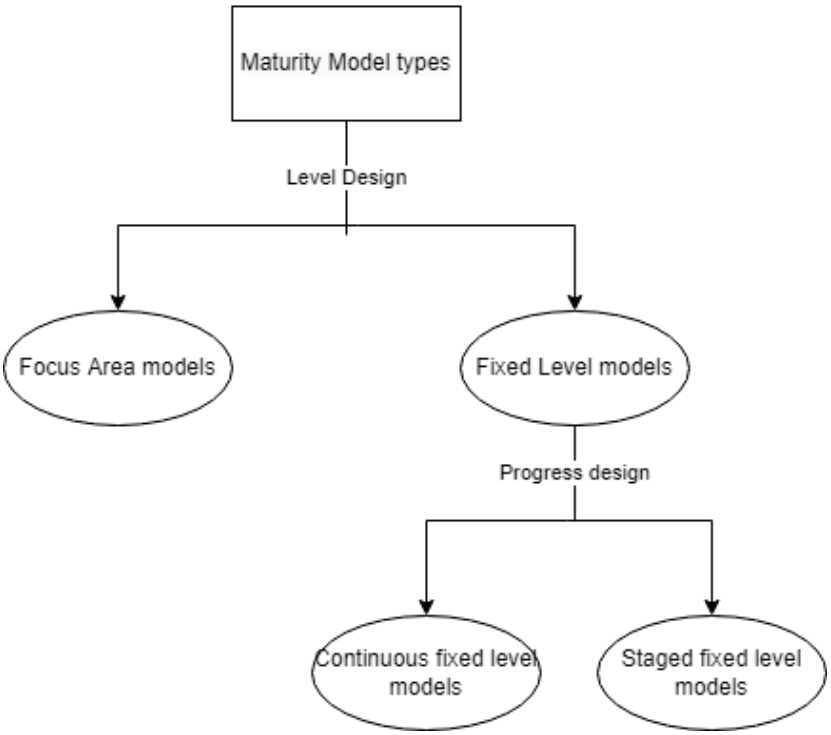


Figure 2.2: A simplified view of Maturity Model design options.

described in Capability Maturity Models (CMM) [11], which is divided into four or five levels, where each one represents an improvement over the last one. They are named to represent how mature that level is or the goals that should be met. Since fixed level maturity models are the most common type of models in DevOps, they are the focus of this explanation and study. If the model is somewhat complex, it is divided into sections, where a specific area or part of the model is described. This is useful to identify sections with lower maturity than others. The specific terms change often, but the ideas behind them, are the same. For example, some models refer to these *sections* as *areas*.

As we can see in Figure 2.3, in CMM each maturity level contains some Key Process Areas (KPA). They identify the issues that must be addressed to achieve a maturity level. Each key process area identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important for enhancing process capability. CMM also details other components such as Common features and Key Practices, but we will not detail them as they are specific to the CMM.

Depending on the restrictions of the model to achieve the maturity for every level, we can distinguish between "staged maturity models" and "continuous maturity models". All KPAs and its goals need to be in place for staged maturity models to achieve a certain maturity level. In contrast, continuous models can score maturity at different levels allowing for a more gradual and varying improvement path. [10]. The difference can be easily seen with an example: if the KPAs for level 1 and 3 are mature enough (its goals are met) but those for level 2 are not, a staged maturity model will rank its overall maturity as 1, where the continuous maturity model will set it as 3. However, not all components are inherited from the CMM, and slight variations may exist for each individual instance of a model.

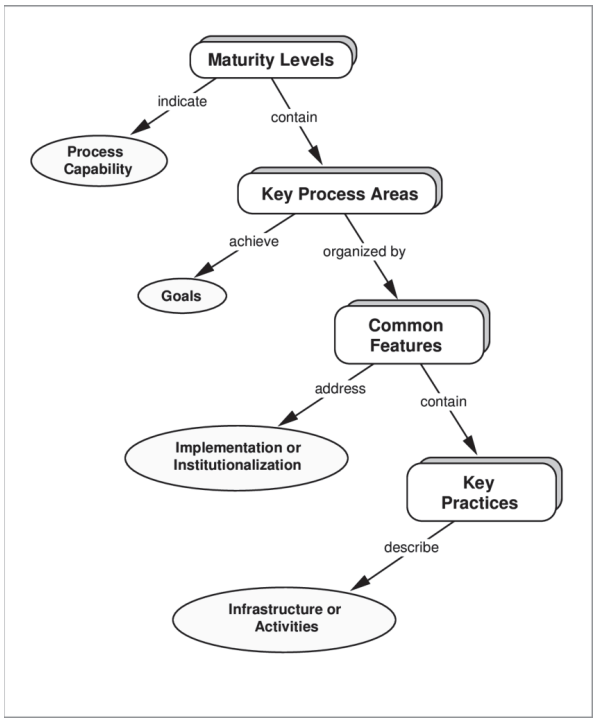


Figure 2.3: "The CMM Structure", from Capability Maturity Model [11]

2.1.3 DevOps Maturity Models

Maturity models are used to guide their users in the process of improving their projects, showing the current maturity levels, and indicating which aspects need to be addressed, making it a perfect instrument for DevOps to adapt. DevOps Maturity Models (DOMMs for short) are more commonly found as "staged fixed level" maturity models, but there are instances of "focus area" and "continuous fixed level" models. An example of this is presented in better detail in Chapter 4, Table 4.2. Another important element in maturity models specific to DevOps implementations are "dimensions" , groups containing various sections that share characteristics in common. Dimensions may not be present in all models, but they are more common in those with numerous sections to help define the "theme" they share.

Level \ Dim.	High Order Dimension (D1)			
	Section 1 (S1)	Section 2 (S2)	...	Section m (Sm)
Level 1	KPA S1L1	KPA S2L1		KPA SmL1
Level 2	KPA S1L2	KPA S2L2		KPA SmL2
...				...
Level n	KPA S1Ln	KPA S2Ln	...	KPASmLn

Table 2.1: Example of the Structure of a DevOps Model.

One of the usual forms of representation for these models is a table, similar to Table 2.1, showing the different levels, sections, and KPAs.

Project maturity for staged fixed level models is represented most of the time by showing the maturity levels with colors or symbols for each KPA, providing direct visualization of its current status (Figure 2.4). This format can be made more complex if a higher level of detail needs to be visualized.

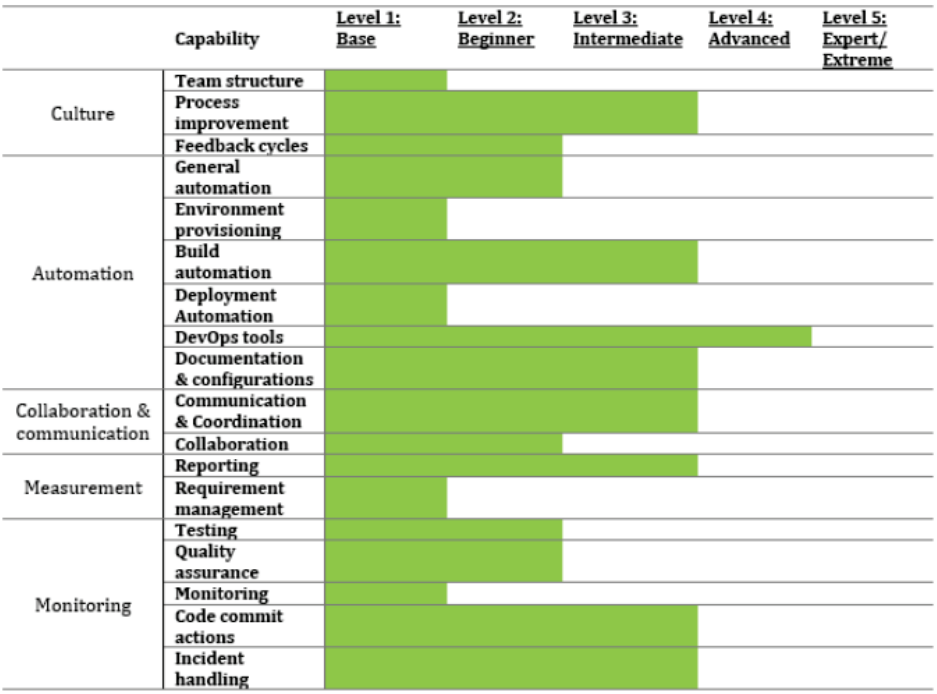


Figure 2.4: Example of Radstaak's Maturity Assessment [12].

2.2 Business Intelligence

The concept of Business Intelligence (BI) was used for the first time as a common name for describing "concepts and methodologies for the improvement of business decisions using facts and information from supporting systems" as described by H.Dresner in 1989 [13]. Nowadays, BI is seen as the combination of business analytics, data mining, data visualization, data tools and infrastructure, and best practices to help organizations make more data-driven decisions. It should be considered as an umbrella term that covers the processes and methods of collecting, storing, and analyzing data from business operations or activities to optimize performance [14]. When projects in companies start focusing on these specific concepts and ideas, it is considered a BI project.

To begin executing the BI processes mentioned before, projects start using different tools that take care of the various aspects. In this thesis, we refer to them as "BI tools"; specific software products that cover most if not all of those needs. These tools usually cover three main categories [15],[16]: data management, data discovery, and reporting. Data management focuses on the organization, transformation, and structuring of the input data. Data discovery supports the analysis and understanding of the data, and it allows us to come to meaningful conclusions with different methods. Reporting is the data visualization, offering a more understandable format to convey the analysis results.

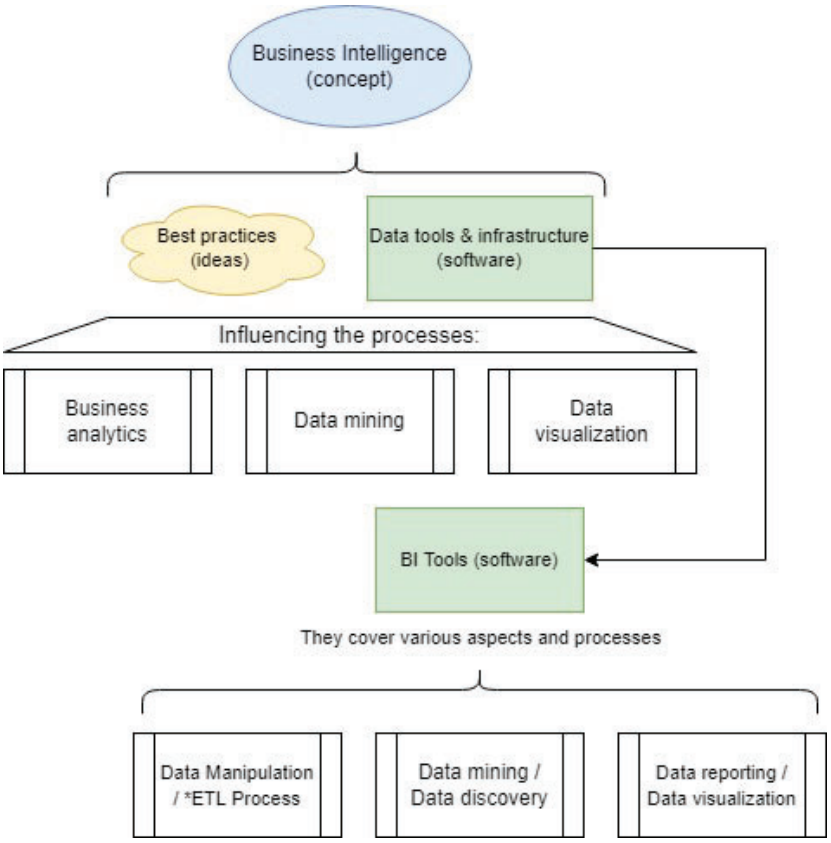


Figure 2.5: Diagram representing BI's concepts.

These three categories are the core components of what we use to consider a tool to be BI-specific. In our opinion, tools may not cover all the categories, and they will need other tools to be used in BI projects. However, if a piece of software focuses solely on one category, we will consider that an application designed specifically for that matter, not a BI tool.

BI tools aim to extract knowledge from data while focusing on three concepts: reliability, availability, and attractive user experience [1]. The first characteristic is a necessity in all IT projects, but it is even more significant in BI tools since the results presented by the tool are used directly to make critical decisions. It is mandatory to offer consistency in the data shown to ensure the correct results based on the data quality. Availability is a crucial pillar that is also related to the business and its users since they must be able to access it during working hours. It has to be stable and up to their requirements, especially when dealing with time-sensitive data. This aspect in particular may not look as relevant as the others, but in our experience as well as in other experts' opinion, Availability is a recurring problem in the day to day of BI projects. The last one is very relevant since the user may not be proficient with the tool being used, something more common the higher the user's responsibility for that company. The data must be user-friendly and adapted to consumer expectations and capacity, which usually ties to the reporting aspects of the tool we mentioned before.

Both reliability and availability align with DevOps's improvements, making promising potential results when implementing the methodology into BI projects. This is reinforced further by the fact that the work done by the development team is often linked directly with

various aspects that operations are in charge of, like database administration and maintaining the production environment running, suggesting an easier implementation of DevOps in BI projects.

Now that we have defined all the relevant concepts, we can see the common areas where BI projects and DevOps models coincide. For the rest of the thesis, we minimize the scope in which DOMMs are implemented to focus solely on the projects. This will help us better measure the impact of the changes proposed by the maturity models to what will be specifically BI, which is the BI tools and processes.

As mentioned before, when implementing a DOMM, changes are done to four areas of interest: people, culture, technology, and processes. For BI projects, these areas of interest are correlated directly with some of its components: project team members as people, regardless of whether they belong to the development or operations team; BI's best practices as culture; data tools & infrastructure as technology, which includes BI tools; and BI specific processes (business analytics, data mining, data visualization) as processes. These correlations are represented in Figure 2.6. However, this is a simplified view of how BI is directly affected by DevOps, as the culture changes, for example, reach far more into the project than just the "best practices" set by BI.

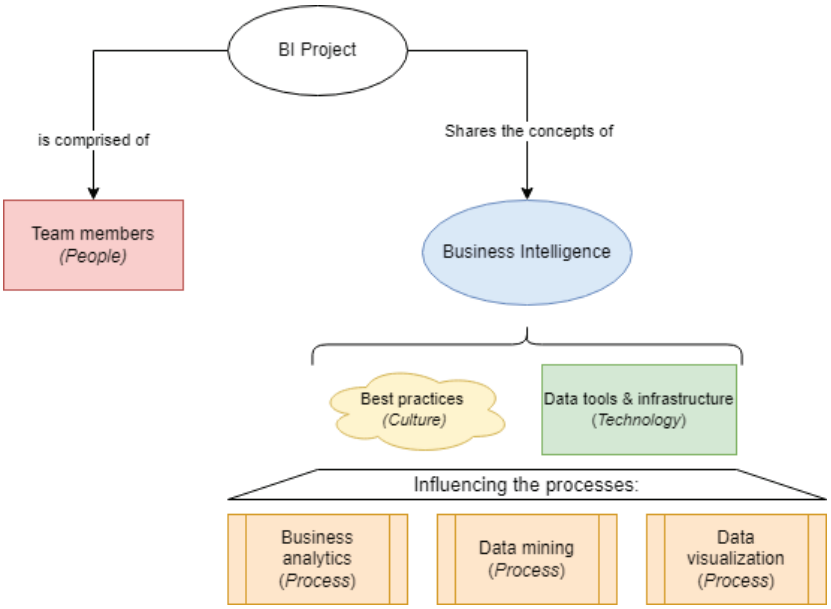


Figure 2.6: Simplified view of a BI project, its different areas and DevOps influences.

Chapter 3

Related Work

This chapter details our findings regarding the state of DevOps models, maturity models, and BI, especially in the academic context. We look at other articles and papers and explain what important insights they provide us with, what aspects we defer from, and why.

3.1 DevOps Model Studies

One of the most critical elements for developing this thesis is the existence of other publications for the same topics discussed in this project. These articles provided us with significant amounts of information that had already been validated, helping us answer questions that we would otherwise have to check by ourselves. An excellent example is the "Systematic Literature Review for DevOps Models" performed by M. Gasparaité et al. [17] and the "Developing a DevOps Maturity Models" paper by M. Radstaak [12].

Gasparaité's systematic literature review was a fundamental part of our study since the analysis of the maturity models performed in Chapter 4 made it necessary to conduct this academic analysis to obtain a list of DOMMs. Not only had this task already been completed by M. Gasparaite et al., but it had an in-depth look at the structure and characteristics of the models at hand, which also helped us speed up the analysis process for the models we decided to analyze. Another remark about this paper would be that the year of publication of the models is not considered in the quality assessment, a possible issue when reviewing the publications that were the foundation for what DOMMs came to be.

Another study that we read and used for reference to our study was the master thesis of M. Radstaak, "Developing a DevOps Maturity Model" [12]. Here, he researches and executes a detailed design process to create a DOMM based on existing maturity models. The design focuses not only on DevOps but also on Continuous Delivery (CD), one process present in highly mature DevOps projects. One of the results from this project was the creation and validation of a maturity model that, funnily enough, appears in the previously mentioned Systematic Literature Review (SLR). In one of the chapters, the author also performs a systematic literature review on maturity models, including gray literature, which led him to find and consider CD models, something we are not interested in. The search terms used for his review process also seemed to provide fewer results, probably because they were more strict than those used in Gasparaité's study.

Besides the last two publications mentioned, most studies done on DOMMs have a very simplistic description of the process to obtain the models themselves, with no way of replic-

ating it or validating the sources used. However, they do contain interesting analyses of the models being studied and results that can validate the efficacy of maturity models in the real world.

One of these articles is "DevOps and organisational performance: the fallacy of chasing maturity" by C. Marnewick et al. [18], which focuses on studying DevOps maturity's impact on organisational performance. The list of maturity models that the authors have shown here is similar to that of the other two systematic reviews, but in this one, we do not have a detailed view of the sources used nor the process taken to obtain these models. That being said, the research methodology for the case study followed to understand the effects of DevOps maturity models in a real institution is appropriately documented. Not only that, but they explain the results obtained and reach fascinating conclusions regarding the variety of maturity models and how they are used. It is clarified that organizations are better off creating their own models based on the various available models. We used this insight as a focal point when developing our dashboard to represent the maturity models that we studied and create that functionality to test.

We find other instances of papers performing analysis on DevOps models with one big issue: the name used to describe each model. At face value, this does not seem like a problem, but duplicates start to appear when the authors take these names as the unique identifier for these models. This issue was a problematic result for this study [19] since three out of the eleven maturity models they studied are duplicates of other models, threatening the study's validity, especially when the results of the analysis showed different information for said duplicates. This article reinforces the importance of a clear and systematic approach to these review procedures, such as B. Kitchenham's [20], which we followed in Chapter 5 for the SLR.

3.2 BI and Tool Analysis

For this thesis, the BI aspect is the most difficult one to tackle. We are carrying out the analysis of the BI tools available in a systematic manner, something that has not been carried out and published yet. The reason behind this problem is pretty straightforward: BI as a whole is very distant from academia and more close to business-related areas, leading to fewer studies being performed about it. Not only that but the software that is commonly used tends to require a paid license or is very rudimentary when it does not have a paywall behind it. The lack of features may be caused by low support for those products or because "freemium" versions of popular tools usually restrict the most interesting features to the paying user. Even open-source software for BI is scarce and provides subpar functionalities when comparing it to similar commercial options, being described by other studies as "having a long way to go" [21].

The studies that we find that various evaluated tools only focus on specific ones or decide to select only a few ones, mainly looking at the BI-specific features of said tool [21] [22]. These analyses were in-depth but lacked the volume to reference them in our studies. That being said, we take note of some ideas and executions described for this project, such as differentiating between open source and commercial tools and using Gartner's "Magic Quadrant for BI Platforms Report" as a starting point for finding tool information.

For our study, we need the opposite; being able to review the tools that are the most representative of the current market and obtaining the relevant information from them to

carry out the analysis of the state that these tools are in. To do so, we performed a Systematic Review, similar to the procedures of a Literature review but with slight changes, since the sources for our study were not going to contain academic literature. The process followed in "A Systematic Literature Review of Software Product Line Management Tools" [23] is our starting point. Still, as mentioned before, we decided to change the sources used to obtain our primary studies, leading us to a high volume of results that could not have been achieved if the input data had come from academic literature.

One article we find handy for this project and anyone interested in BI is A. Nogués and J. Valladares' book on "Business Intelligence Tools for Small Companies" [1], where they describe in detail many concepts and explain the reasoning behind them. Chapter 2 is of much use to us for the focus on Agile methodologies in Business Intelligence, highlighting the importance it can have in many cases and the benefits it provides when adequately adopted. However, we have to mention the lack of DevOps as a disappointment for us, especially seeing the levels of detail put behind every chapter in the book. Still, since the Agile Methodology recommendation chapter is based on the personal considerations of the authors, that does not mean that the ideas proposed in this book are better than other Agile alternatives.

Chapter 4

DevOps Models & Features

4.1 Summary

This chapter addresses the first sub-question **SQ1** regarding the methods to analyze the DevOps Maturity Models (DOMMs). After that, we continue with the review of the models listed in the Systematic Literature Review performed by M. Gasparaite et al [17] and the subsequent analysis of its results to determine which models are valid for our research. Then, it is followed by the transformation of the goals of each model into technical and process metrics, which prepares the model's goals to be used in future chapters to carry out the research analysis of the BI tools.

4.2 Introduction

This chapter aims to obtain the information needed about DevOps models and begin the analysis. Before doing so, a few aspects need to be addressed to know how to proceed:

How do we represent the current state of DevOps models?

Before starting this thesis, we assumed that there would exist a standard model designed to be used in any project and backed up by many companies using it. However, as explained in Chapter 2, DevOps does not have an unequivocal definition. We could only find projects that attempt to design these kinds of DOMMs with uncertain degrees of success [24].

To identify existing DOMMs and used them as reference, a SLR on the DevOps models is necessary. Since this study has already been carried out by M. Gasparaite et al [17], we use their results as baseline. The list of models we obtain from their publication will be a starting point, but a more extensive analysis of the characteristics, descriptions, and goals will be carried out. With our final list of maturity models, we will have a representative sample of the existing DevOps models.

What characteristics are we going to analyze for each model?

Since the existing models are not homogeneous in many different aspects, it is important to set a baseline for the structure we will use. We begin by analyzing those models with a similar representation and then adapting them to fit into this "baseline" structure. Once we have the list of models adapted, we can normalize the key aspects of each one of the different

sections and KPAs of each one of them. This process will be done to find the shared goals in between models and lay the grounds needed to continue the analysis for Chapter 5, where these metrics and characteristics will be the different measures to be observed for each tool.

4.3 Literature Review Analysis and DOMM Selection

This part of the project started with the intention to carry out a Systematic Literature Review or a similar kind of study by ourselves to obtain a list of relevant DOMMs. However, this was deemed pointless once we found that M. Gasparaite et al. [17] had already published a SLR regarding DOMMs, which helped list those relevant to our study.

We have to mention an important point before we continue with their result analysis: they added a Quality Criteria (QC) section (section 3.4 in their publication) in their study. This QC was applied in addition to the previous inclusion and exclusion criteria, which "is used to assess the quality of the studies that were selected ...". For their study, selecting only models that reach high scores is needed and allows them to narrow down the list of models to take into account. However, for us, the limitation for only using those DOMMs that achieved high levels of quality would mean that we are discarding many viable options and not properly representing the state of the current DevOps models. For this reason, we have decided to lower the point score used in Gasparaité's study to consider a larger number of papers to be analyzed and used in our investigation.

Study	Short Name	Year of publication	Score
[25]	Feijter	2017	4
[26]	Bucena	2017	3
[27]	Mohamed	2015	3
[12]	Radstaak	2019	3
[28]	IBM	2013	2
[29]	HP	2013	2
[30]	CapGemini	2015	2
[31]	Poelwijk	2016	2

Table 4.1: Results of [17]'s Quality assessment we considered suitable for further analysis

As we can see in Table 4.1, with the quality score threshold lowered by one point, we have doubled the list of models, but we still need to look into why the quality scored lower and consider if they are going to be useful in our study. Looking into Table 4 in [17], the reason that lowers the result for all of these models is the same: they are not based on scientific papers. For the authors, the reason behind this was that "papers that refer to other scientific papers are more likely to be relevant and suitable for further investigation. The same goes for the models that are referred to by other papers."

However, one aspect that was not taken into account when looking into this criteria was the year of publication. This detail is significant since both IBM and HP models (2013) were the first to be published from this list, meaning that the foundation for DOMMs was non-existent and led them to be the pioneers in this area. Adding to the fact that they are two models referenced in multiple other publications, this leads us to consider them for further analysis. For the other two instances, CapGemini (2015) and Poelwijk (2016), they could be

considered early cases of maturity model publications. Still, they were only found in academic sources as cited studies, being Google the data source for this study (Table 3 in [17]). For this reason, more detail will be put into their analysis when deciding to add them to the final list of maturity models.

Names	Representation	Nº of levels	Process Areas
Feijter	Focus Area	11	16
Bucena	Staged	5	4
Mohamed	Staged	5	4
Radstaak	Continuous	5	18
IBM	Staged	4	4
HP	Staged	5	3
CapGemini	Staged	5	3
Poelwijk	Staged	4	4

Table 4.2: Representation and details for the selected Maturity Models.

Once validated this list of models to analyze from M. Gasparaite et al. [17], an in-depth look into the structure, definition, and components of the models must be executed to decide which one to use for reference in the next steps project.

To do so, we are going to select only those models that have a staged fixed-level design. This filter is put in place to ensure that each model's representation is bound to a similar structure, the previously mentioned "baseline". Continuous and focus area models behave differently when calculating the maturity, which puts aside both Feijter and Radstaak.

For the rest of the models, we did a detailed analysis of their characteristics and the descriptions for each level and KPA. Here we found Mohamed's model has described the different goals without much detail, making it difficult to detail them with specific terms and parameters, an essential step for the next two sub-questions. This problem also arose with Poelwijk's Model, not only when looking at the goals themselves but also with the level definitions set in place, defining the first one as "initial level", where no progress has been achieved because of the lack of requirements, making it broader by definition. Taking this into account with our previous statements, we decided also to drop Poelwijk's model as well as Mohamed's.

Regarding HP's Model, the source link used in the SLR redirected to a broken page, and any other reference to it that we found in different publications all had the same issue. We had some information about this model from other publications that analyzed it, but none with a sufficient level of detail for our analysis. Since one of the defined criteria from M. Gasparaite et al.'s work was to exclude sources with partial availability; we haven't included this model in our final selection.

After removing all the previously mentioned models, the final list of DOMMs is the following:

- I. Bucena, M. Kirikova's Proposed DevOps Maturity model [26]
- IBM's DevOps maturity model [28]
- Capgemini's DevOps Maturity Model [30]

4.4 Model Transformation and Analysis:

As explained in Chapter 1, since we are carrying out a theoretical analysis the only sections and KPAs that will be of use will be those that represent the technological advancements for the project. From that list of technical section and KPAs, we will extract a list of specific goals for each maturity level to find any common objectives between each other. Then we simplify those goals into a set of common features to analyze the project's capabilities on a theoretical level.

That set of characteristics and processes will be then used in Chapter 5 as the features that need to be informed to; then carry out the theoretical analysis of the BI tools to report their "maximum maturity level" and conclude this aspect of this project.

- Bucena:

  For Bucena's DOMM we can clearly distinguish between those areas intended more to guide the users and practices (*Process, Culture & People*) rather than the technological aspects of the tools (*Technology*), especially since Table 2 of this model (Included in Appendix section A.1) represents the relevant practices in a per-goal basis for the technology area. The "Process" section also includes multiple KPAs and goals related to "testing", "delivery" and "deployment", and since those are some of the key aspects in DOMM, we are selecting it too.

	TECN				PROC		
	T1	T2	...	T9	PR1	...	PR6
Level 1	Environments are provisioned manually	Manual tests or minimal automation		No tools or minimal tool usage for issue tracking	Inconsistent delivery process		Uncontrolled or reactive processes (not applied management)
Level 2	All environment configurations are externalized and versioned	Functional test automation		All issue and bug reports are tracked	Scheduled delivery process		Processes are managed, but not standardized
...							
Level 5	Environment provisioning fully automated	Chaos Monkey		Continuous delivery process	Development process integrated with Six sigma		Highly optimized & integrated processes

Figure 4.1: Simplified representation of Bucena's model.

- IBM:

  The way IBM's model was designed focused on highlighting for each section the most important KPAs and goals. The section "*Plan/Measure*" is centered around practical project metrics, like "requirements" or "portfolio management". The other three sections however note more technical aspects of the projects to take into account, such as "code", "tests" and "deployment".

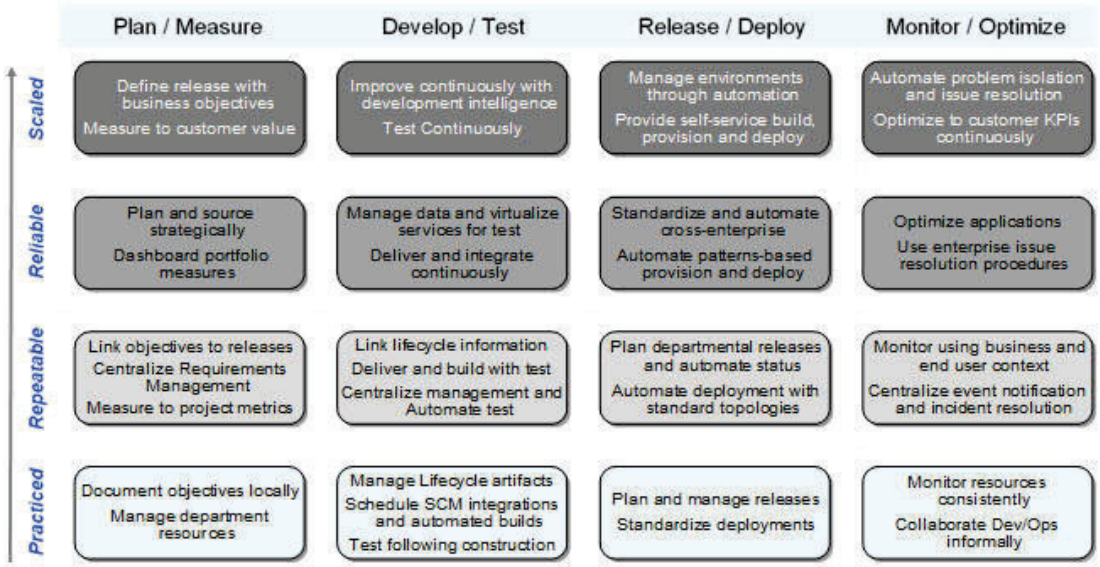


Figure 4.2: IBM's Maturity Model [28]

- CapGemini:

  This case is very similar to that one of Bucena's: it has a clear differentiation between *People, Processes,* and *Tools*; but some of the key changes are present in the Process area, leading us to select both of these sections for the next stage.

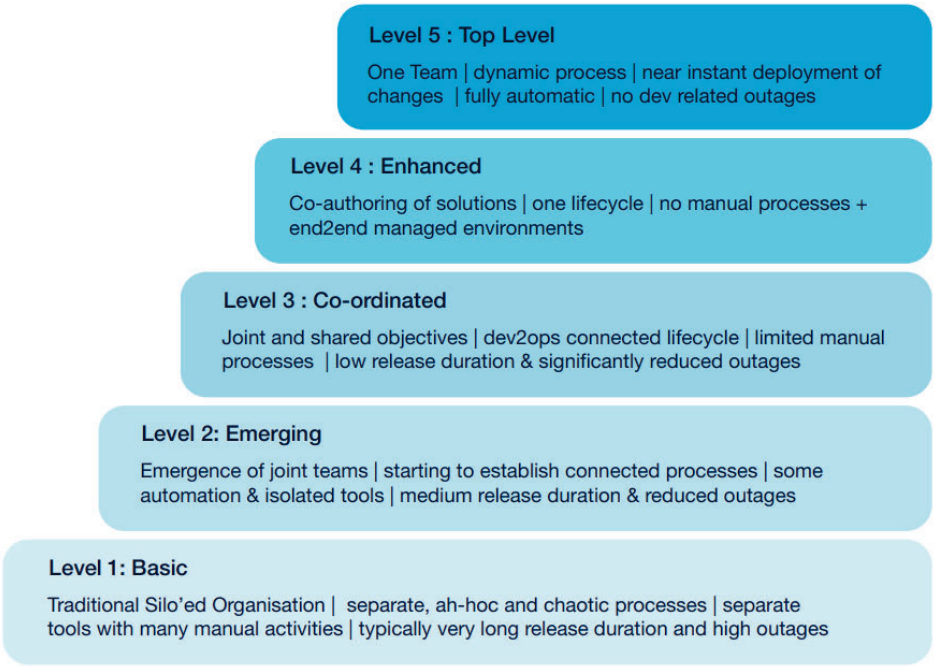


Figure 4.3: CamGemini's Maturity Model [30]

Model	Selected sections (Technical)	Non-selected sections (Practical)
Bucena	Process Technology	People Culture
IBM	Develop/Test Release/Deploy Monitor/Optimize	Plan/Measure
Capgemini	Tools Process	People

Table 4.3: Selected sections for the goal analysis.

Now that we have in Table 4.3 the list of sections to analyze, we begin the manual process of goal extraction and transformation for each and every one of them. To do so, we read through the descriptions and ideas for every KPA to summarize them into core concepts, sometimes creating more than one for each level's goal.

With that information, we proceeded to create tabular representations of the models, the sections, and the different KPAs and goals per level they may have. At this point we realized that the goals also described some of the requisites as process improvements rather than strictly technical, meaning that we had to also separate those objectives into two groups: technical goals and process goals.

Technical goals are directly correlated to the technical aspects of the project, like the functionality provided by a piece of software. One example is Version Control, where this goal can only be met by the technical functionality provided by a program. Process goals on the other hand are more "task-oriented". Two good examples would be Agile development or regular documentation validation, as they are goals reached by actively performing said tasks or processes. The separation of these two types of goals helps us focus on the theoretical aspects of these models.

Model	Section	Maturity Level	Tech. Goal Count	Proc. Goal Count
IBM	Develop/Test	1.Practiced	2	2
		2. Repeatable	3	1
		3. Reliable	4	0
		4. Scaled	1	1
	Release/Deploy	1. Practiced	1	1
		2. Repeatable	1	2
		3. Reliable	2	1
		4. Scaled	2	0
	Monitor/Optimise	1. Practiced	1	1
		2. Repeatable	2	1
		3. Reliable	0	2
		4. Scaled	2	1

CapGemini	Process	1. Basic	1	0
		2. Emerging	0	1
		3. Coordinated	0	1
		4. Enhanced	0	1
		5. Top Level	2	0
	Tools	1. Basic	2	1
		2. Emerging	4	3
		3. Coordinated	3	2
		4. Enhanced	2	3
		5. Top Level	4	1
Bucena	Process	1. Initial	1	5
		2. Repeatable	1	6
		3. Defined	1	5
		4. Managed	1	6
		5. Optimized	1	3
	Technology	1. Initial	2	7
		2. Repeatable	7	3
		3. Defined	5	6
		4. Managed	3	6
		5. Optimized	2	2

Table 4.4: Overview of the number and type of goals per level and section.

Once that separation was completed, we grouped into various specific characteristics the goals listed, unifying and simplifying the different objectives. We obtained a total of 28 standardized features that covered all the levels and areas to analyze related to the technical aspects measured for each maturity model. The complete list can be found in the Annex section A with the names and descriptions of each one of them.

In the next chapter, we proceed to obtain our selection of BI tools and for each one of them carry out the analysis and study of the technical features we have just described, setting up the grounds to finish the comparison between the models and tools.

Chapter 5

BI Tools

5.1 Summary

This chapter presents a Systematic Literature Review (SLR) following the structure and guidelines proposed in Kitchenham et al. [20] within the context of BI tools available in today's market. We have adapted and applied said techniques to identify those tools that represent the software used in this industry. Since this is done by selecting data sources outside academia and no literature is analyzed, we refer to this as "Systematic Tool Review" (STR). Because the structure of our work remains unchanged, this section consists of three main phases: planning, conducting, and reporting.

In the planning stage, we define the need for our review, what questions we address, and the review protocol to follow.

In the conducting phase, we define our data sources and the selection of primary studies. After that, we perform the review and obtain the final list of studies, showing the different processes and filters the information extracted has gone through. The data extraction and transformation process is also explained in this chapter.

In the reporting step, we review the results. To do so, we create a visualization tool to represent more easily the information we obtained, letting us join it to the maturity model information to begin answering the questions posed in the planning stage. This visualization dashboard will be described in Chapter 6 in more detail.

5.2 Systematic Tool Review

5.2.1 Planning

This phase focuses first on identifying the need for the review in our investigation and defining the protocol to follow.

I. Review Motivation

Background:

The need for this review comes from having to answer Sub-question #2 (**SQ2**):Which BI tools are available in the market and how they fit in DOMMs? With this review we obtain the list of tools, and the DOMM-specific analysis is done on a smaller but market-representative subset of said tool list. The focus on the software of BI projects comes from the fact that it

is the only DevOps aspect that is dependent on the technology the project is built on, unlike the culture, people and processes.

As explained in Chapter 3, BI is a popular area of research with hundreds of academic publications each year. However, the studies that focus on the software aspect of BI, more specifically, the tools commonly used in such projects, are very rare. Not only that, but the studies often analyze a single tool in itself. In some instances, multiple tools are compared with no clear reason for choosing them instead of the many other options available in the market [21]. In addition, there is no published analysis or research on DevOps maturity focused specifically on BI tools; nor a review of the state of BI tools as a whole. It is this lack of previous work for this field that we have decided to carry out a SLR to answer the initial sub-question and, at the same time, serve us to respond to the research-specific questions.

II. Review Protocol

Research Questions:

In this review, not only we need to obtain the tool list but also answer the proposed research questions in order to fully cover all aspects of **SQ2**. The research questions we want to answer relate to the DOMMs and this thesis's main research question regarding how well prepared BI tools are for its implementation. It is divided into the following sub-research questions:

SRQ1 **What are the maximum maturity levels reached by the tools in the DevOps Maturity Matrix?**

SRQ2 **What are the most common gaps for all the tools in the DevOps Maturity Matrix?**

To begin answering that question, we must define the strategy to follow to obtain the list of primary studies. From the previous section, we can make two important statements: "There is a low volume of tool analysis/representation in academic sources" and "The analysis we can find is a bad representation of what products are being used in the industry". A typical SLR would not be useful to our research objectives for these two reasons. Instead, we changed the type of data sources used to search for primary studies from traditional academic databases to Technology Research Services (TRS, description found in annex section B), letting us carry out a systematic review that properly samples and represents the available tools of the market that we will refer now as Systematic Tool Review (STR).

Primary Study selection strategy:

To decide which data sources to use in this review, we looked for various TRSs, performed an initial scoping of the potential PS available, and analyzed them to determine their validity to our research. The considered data sources are:

- **G2** [32]: Contains a small sample size of existing tools. It does not include an extensive group of tools, as the ones shown seem to be curated. Their "methodology" is a bit obscure, not specifying where the data is coming from.
- **Gartner** [16]: the most popular one of the bunch. Over the years, it has acquired three different companies focused on technology reviews and recommendations: **Capterra** [33], **GetApp** [15], and **SoftwareAdvidce** [34]. (although it's not always the same information in all three sites) since they operate under the same parent company.

However, Gartner provides not only research and articles but also reviews for products with peer insights [35].

- **IDC** [36]: It does not provide the information we need, focusing on analysis articles instead of individual tool information. There is no clear software analysis done, and the content, mainly research documents for the tools, is blocked behind large paywalls, rendering the site useless to our purpose even if the content met our needs.
- **Forrester** [37]: It is a pretty complex platform where the information format is akin to IDC but without the monetary costs. However, it still leads just a search engine for generalized reports that don't provide information about the individual tools.
- **BetterBuys** [38]: This website provides a curated review for an extensive list of tools. It's handy for curated descriptions and information about one tool but not specific enough when it comes to a complete list of features, making it hard to extract the information needed.

The final list of data sources used for this review process is three sites that operate under the Gartner corporation: *Capterra, GetApp,* and *SoftwareAdvice*. These three sources provide us with an extensive list of BI tools and software and extensive information about the tools themselves, their features, and even the scores and reviews that users have posted. These sites' search terms are straightforward and can be summarized into choosing the "Business Intelligence" category. It is important to remark that all three sites add a description of the review process and its review guidelines to verify the integrity of that data.

For this type of review, the Primary Studies (PS) will be the individual tools we have found listed from the data sources. The data extracted from these will be the product's characteristics and any information that may be relevant to our investigation. The data synthesis will focus on normalizing the information we obtained from the different sources to identify duplicate PSs, ensuring quality data.

Quality assessment:

The quality assurance for this review focuses on the data quality categories [39] most relevant for our case: For the sources, we select those that provide accurate, relevant, and complete data of our primary studies; and for those primary studies, we make sure that the information extracted from them is relevant and of consistent representation, something that, as mentioned before, is checked once again in the data synthesis stage.

Data extraction starategy:

Since the primary studies and their data are each listed in individual registries for the different domains we source it from, we develop a crawler and scraper tool to help us extract that information. For every data source, we are coding a separate instance since the inner workings of each site vary the structure, ensuring that the information obtained complies with the quality standards we set up. For necessary information not provided on the PS websites, we use the link provided in them to access the product's site and use it as a starting point to find the rest of the relevant data that may be needed.

Selection criteria:

The inclusion and exclusion criteria for this review are put in place to accommodate the complete list of available BI tools (provided by our sources) into a set that better represents the top programs of this industry. This selection helps us focus on those more popular solutions, which will make a bigger impact on the BI community. This also has an advantage:

the larger and more relevant set of features present for those tools compared to others. This decision could be considered a limitation of the scope of the PS that we select and would lead to future work to be made where a bigger portion of tools are reviewed. For this study, however, this restriction helps us with the volume of data needed to be extracted, specifically for the tool features and the manual process behind extracting that information.

Before beginning with the selection criteria for the BI tools we choose, we must define what constitutes a BI tool or what it takes to consider it one. There is a consensus on three main categories in BI systems [15],[16]: data management, data discovery, and reporting. **Data management** focuses on the organization, transformation, and structuring of the input data; **data discovery** supports the analysis and understanding of the data, and it allows us to come to meaningful conclusions with different methods, and; **reporting** is the data visualization, offering a more understandable format to convey the results of the analysis.

The three categories are the "full-stack" of BI parts of a project and are often covered by only one tool. However, there are many projects where this stack is composed of interconnected tools, each taking care of one or more key functions (including the three mentioned). However, for our study, we only want tools that can cover at least two of the three categories with their base functionalities (no plugins or third-party solutions), which is the limitation that we will set when applying the exclusion criteria. The reason to do this is to ensure that we obtain the tools that are closer to being considered BI and not those that can be of support in these kinds of projects, but that need others to be useful.

Now that we have explained the reasoning behind the selected criteria, the list of the inclusion criteria is:

IC1 The tool is labeled with at least two BI categories.

IC2 It is a relevant tool in the market.

And the exclusion criteria are as follows:

EC1 It must be publicly available (no beta versions or in-house solutions).

EC2 It must be localized to or available in English.

EC3 It should not be a niche solution.

EC4 It should still be supported (no legacy tools).

To define if a tool is "relevant" in the market, we reviewed the criteria used to obtain various "Top Contenders" lists of software in various TRSs [40] [41] [42]. We found out that the criteria the three sites have in common is the following:

- At least 20 reviews this last 24 months.
- Minimum normalized rating (rating data must not contain high deviations).
- Evidence of listed functionality.
- Serves North American users.
- Software isn't a niche solution.

5.2.2 Conducting

As mentioned before, no grand scale Systematic Review has been performed on BI tools, hence the lack of references for this exact procedure. There are cases [23] where the studies

centered around tool reviews but the representation in academic sources was more favorable than for this case, and a SLR was performed. For our case, we executed the protocol planned in the previous phase.

After developing the scraping tool used to obtain the list of PS from each source, we managed to obtain a list of 1782 items. Once the data was merged, and duplicate PS were removed, this list got reduced to around 900 different tools.

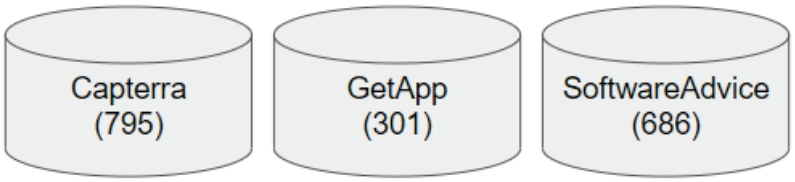


Figure 5.1: Initial list of primary studies Data extraction date: July 2021

To obtain our selection of BI tools, we proceeded the filtering stage with the inclusion and exclusion criteria. The data gathered from the three sources regarding the Primary Studies included all the information needed to begin with the filtering procedure. When it was missing, it could be easily added to our list, such as the case of available languages.

After merging the three data-sets and performing all the selection criteria filtering except analyzing if they could be considered BI tools (**IC1**), we obtained a list of 28 programs. For every one of them, we also extracted the information regarding BI-specific features, which was necessary to the evaluation of the BI categories for each tool for the inclusion criteria. In order to execute the filtering process more easily and also better visualize the information extracted, we decided to develop a dashboard that ingests the output data directly from the scraper. This dashboard is explained in more detail in Chapter 6.

With the tool ready, it is as simple as checking which programs did pass the BI consideration criteria **IC1** to obtain the final selection of valid BI tools (Figure 5.2). This list is comprised of 24 tools and can be found in Annex section D

At this point of the process, we begin the data extraction for the information regarding the specific software features of each tool. This step is done by manually checking each tool's website for this information. It is a demanding process that requires going through each tool's documentation and checking every feature listed for the DOMMs in Chapter 4. The result of this process can be partially seen in Figure 5.3, where every tool has informed if the feature is present or not or if it would allow for it to be used.

5.2.3 Reporting

For this case, the reporting is done through the dashboard visualization created we mentioned before and with the next section of this chapter. The goal is to communicate the results of the review and provide the details of the information obtained, but because of the nature of this thesis and the volume of information present in the annex and a GitHub repository

Figure 5.2: Final visualization of the tools and their Data Manipulation capabilities

Top Tools's Features

BI Features	Tool Name												
	Alteryx Designer	BOARD	Domo	Dundas BI	Exago BI	Grow	ibi	Klips	Logi Analytics	Looker	Microsoft Power BI	MicroStrat... Analytics	Mode
Ad-hoc Deployments	True	True	True	True	True	True	True	True	True	True	True	True	True
Ad-hoc Testing	True	False	True	True	True	False	True	True	True	True	True	True	True
Automated Builds	True	True	True	True	True	True	True	True	True	True	True	True	True
Automated Deliveries	True	True	True	True	True	True	True	True	True	True	True	True	True
Automated Deployment	True	True	True	True	True	True	True	True	True	True	True	True	True
Automated Environment Deployment	False	False	True	True	False	True	True	False	False	True	True	True	True
Automated Environment Management	True	False	False	True	True	True	True	False	False	False	True	True	True
Automated Integrations	True	False	True	True	True	True	True	True	True	True	True	True	True
Automated Monitoring	False	True	False	True	True	True	True	True	False	True	True	True	True
Automated Provisioning	True	True	True	True	True	True	True	True	True	True	True	True	True
Automated Testing	True	True	False	True	True	True	True	False	True	True	True	False	False
Builds with Test	True	False	False	True	True	False	True	False	True	True	True	False	True
Centralized Management	True	True	True	True	True	True	True	True	False	True	True	True	True
Code Repository	False	False	True	False	False	False	True	False	True	True	False	False	True
Continuous Delivery	True	False	True	True	True	True	True	True	True	True	True	True	True
Continuous Deployment	True	True	True	True	True	True	True	True	True	True	True	True	True
Continuous Integration	True	False	True	True	True	True	True	True	True	True	True	True	True
Continuous Monitoring	False	True	True	False	True	True	True	True	False	True	True	True	False
Continuous Testing	True	False	False	True	True	False	True	False	True	True	True	False	False
Data Management	True	True	True	True	True	True	True	True	False	True	True	True	True
Data source Monitoring	False	True	True	True	True	True	True	True	False	True	False	False	False
Migration tools	True	False	True	True	False	True	True	False	True	True	True	True	True
Product Monitoring	False	True	True	True	True	True	True	True	True	True	True	True	True
Rollback tools	True	False	True	True	False	True	False	False	True	True	False	False	True
Self-healing	False	False	False	True	False	False	False	False	False	False	False	False	False
Self-Service Deployment	True	True	True	True	True	True	True	False	False	False	True	True	False
Software Configuration Management	True	False	True	True	True	False	True	False	False	True	True	True	False
Ticketing	False	True	True	False	False	False	False	False	False	True	False	False	False
Tracked dependencies	False	False	True	True	True	False	False	False	True	True	True	False	True
Version Control	False	False	True	True	False	False	True	False	True	True	False	False	True
Version Monitoring	False	False	False	False	False	True	False	True	False	False	False	False	False
Virtualization / Cloud / Containerization	True	True	True	True	True	True	True	True	True	True	True	True	True

Figure 5.3: Final tool list and feature matrix.

5.3 Analysis and Results

Now that we have all the information needed, we can start answering the questions posed at the beginning of this section. We prepared some more visualizations in the dashboard, such as Maturity Matrix (Figure 5.4), that displays for each section how the tool is prepared for each maturity model listed from the previous chapter, breaking down per level its capacity.

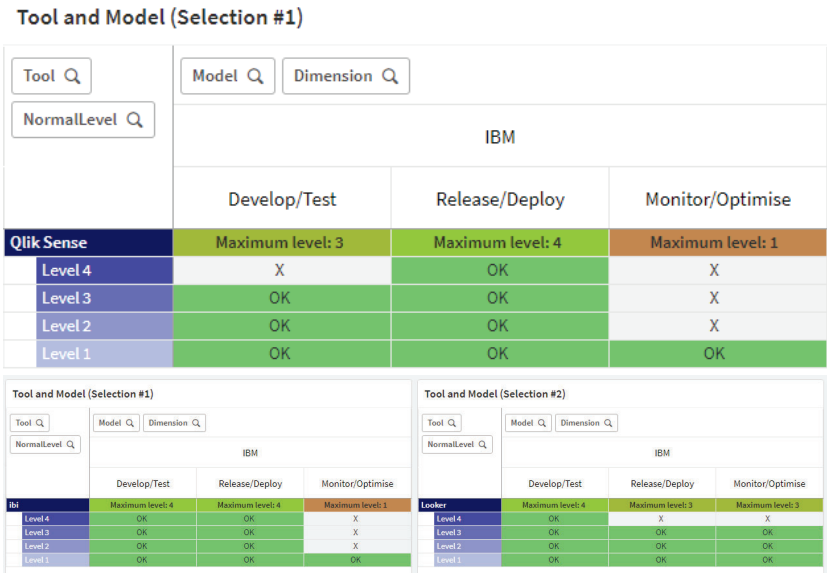


Figure 5.4: Maturity Matrix representation for selected tools and models

SRQ2: What are the most common gaps for all the tools in the DevOps Maturity Matrix?

Depending on which model we focus on, we can distinguish areas that perform worse than others. For CapGemini's model, code repository, version control, and version monitoring appear to be the biggest gaps, making the last maturity levels unobtainable for most tools. In Bucena's case, the issue also appears to be the lack of code repository availability, appearing only on 30% of the studied tools, but what really hinders all cases is the self-healing aspect, only present in one tool and blocking level 4 technology maturity for this model. In IBM, the bottleneck not only comes from the self-healing aspect but the ticketing feature, one aspect required to reach level 2 maturity for monitoring and optimization.

SRQ1: What are the maximum maturity levels reached by the tools in the DevOps Maturity Matrix?

Once we have looked at the gaps, we can see the issues it has caused to some maximum levels, restricting them to lower tiers. As mentioned before, Bucena and IBM have had intermediate levels of their maturities locked by the lack of features present in the tools. This is due to the fact that the models we studied are staged and require all goals to be achieved to advance to the next level. This does not mean that maturity is low in the whole model but that some of the most necessary features of the products regarding the technological aspect for Bucena and monitor/optimization for IBM are missing, and they would need to be prioritized.

Besides answering those two questions, what other insights have we obtained? It is important to note that the observed gaps can be easily explained within the context of how BI projects work. As mentioned before, BI tends to focus less on coding and scripting for development, using programs and artifacts that have a harder time being adapted into code repositories and version control. A similar case would be self-healing since it often relies on automated rollback procedures, something made harder by the last two features mentioned. We could argue that this is a problem coming from the maturity models chosen, seeing as these sections should not be present for these kinds of tools and that a shift in focus towards BI-centric goals would be more optimal to analyze the maturity of the tools properly.

For the case of ticketing though, the explanation would be that those kinds of features are rarely included by the tools themselves, and is a feature that is often provided, even in non-BI projects, by third-party services, such as Jira, ServiceNow, or BugZilla. This explanation however still shows that BI tools are often not enough by themselves and that external programs or services are often needed to support these tools and to help projects achieve better maturity levels.

Chapter 6

Method and Validation

6.1 Summary

This chapter is intended to present the dashboard tool and the method followed to develop it; as well as the validation process followed to verify if the information and insights obtained from the results of the STR coincide with real-world cases. This is achieved with a case study performed in a company to various active BI projects.

6.2 Introduction

In order to answer **SQ3** (How can we model implementation of DevOps methodologies in BI projects?) we decided to develop a method. This method would use the information obtained in previous chapters to model the implementation of DevOps methodologies in BI tools and projects.

6.3 Method

The core sections of this method can be split into two distinct tasks:

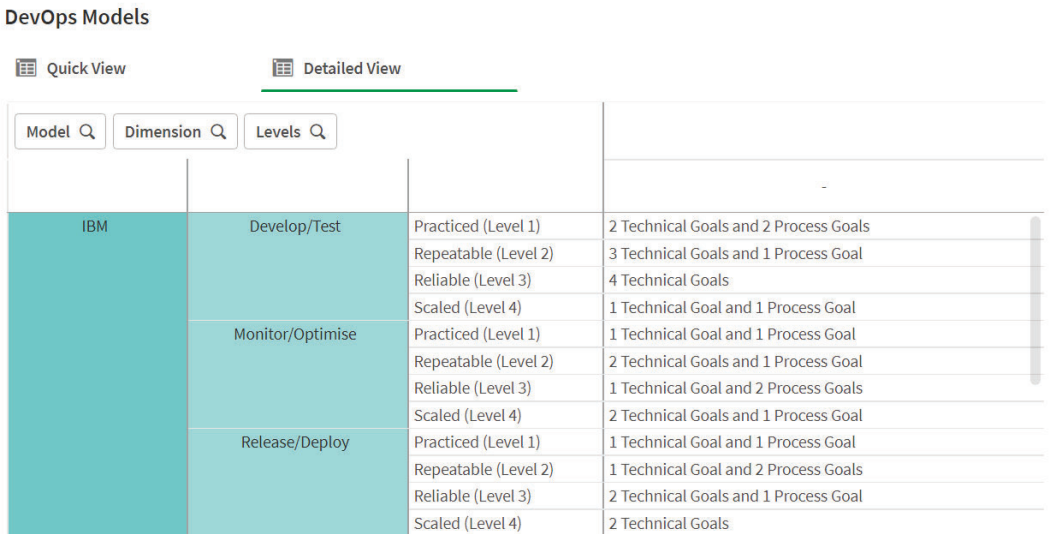
- Importing the different aspects to the model
  - DOMMs and its Technical and Process goals
  - BI Tools and their technical features
- Correlating those components between each other to represent the implementation.

The correlation of these two concepts is done with the Technical goals of the DOMMs and the Features of BI Tools. This part, however, ignores the Process goals, as they would correlate with the Process being executed at the project level. One important aspect here is the nature of the selected DOMMs, as they all are Staged fixed level. This means that to measure the maturity levels for the tools and consider a maturity level reached all goals and KPAs for a level must be in place, as explained in Chapter 2.

6.4 Dashboard: Approach Visualization

This dashboard is a proof of concept of how to model the data extracted from previous chapters and visualize its information to obtain useful insights. In our case, our idea was to use this dashboard prototype to help us answer the research questions and validate our results.

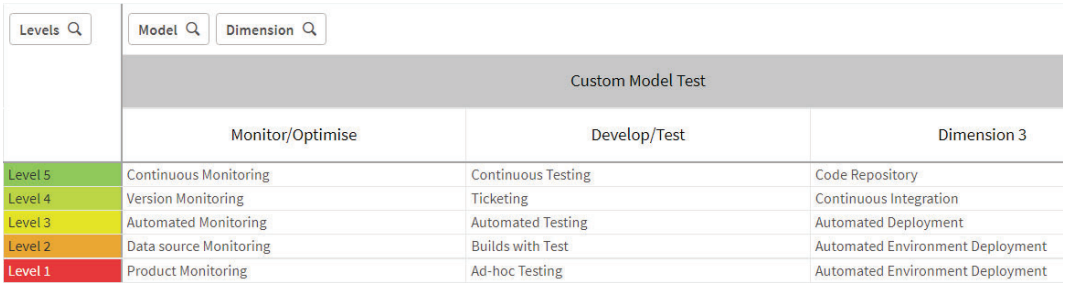
The first section of this dashboard is dedicated to the DevOps models. We included the three DOMMs selected for our study and a tabular visualization to break down their goals for each KPA and level.



IBM	Develop/Test	Practiced (Level 1)	2 Technical Goals and 2 Process Goals
		Repeatable (Level 2)	3 Technical Goals and 1 Process Goal
		Reliable (Level 3)	4 Technical Goals
		Scaled (Level 4)	1 Technical Goal and 1 Process Goal
	Monitor/Optimise	Practiced (Level 1)	1 Technical Goal and 1 Process Goal
		Repeatable (Level 2)	2 Technical Goals and 1 Process Goal
		Reliable (Level 3)	1 Technical Goal and 2 Process Goals
		Scaled (Level 4)	2 Technical Goals and 1 Process Goal
	Release/Deploy	Practiced (Level 1)	1 Technical Goal and 1 Process Goal
		Repeatable (Level 2)	1 Technical Goal and 2 Process Goals
		Reliable (Level 3)	2 Technical Goals and 1 Process Goal
		Scaled (Level 4)	2 Technical Goals

Figure 6.1: DevOps Model breakdown.

We added to the dashboard a prototype for the "custom maturity model creator" with limitations, such as only having one goal per maturity level. Some of the features included were letting the user decide how many levels and KPAs the model would have and changing their names. The importance of creating custom maturity models comes from the literature study done for this thesis, where academic studies [18] regarding the effectiveness of DevOps conclude that "Organizations are better off creating their model based on the various models available".

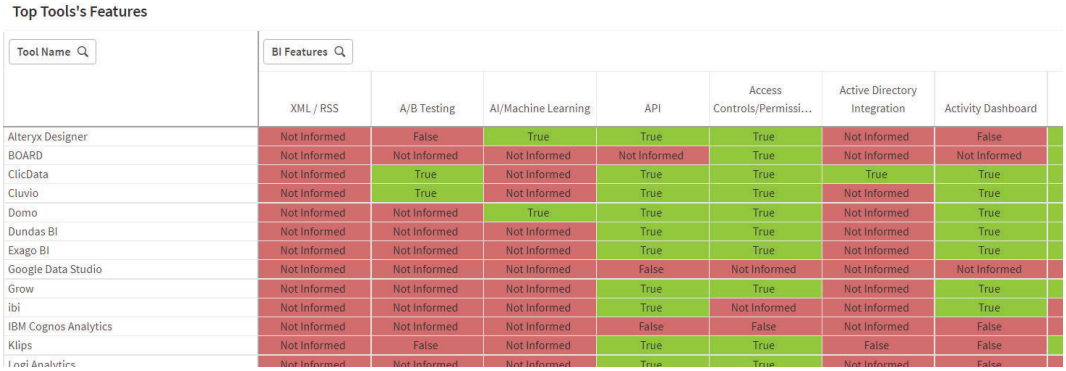


Levels	Monitor/Optimise	Develop/Test	Dimension 3
Level 5	Continuous Monitoring	Continuous Testing	Code Repository
Level 4	Version Monitoring	Ticketing	Continuous Integration
Level 3	Automated Monitoring	Automated Testing	Automated Deployment
Level 2	Data source Monitoring	Builds with Test	Automated Environment Deployment
Level 1	Product Monitoring	Ad-hoc Testing	Automated Environment Deployment

Figure 6.2: Example of how the Custom Maturity Model could be configured.

The next section of the dashboard focused on the BI tools. First, we listed those tools whose data we extracted was more detailed, having information like pricing, supported lan-

guages and devices... etc. The second page has a clear list of the selected PS we used in the comparisons and a visualization for their BI features (Figure 6.3).

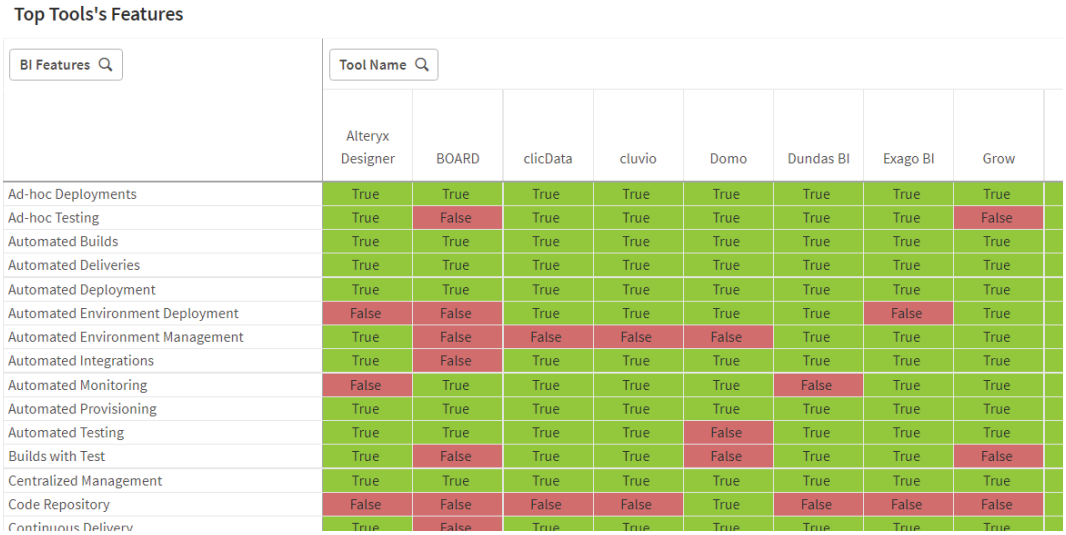


Top Tools's Features

Tool Name	XML / RSS	A/B Testing	AI/Machine Learning	API	Access Controls/Permissi...	Active Directory Integration	Activity Dashboard
Alteryx Designer	Not Informed	False	True	True	True	Not Informed	False
BOARD	Not Informed	Not Informed	Not Informed	Not Informed	True	Not Informed	Not Informed
ClicData	Not Informed	True	Not Informed	True	True	True	True
Cluvio	Not Informed	True	Not Informed	True	True	Not Informed	True
Domo	Not Informed	Not Informed	True	True	True	Not Informed	True
Dundas BI	Not Informed	Not Informed	Not Informed	True	True	Not Informed	True
Exago BI	Not Informed	Not Informed	Not Informed	True	True	Not Informed	True
Google Data Studio	Not Informed	Not Informed	Not Informed	False	Not Informed	Not Informed	Not Informed
Grow	Not Informed	Not Informed	Not Informed	True	True	Not Informed	True
ibi	Not Informed	Not Informed	Not Informed	True	Not Informed	Not Informed	True
IBM Cognos Analytics	Not Informed	Not Informed	Not Informed	False	False	Not Informed	False
Klips	Not Informed	False	Not Informed	True	True	False	False
Logi Analytics	Not Informed	Not Informed	Not Informed	True	True	Not Informed	False

Figure 6.3: Visualization of the BI Features for each tool.

Having both sets of information allowed us to create the last section of the tool that focused on analyzing the data. It starts with a tabular view of the different goals extracted and the list of tools, where every cell shows whether or not the tool passes that goal according to our theoretical analysis.



Top Tools's Features

BI Features	Alteryx Designer	BOARD	clicData	cluvio	Domo	Dundas BI	Exago BI	Grow
Ad-hoc Deployments	True	True	True	True	True	True	True	True
Ad-hoc Testing	True	False	True	True	True	True	True	False
Automated Builds	True	True	True	True	True	True	True	True
Automated Deliveries	True	True	True	True	True	True	True	True
Automated Deployment	True	True	True	True	True	True	True	True
Automated Environment Deployment	False	False	True	True	True	True	False	True
Automated Environment Management	True	False	False	False	False	True	True	True
Automated Integrations	True	False	True	True	True	True	True	True
Automated Monitoring	False	True	True	True	True	False	True	True
Automated Provisioning	True	True	True	True	True	True	True	True
Automated Testing	True	True	True	True	False	True	True	True
Builds with Test	True	False	True	True	False	True	True	False
Centralized Management	True	True	True	True	True	True	True	True
Code Repository	False	False	False	False	True	False	False	False
Continuous Delivery	True	False	True	True	True	True	True	True

Figure 6.4: Visualization of the Technical Features for each BI tool.

The second sheet was designed to visualize the theoretical representations of how the tools adapt to each model. This representation was used to answer the second part of SQ2, "how [BI tools] fit in DOMMs". It is important to remark that the tabular view shows how many goals each tool reaches but that this number alone does not accurately represent how well they adapt to the models. This is because our studied models are not "continuous maturity models", and missing a "basic" goal translates to poor maturity overall in that KPA.

Example shown on Figure 6.5: ibi has more goals met for IBM's model, yet Looker ranks higher overall in the maturity matrix representation.

Figure 6.5: Comparison between ibi and Looker's maturity for the IBM Model.

Based on the work of Samer I. Mohamed "DevOps Maturity Calculator" [43], we decided to include an "actual project maturity" section. Here users would tweak which goals they were reaching and which ones were not independently from the theoretical capacity of the tool to simulate the maturity of the project. This was used in the validation step. We used this to create the comparisons in Figures 6.6 and 6.7. We did not finish this feature in its entirety, and its state was not user-friendly, so it is missing from the final version of the dashboard.

One idea left for future work was the development of a similar interface that would let users compare BI tools. Users would input their maturity models or select one from the ones already included in the app to visualize which BI tool would be better suited in terms of features to meet that model's goals. It could also inform about said features differently, where users choose the tool. Then they can see its gaps and how they perform for the various maturity models loaded (as well as using the previously mentioned custom model).

6.5 Validation

The results obtained in the STR rely on multiple data sources and a rigorous approach to getting the information. However, we still need to validate them against the maturity of real-world cases. We decided to recreate the current maturity levels of a DevOps model for multiple BI projects of the collaborating company using the dashboard tool we developed. The maturity model chosen was IBM, as it is a complete model in terms of KPAs.

6.5.1 Reporting Project (Qlik Sense)

This project is, compared to others, of small to medium scope, dealing with low volumes of data and a simple model consisting of five tables joined between each other. It consisted of a single file where all the extraction, transformation, and visualization elements resided. Every week, the dashboard was used for a smaller subset of users who would routinely visualize specific information and generate reports. Since the data is not as critical as other cases and rarely deviates from the defined structure, monitoring the app is considered a low priority. This information is relevant to the results obtained and is provided to give an idea of this project's relevance.

We measured the different goals for this case and found exciting results comparing the expected maturity and real values. As expected, the features listed for this tool corresponded with what it had to offer in real life, but the maturity differences between the theoretical results and the ones obtained by doing this study came from the implementations themselves. In our results, various goals such as Automated Testing, Self-service deployment, and Continuous Monitoring were not being met since it depended on the development team choosing to implement them rather than the feature being present in the tool itself. On the other hand, the ticketing aspect was a priority, and they managed to reach that goal in the project using a third-party tool (ServiceNow). With this information plotted (Figures 6.8 and 6.9), we can see that there exists a difference between the theoretical maturity of the tool and how it is being used in the real world.

6.5.2 Company Operations Project (QlikView)

On the other side of the spectrum, this project has been inactive development for a couple of years and keeps evolving. Many users use it daily to check the information of many dif-

Tool and Model (Selection #1)

Tool | Model | Dimension | NormalLevel | IBM

	Develop/Test	Release/Deploy	Monitor/Optimise
Qlik Sense	Maximum level: 3	Maximum level: 4	Maximum level: 1
Level 4	X	OK	X
Level 3	OK	OK	X
Level 2	OK	OK	X
Level 1	OK	OK	OK

Figure 6.6: Theoretical maturity for Qlik Sense.

Tool and Model (Selection #1)

Tool | Model | Dimension | NormalLevel | IBM

	Develop/Test	Release/Deploy	Monitor/Optimise
Qlik Sense	Maximum level: 1	Maximum level: 3	Maximum level: 3
Level 4	X	X	X
Level 3	X	OK	OK
Level 2	X	OK	OK
Level 1	OK	OK	OK

Figure 6.7: Tool maturity of the studied project's Qlik Sense implementation.

ferent data sources that need to be combined in a specific transformation process. Downtime accessing the visualizations can be critical, especially in the morning hours, and the myriad of automated reports need the information to be up to date to be sent. It is one of the most critical parts of this team's responsibilities and requires two people in charge of it to develop new visualizations and issue resolution.

After comparing the theoretical analysis and the project's results, we could see that this case was similar to the previous one. Here, the theoretical estimation matches the real maturity of the project's tool, and lacking necessary features is solved by incorporating other tools to cover that area of the project. Since ticketing is very important to logging incidents and bugs, it is applied company-wide and used for this case, improving Monitor and Optimization maturity for this project (and many others).

To validate the project representation performed for the two real cases, we conducted interviews with various team members involved in its development, support, or management. Our objective was twofold: ask about the procedure of the maturity evaluation for every project and show them the results obtained for this chapter to see if they agreed on it. We also asked the users how they felt about a tool that streamlined the maturity evaluation procedure and showed them the prototype.

The results for these interviews were very positive regarding the representation of the maturity of the projects, showing no disagreement at all, yet the maturity evaluation section was a bit more concerning. Development team members answered that they did not choose

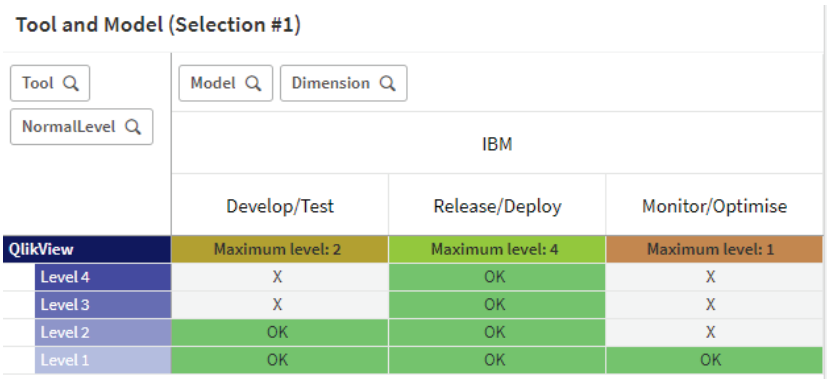


Figure 6.8: Theoretical maturity for QlikView.

	Develop/Test	Release/Deploy	Monitor/Optimise
QlikView	Maximum level: 2	Maximum level: 4	Maximum level: 3
Level 4	X	OK	X
Level 3	X	OK	OK
Level 2	OK	OK	OK
Level 1	OK	OK	OK

Figure 6.9: Tool maturity of the studied project's QlikView implementation.

the BI tool used, but the final client set it. And management revealed that the client often suggested maturity models, with some leeway given to them to choose a model in case the client did not perform any maturity analysis for the project.

When showing the prototype, the team members positively responded to it, letting developers better understand how the DevOps models measure the tool's maturity. Management remarked on the tool's usefulness and model comparison, specifying that representing to the clients the different maturity model options and the list of existing BI tools available could be helpful as leverage when the decisions regarding these matters are discussed.

Chapter 7

Conclusions

To wrap up this thesis, we will go over the research question, and sub-questions posed at the beginning, and we will answer them, adding an explanation to those results within this context. We will also go over the next steps and future work proposed.

***RQ:** How are BI projects currently implementing DevOps?*

As seen from the results obtained, BI tools are very different from each other and have different approaches to provide the software users with the sector-specific functionalities that they need, yet the features related to the tool itself that is also relevant for improving DevOps implementations into the projects are present in the majority of products, at least to the intermediate levels. Some aspects of these models are rarely found in these tools, sometimes from their high complexity to be implemented or for the fact that such specific functions are already supported by different software or services and adding them to the tool would be more trouble than what it is worth. However, It is important to note that teams that seek to reach the highest maturity levels may need to take into account what tool they are choosing, as some options will eventually limit the maximum maturity levels, and making up for its shortcomings is more complicated for certain functionalities than for others.

SQ1: Which are the state-of-the-art DevOps Maturity Models?

The current state of DevOps models is a very complex issue, as seen in Chapter 4 and in many other papers, such as Gasparaite's review [17], since even models that have been validated lack details in the required assessment. At this point, we have come to a similar conclusion to C. Marnewick et al. [18], which is that independent of the project's technology, existing DOMMs are a good starting point to start improving, but they should be adapted to create a custom model that fits the project or organizations' needs.

SQ2: Which BI tools are available in the market, and how do they fit in DOMMs?

The results we obtained regarding the BI tools are a good representation of the market leaders that currently exist for this area, being formed by a large number of different software products across this category. A very important part of this project was the dashboard created to view the information of each tool, but also to compare it to the maturity models and their goals. The results of that comparison found in Chapter 6 showed us that BI tools are well prepared for DevOps but that certain areas still require specific attention.

SQ3: How can we model implementation of DevOps methodologies in BI projects?

To be able to model the implementation, we must correlate both concepts together. As explained in chapter 6, it can be done by matching the technical features BI tools have with the goals described in the KPAs of DOMMs for the implementation of the "technological" aspects of a project. For other elements relating to Processes, Culture, and People, this would have to be done on a project-by-project basis.

7.1 Future Work

One of the most interesting aspects that have yet to be answered is if the creation of a BI DevOps maturity model would be beneficial for these types of projects. We have observed that existing DOMMs are not a perfect fit for BI-specific cases, where certain goals are still expected to be met when they are not relevant in this context. It would require an in-depth analysis of BI and carrying out a design relevant to this area with a similar procedure as the one M. Radstaak [12] followed, reporting and validating its usefulness.

Another aspect that is of very much interest and would still need more time for its development and validation is the maturity dashboard proposed as a prototype to the BI team in the company since the feedback obtained from them was very positive and would be of much use to teams that are looking for a way to measure and visualize the current maturity of their project in a simple manner. As mentioned before, this last aspect is inspired by the previous work of Samer I. Mohamed [43], where he proposed a "DevOps maturity calculator", the objective of which was similar to ours: allow users to represent the project and obtain its maturity scores. Developing this kind of tool and testing its effectiveness would be very useful and an interesting starting point for future investigations.

Bibliography

[1] Albert Nogués and Juan Valladares. *Business Intelligence Tools for Small Companies*. 2017. DOI: 10.1007/978-1-4842-2568-4.

[2] Guillermo Antoñanzas Martinez. "Project Files for: Business Intelligence Adoption of DevOps Methodologies". In: (2022). DOI: 10.5281/ZENODO.6394673.

[3] Daniel Stahl, Torvald Martensson and Jan Bosch. "Continuous practices and devops: beyond the buzz, what does it all mean?" In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, Sept. 2017. DOI: 10.1109/seaa.2017.8114695.

[4] In: *What is DevOps? - Amazon Web Services (AWS)*. URL: https://aws.amazon.com/devops/what-is-devops/.

[5] Monika Gasparaite and Saulius Ragaisis. "Comparison of devops maturity models". In: *IVUS* (2019). URL: https://www.semanticscholar.org/paper/Comparison-of-devops-maturity-models-Gasparaite-Ragaisis/6804c210d4106c91d6b5aeabd86804ca3decdea3.

[6] A Wiedemann et al. *The DevOps Phenomenon*. Apr. 2019. URL: https://queue.acm.org/detail.cfm?id=3338532.

[7] Christiane Gresse von Wangenheim et al. "Systematic Literature Review of Software Process Capability/Maturity Models". In: *10th International SPICE Conference on Software Process Improvement and Capability Determination, SPICE 2010* (Apr. 2010). URL: https://www.researchgate.net/publication/228738042_Systematic_Literature_Review_of_Software_Process_CapabilityMaturity_Models.

[8] Maximilian Röglinger, Jens Pöppelbuß and Jörg Becker. "Maturity models in business process management". In: *Business Process Management Journal* 18.2 (13th Apr. 2012), pp. 328–346. DOI: 10.1108/14637151211225225.

[9] Majid Al-Ruithe and Elhadj Benkhelifa. "Cloud data governance maturity model". In: *Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing - ICC '17*. Vol. 151. 10. New York: ACM Press, 2017, pp. 1–151. DOI: 10.1145/3018896.3036394.

[10] Jos van Hillegersberg. "The Need for a Maturity Model for Maturity Modeling". In: *The Art of Structuring*. Springer International Publishing, 2019, pp. 145–151. DOI: 10.1007/978-3-030-06234-7_14.

[11] M Paulk et al. "Capability maturity model, version 1.1". In: *IEEE Software* 10.4 (July 1993), pp. 18–27. DOI: 10.1109/52.219617.

[12] M. Radstaak Jeroen. *Developing A Devops Maturity Model*. URL: http://essay.utwente.nl/77808/1/ThesisJeroenRadstaak.pdf.

[13] D Power. *A Brief History of Decision Support Systems*. 2007. URL: https://dssresources.com/history/dsshistory.html.

[14] Tableau. *Business Intelligence: What It Is, How It Works, Its Importance, Examples, & Tools - Tableau, accessed 2021*. URL: https://www.tableau.com/learn/articles/business-intelligence.

[15] *GetApp BI Tool List*. Accessed June 2021. URL: https://www.getapp.com/business-intelligence-analytics-software/business-intelligence/.

[16] *Gartner: Global Research and Advisory Company*. Accessed June 2021. URL: https://www.gartner.com/.

[17] M Gasparaite. *Systematic Literature Review of DevOps Models*. DOI: 10.1007/978-3-030-58793-2_15.

[18] Carl Marnewick and Josef Langerman. "DevOps and Organizational Performance: The Fallacy of Chasing Maturity". In: *IEEE Software* 38.5 (Sept. 2021), pp. 48–55. DOI: 10.1109/ms.2020.3023298.

[19] Luciano Monteiro et al. *Methods of Implementation, Maturity Models and Definition of Roles in DevOps Frameworks: A Systematic Mapping*. 2021. DOI: 10.1109/CSCI51800.2020.00327.

[20] Barbara Kitchenham. "Procedures for Performing Systematic Reviews". In: *Keele, UK, Keele Univ.* 33 (Aug. 2004). URL: https://www.researchgate.net/publication/228756057_Procedures_for_Performing_Systematic_Reviews.

[21] Jorge Bernardino and Marco Tereso. "Business Intelligence Tools". In: Aug. 2013, pp. 267–276. ISBN: 978-94-007-4721-0. DOI: 10.1007/978-94-007-4722-7_25.

[22] Parth Nagar et al. "Comparison of generalized and big data business intelligence tools". In: *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)* (2016), pp. 3585–3588. URL: https://www.semanticscholar.org/paper/Comparison-of-generalized-and-big-data-business-Nagar-Atriwal/7bc9179390b05bc2254fee24cde99ef936143ffa.

[23] Juliana Alves Pereira, Kattiana Constantino and Eduardo Figueiredo. "A Systematic Literature Review of Software Product Line Management Tools". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 73–89. DOI: 10.1007/978-3-319-14130-5_6. URL: https://www.researchgate.net/publication/275224595_DevOps_shifting_software_engineering_strategy_-_value_based_perspective.

[24] Abubaker Wahaballa et al. "Toward unified DevOps model". In: *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*. IEEE, Sept. 2015. DOI: 10.1109/icsess.2015.7339039.

[25] R De Feijter et al. *Towards the adoption of DevOps in software product organizations: a maturity model approach*. Tech. rep. Utrecht University, 2017. URL: http://www.cs.uu.nl/research/techreps/repo/CS-2017/2017-009.pdf.

[26] Marite Kirikova Ineta Bucena. "Simplifying the DevOps Adoption Process". In: Vol-1898.Paper 14 (). ISSN: 1613-0073. URL: http://ceur-ws.org/Vol-1898/paper14.pdf.

[27] Samer Mohamed. "DevOps shifting software engineering strategy - value based perspective". In: Feb. 2015. URL: https://www.researchgate.net/publication/275224595_DevOps_shifting_software_engineering_strategy_-_value_based_perspective.

[28] Paul Bahrs. "Adopting the IBM DevOps approach for continuous software delivery Adoption paths and the DevOps maturity model". In: (12th Oct. 2013). URL: https://developer.ibm.com/articles/d-adoption-paths/.

[29] BTopham. "DevOps and OpsDev: How Maturity Model Works". In: (). URL: https://web.archive.org/web/20160305234317if_/http://community.hpe.com/t5/Business-Service-Management/DevOps-and-OpsDev-How-Maturity-Model-Works/ba-p/6042901.

[30] Capgemini. *A Capgemini Architecture Whitepaper - 2nd Edition DevOps - The Future of Application Lifecycle Automation*. URL: https://www.capgemini.com/de-de/wp-content/uploads/sites/5/2016/03/devops-the-future-of-application-lifecycle-automation.pdf.

[31] *A maturity model for DevOps teams Posted on May 30*. Tech. rep. 30th May 2016. URL: https://blog.leaseweb.com/2016/05/30/maturity-model-devops-teams/.

[32] *G2.com: Best Business Intelligence Software in 2021.* Accessed June 2021. URL: https://www.g2.com/categories/business-intelligence.

[33] *Capterra BI Product reviews*. Accessed June 2021. URL: https://www.capterra.com/business-intelligence-software/?sortOrder=most_reviews.

[34] *SoftwareAdvice BI Tool list*. Accessed June 2021. URL: https://www.softwareadvice.com/bi/p/all/.

[35] *Gartner: Business Intelligence (BI) Tools Reviews 2021*. Accessed June 2021. URL: https://www.gartner.com/reviews/market/analytics-business-intelligence-platforms.

[36] *IDC: Market Intelligence Firm*. Accessed June 2021. URL: https://www.idc.com/.

[37] *Forrester - Frontpage*. Accessed June 2021. URL: https://www.forrester.com/.

[38] *Better Buy - Top BI Tools*. Accessed June 2021. URL: https://www.betterbuys.com/bi/#review-content.

[39] Diane M. Strong, Yang W. Lee and Richard Y. Wang. "Data quality in context". In: *Communications of the ACM* 40.5 (May 1997), pp. 103–110. DOI: 10.1145/253769.253804.

[40] *Capterra: Research methodologies*. Accessed June 2021. URL: https://blog.capterra.com/research-methodologies/.

[41] *GetApp: Category leaders methodology*. Accessed June 2021. URL: https://www.getapp.com/category-leaders-methodology/.

[42] *Softwareadvice: Frontrunners methodology*. Accessed June 2021. URL: https://www.softwareadvice.com/frontrunners-methodology/.

[43] Samer Mohamed. "DevOps Maturity Calculator DOMC-Value oriented approach". In: *International Journal Of Engineering Research and General Science* 2 (Mar. 2016). URL: https://www.researchgate.net/publication/315671069_DevOps_Maturity_Calculator_DOMC-Value_oriented_approach.

Appendix A

Appendices

A List of Technical Features

1. Ad-hoc Deployments: the capability to deploy a version of the current project manually.
2. Ad-hoc Testing: basic, manual testing with minimal or no automation. It includes Unit testing or static code analysis.
3. Automated Builds: The build process starts automatically, usually after a change is made.
4. Automated Deliveries: Scheduled or automated delivery processes for the products.
5. Automated Deployment: Deployment automation, either by scripts or RPAs, can affect one or more environments
6. Automated Environment Deployment: Environments can be installed and configured automatically.
7. Automated Environment Management: Manage environments through automation, usually with scripts and daemons.
8. Automated Integrations: Product Integration is automatic. It's a separate part of the delivery or deployment process and can independently be scheduled or automated.
9. Automated Monitoring: Monitoring processes are automated and used to track different types of information.
10. Automated Provisioning: Data provision is handled automatically in the project.
11. Automated Testing: Tests are now executed automatically, usually scheduled in batches with scripts.
12. Builds with Test: Tests are now executed in the build phase and are taken into account to decide if the product gets delivered or not.
13. Centralized Management: There is a system in place to manage different types of tasks and processes related to the current project or product, like the different automated parts it may have.
14. Code Repository: All the project's code is saved and kept in a Code Repository.
15. Continuous Delivery: A Continuous delivery process takes place.
16. Continuous Integration: A Continuous Integration process takes place. Integration

covers the entire product and it is connected to acceptance testing.

17. Continuous Testing: A continuous testing process takes place.
18. Continuous Monitoring: Continuous monitoring process. This includes specific monitoring tasks such as KPI monitoring.
19. Data Management: The capability to handle and manage the data used by the product. (Its sources, data models, volumes...)
20. Data source Monitoring: Monitoring focused on the input data from any source to obtain different metrics like performance, volume....
21. Product Monitoring: Monitoring focused on the data generated specifically by the product.
22. Self-healing: It focuses on automatic problem localization, isolation, and issue resolution
23. Self-Service Deployment: It can provide self-service builds, provisionings, and deployments.
24. Software Configuration Management: Or SCM for short is used to manage the different configurations of a project to all its parts.
25. Ticketing: A service in place to allow individuals to report and track issues to the Dev team.
26. Version Control: Different versions of the product are saved. This is usually done to the code only but can be used with artifacts, builds, or even configuration files.
27. Version Monitoring: Monitoring focused on the values measured in each product release and the differences found between each of them.
28. Virtualization / Cloud / Containerization: Support for process virtualization.

B Technology Research Services

Technology research services provide buyers of software and hardware with data and information to make a more informed purchasing decision. These services may source their data from technology analysts who have built relationships with technology vendors to better understand tools or crowdsourced peer data that build recommendations based on unbiased user feedback. Additionally, technology research firms often run surveys and speak with customers directly to gain further knowledge of software and hardware.

C Bucena's Technological Practices

TECHNOLOGY maturity levels				
ID	**Repeatable (2)**	**Defined (3)**	**Managed (4)**	**Optimized (5.)**
T1	*Hardware maintenance practices*	Integrated configuration management		
		Server virtualization practices		Infrastructure as code
T2	Automated testing		Continuous testing	Continuous experimentation and learning
			Continuous monitoring	
			Automated dashboards	
T3	Collaborative development	Integrated Deployment planning	Continuous testing	Continuous monitoring
	DB management practices			
				Application monitoring
T4	Build automation	Continuous integration	Continuous deployment	Integrated change management
				Constant, effortless communication and collaboration
				Active Stakeholder participation
T5	Collaborative development	Continuous integration	Common metrics for Dev.t. and Op.t.	Continuous delivery
			Continuous monitoring	Constant, effortless communication and collaboration
T6	Continuous business planning	Continuous integration	*Knowledge management practices*	–
	Integrated Deployment planning		Constant, effortless communication and collaboration	
		Active Stakeholder participation		
T7	Integrated configuration management			–
		Infrastructure as code	Continuous experimentation and learning	
T8	Application monitoring	Continuous customer monitoring and feedback	Automated dashboards	–
	Continuous monitoring			
	Common metrics for Dev.t. and Op.t.			
T9	Production support	Integrated change management	Shared goals, values, respect, trust and incentives	–
	Shared responsibility, ways of working and collective ownership	Integrated Deployment planning		

Figure A.1: Table 2 in Bucena's Maturity Model [26]: DevOps practices relevant to Maturity Model Technology area

D Final BI Tool list

Tool Name	Website link
Domo	www.domo.com/features
BOARD	www.board.com/en/features
Grow	www.grow.com/product/features
Holistics	www.holistics.io/features/
Klips	www.klipfolio.com/klips
Looker	looker.com/product/new-features
MicroStrategy Analytics	community.microstrategy.com/s/documentation
Mode	www.mode.com/
Qlik Sense	www.qlik.com/us/products/qlik-sense
QlikView	www.qlik.com/us/products/qlikview
SAP BusinessObjects Business Intelligence	www.sap.com/products/bi-platform/features.html
Exago BI	exagobi.com/software/product/reports/
cluvio	www.cluvio.com/features.html
clicData	www.clicdata.com/product/
Alteryx Designer	www.alteryx.com/products/alteryx-platform/alteryx-designer
Dundas BI	www.dundas.com/dundas-bi/platform
ibi	www.ibi.com/
Logi Analytics	www.logianalytics.com/dashboard-data-visualization
Microsoft Power BI	data-flair.training/blogs/power-bi-features/
Phocas Software	www.phocassoftware.com/phocas-analytics
Sigma Computing	www.sigmacomputing.com/
Sisense	www.sisense.com/solutions/developer/
Tableau	www.tableau.com/products/server
Zoho Analytics	www.zoho.com/analytics/business-intelligence-bi-software.html