School of Engineering Science
Software Engineering
Impact and Benefits of Digitalization
Assoc. Prof. Ari Happonen

DEVOPS PRACTICES FOR SOFTWARE DEVELOPMENT AND CONSULTING FIRMS

A case for EfiCode Oy

2021

Fahad Ahmed

Erika Bottacci

Semi Rantanen

Joonas Romppanen

Noella Verschuren

ABSTRACT

Lappeenranta–Lahti University of Technology LUT

LUT School of Engineering Science

24 pages, 5 figures, and 5 tables

In the last few years, there has been a growing trend in the fast and frequent deployment of new software features. This is a result of companies adopting several different strategies, including the DevOps approach. DevOps focuses on the collaboration between software deployment and operations. In practice, DevOps affects a lot of company processes, culture, products, and many more aspects. However, the concept of DevOps is still ambiguous and difficult for software companies to adopt. Therefore, this paper focuses on how DevOps can improve the business. The research was conducted by analyzing the available literature and in collaboration with a software consulting company EfiCode. The key finding of the research is that the adoption and implementation of DevOps practices are important in software companies. When adopting DevOps, companies need to identify which data can bring the most value and consequently prioritize the data. Also, dashboards should be utilized to assist in the fluid and lean process because of DevOps.

Keywords: DevOps, DataOps, DevOps dashboard

Table of contents

Abstract

1 Introduction

Companies offering software and internet-based services are regularly deploying new software functionality to customers (Lwakatare et al., 2015). This continuous deployment of software has brought both opportunities and challenges for companies (Claps et al., 2015). A new phenomenon that helps facilitate this is DevOps. DevOps addresses the challenge as it addresses the existing gap between development and operations personnel, which is responsible for the deployment, management, and support of the systems at the customer's site (Claps et al., 2015). It is a set of practices intended to reduce the time between making a change in the system and placing the change into the production, while continuously making sure that there is high quality (Bass et al., 2015). In other words, there is a focus on a fast, efficient, and flexible integration of deployment, delivery, and operations to create a lean, fluid connection of the traditionally separated silos. It requires software companies to implement automation, improve agility in designing, delivering, and operating software products and increase communication amongst stakeholders (Lwakatare et al., 2015).

In recent years, DevOps has become a more and more discussed phenomenon within software engineering in both academia and business areas. In many papers, positive expectations of DevOps are discussed, while similarly, organizations are becoming increasingly interested in DevOps and how to receive potential benefits from it. Consequently, it has been identified as an increasingly important aspect (Riungu-Kalliosaari et al., 2016).

However, DevOps practices impact processes, products and services, organizational structures, associated technologies, and business practices and opportunities (Lwakatare, 2017). This brings besides many opportunities, also many challenges. These changes can introduce organizational and business stresses. Consequently, adopting DevOps may bring difficulties for companies since it may require that it introduces the process, personnel, and technological changes and innovations. Furthermore, a successful DevOps adoption is unique to each organization. This makes it difficult to establish best practices in academia and for other organizations to learn from each other (Lwakatare, 2017). Therefore, there is a need to investigate the DevOps phenomenon, examine how it impacts software development companies and how companies can benefit from it.

Thus, as the interest in DevOps continues to grow, there is an increasing need for software organizations to understand how to adopt it successfully. A company that also wants to know more about the possibilities of DevOps, is the company EfiCode. EfiCode is a software company that has adopted DevOps in its software development processes. This paper has as objective to clarify the concept and provide insight into how to successfully adopt DevOps. To provide these insights, the following research question is defined:

"How can DevOps improve the business of a company in the industry of EfiCode?"

To answer the research question, first, the existing literature is reviewed. Based on the literature review, a definition of DevOps is formed, and there is looked at how the term is used in practice. Furthermore, there is looked a list of characteristics that an organization needs to adopt or exhibit to work according to the DevOps thinking. Secondly, an empirical study is conducted. Data is collected by conducting qualitative interviews with EfiCode about the social and organizational aspects of DevOps in their context. The Semi-structured interviews will offer new insights which will be combined with the conducted literature review. Then, a dashboard will be created in which different elements of DevOps will be applied suitable for EfiCode. Finally, a conclusion and discussion will be given about the gained insights.

2 DevOps in software development

DevOps was first introduced between 2007 and 2008 (Atlassian, n.d.), and even though it has been over ten years since its introduction, according to Leite et al. (2020), DevOps is still missing a widely accepted definition. They used in their article the following definition to define DevOps: “DevOps is a collaborative and multidisciplinary effort within an organization to automate continuous delivery of new software versions while guaranteeing their correctness and reliability.” The authors crafted the definition by going through the most cited definitions of DevOps and then formed their own based on those. According to them, DevOps can be seen as an evolution of the agile movement, which means in software development a loop in which development produces small releases and with feedback from the customers they develop the product further.

DevOps processes allow components to be tested and released after completion, a project can be divided into smaller parts, through automation it is possible to reach a good quality. Firstly, the client requirements must be identified. In the development cycle, the developers build the project components. In the operations cycle, these components are tested. If an error is found, the component cannot process to the next stage of DevOps. When each module is delivered to the customer, it is time to gather feedback. Therefore, feedback loops are created to further improve the software (Banica et al. 2017). Deployment pipelines eliminate waste in the software development process by providing quick feedback to teams without running excessive tasks. Software deployment is split into stages with tasks running sequentially or in parallel within each stage. (Atwal 2020) In software development, DevOps reduces time to market and costs of production, but also improves service quality as architectures are adaptable (Capizzi et al. 2020).

Even though DevOps’ purpose is to solve the development problem and give an agile pipeline to organizations, there are still raised questions about the solution. The authors of the article (Leite et al. 2020) points out that people working in such organizations might face the fact that they need to change their work culture radically from the past, which usually is easier said than done. Also, other questions rose about the organizational structures and team line-ups. These are points that every organization needs to figure out themselves and for every organization, the answer or a solution is different as there are not two similar organizations doing the same thing.

The article of Leite et al. (2020) also discusses the meaning of the DevOps for the organisations. Main point in the organisation level that the authors bring out is the close cooperation between the developers and operators. They point out that DevOps naturally breaks the silos that are separating developers and operators from each other. The development of the process began before 2008, as it was realized that agile development does not work without close cooperation between developers and operators, which reduced the conflicts between the two.

Leite et al. (2020) discuss DevOps and its challenges from three different points of view: engineers, managers, and researchers. The biggest advantage that DevOps brings for the engineers is the fact that they can call themselves DevOps engineers, this has a straight affect on the engineer's salary, according to the article. Other than that DevOps mainly brings only challenges, more responsibilities, and new things to learn for every engineer working in a DevOps organization. The fact that DevOps itself contains many different tools that need to be learned is a challenge itself for anyone working at the implementation level.

The biggest challenge for the engineers is to create an architecture and infrastructure for the company's new system. In the webpage of AltexSoft (AltexSoft, 2019) it is said that after the maintenance and upkeep biggest hidden costs of modernization of the software system are the costs of the implementation. Even though the modern systems are quite integration-ready third-party tools and services require a significant amount of custom code, there is always the risk that the final integration might not work with the tools and account features must be updated later.

Management in DevOps organizations is responsible for the organization's performance and of the tools in use. The management is the one who makes the call about how the organization will adopt the DevOps and how the organization will be built, which are some of the most critical questions that management needs to find answers to. As the management is responsible for the DevOps implication, it is also responsible for the feedback loops that bring vital information about the product and things to focus on from the customers.

Another thing that has an impact on the organization, which is responsible for the management, is the culture inside the organization. This is a big factor especially in an organization, that is turning from a traditional software organization to DevOps one. If the culture in the company is that fighting against anything new is seen as an okay thing, which

can cause big problems in a situation, where big organisational changes are needed. The key to minimize these challenges, comes from the attitude of the management: how do they see the change in the organization, and do they support the change in the organization.

Even though management might not use all the tools that engineers do, they still need to be able to choose the right ones for the organization and this decision cannot be left to IT-department to decide as Ross and Weill (2002) state in their article “Six IT Decisions Your IT People Shouldn’t Make”. As the decision should be made by the management, the level of the organization, which uses the tools, should be consulted, since the tools must be right for the organization. Management also needs to know and understand whit whom and why people collaborate in different tasks, to be able to choose tools, which help both the implementation work, as well as the early new project solution innovation phases (Salmela et al., 2013).

According to Leite et al. (2020), there is mistrust between the customer and the contractor in normal kind of development projects, where the development phase might take a long time and the customer does not get any version of the product before the test phase. The authors noted that this is the case especially with big corporation clients and if the government is the client. However, the authors’ findings point out that continuous delivery increases the trust between the client and contractor, even in the case of the government being the customer.

Through process automation and continuous delivery (CD) pipeline, it is possible a quick and incremental delivery of new features. CI automates the integration of codes, modules, and parts of whole software and products (Capizzi et al. 2020). Continuous delivery consists of a foundational deployment pipeline, continuous integration of new development, and automated testing leaving the code in a state ready for release into production at any time (Atwal 2020).

The organization must take advantage of tools to ensure rapid interaction and to limit the time spent on solving tasks. Continuous integration (CI) ensures that each piece of code developed is merged and without errors, so to improve the company s possibility to fully set quality standards. Developing software means “compiling code, managing dependencies, generating documentation, running tests, or deploying an application to different environments”. CI’s objective is to integrate the work of developers promptly. This approach

would ensure that systems are constantly tested and advancing in the processes (Ebert et al. 2016).

Managing DevOps can be a tedious task, as poor knowledge and communication management often lead to conflict. Spending time on conflict automatically reduces the time available to solve technical problems and instruct developers. The conflict has economic costs but also creates stress among employees, with negative implications on organizational performance and overall product quality (Atwal 2020). In support of decision-making, different stakeholders must be assigned different roles, as well as levels of authority and responsibility. Stakeholders have a key role in conflict resolution, but also in providing directions on how to further improve services, suggesting changes and updates (Sahid et al. 2017).

It is important to account that some developers might have generally little, if none, knowledge about their contribution to the overall DevOps process. If single members are not able to see the big picture, their motivation and engagement can be limited. Nevertheless, when rework is needed to solve a problem, miscommunication, and lack of accountability for own responsibilities and contributions significantly hinder the processes of optimization (Atwal 2020).

Especially, when a company is developing and enhancing its current process models for their business operations, the management needs to understand the effects these activities have on their organization culture (Santti et al., 2017). If the team members, collaborative partners in business networks, and own employees in the company do not see their value and/or input to a big picture, they might not trust management (Happonen and Siljander, 2020) and work fully towards the set new goals and requested new ways of working.

3 DataOps

DevOps generates a high variety, veracity, and volume of data. For successful product development, the process of data analytics must be integrated within the organization. Thus, for managers and developers, it must be paramount to approach the business holistically. Guaranteeing the efficiency of data pipelines ensures the supply of data within multiple business units (Figure 1). The final goal of DevOps is to use automation to discover, access, clean, and store data. For this reason, data must be scalable and have a certain degree of accuracy (Atwal 2020).

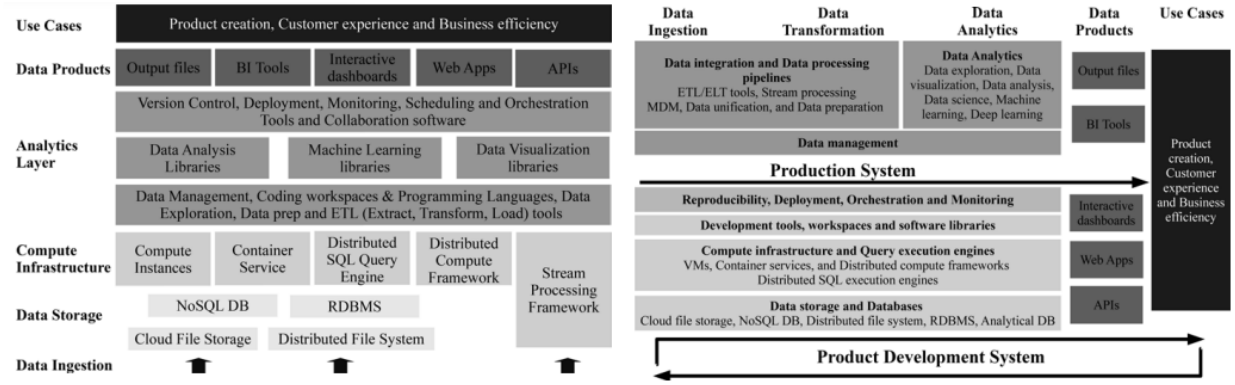

Figure 1. On the left, typical hardware, and software layers in the data lifecycle. On the right, data analytics is a production and product development system. A comparison (Atwal 2020).

To ensure data flow and analytics, and achieve their mission, organizations must set up their data strategy. A strategy helps monitor changes, and guarantee that these changes are topical, rapid, and effective throughout each phase of the data lifecycle management (Atwal 2020). Managing data lifecycle means monitoring the whole process of data flow, from the phase when data is gathered until the moment when data is no longer considered valuable for the company's objectives (Figure 2). Thus, the data lifecycle includes the following stages:

- Capturing data from internal and external sources, either human or machine.
- Storing data into files, databases as data structures, models, or other formats.
- Processing data whilst flowing through each process, thus, cleaning, enriching, processing to make them useful for the following stages.
- Sharing data and integrating them into the system for efficient use. These can be data generated from transactions, analytics, and reports.
- Using data, as well as sharing, to provide insights or employing for further analytics.
- Retiring data according to requirements, costs, value decay. This is done with archiving and purging (Atwal 2020).

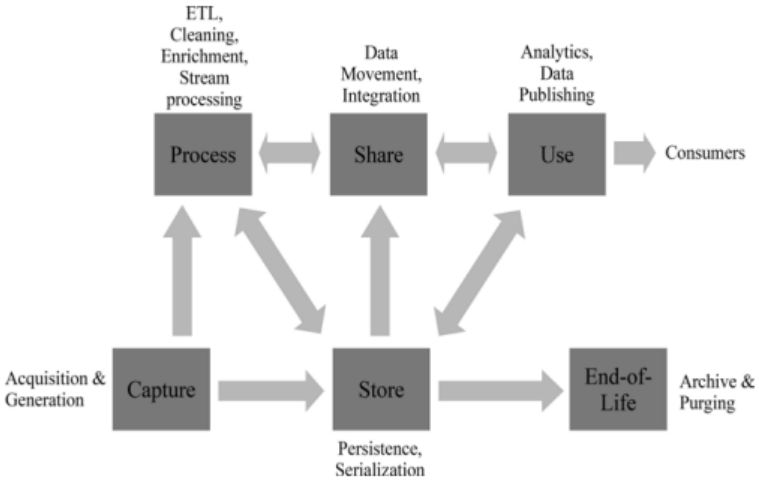

Figure 2. The stages of the data lifecycle (Atwal 2020)

In a DevOps pipeline, a considerable amount of data is generated by changes made by developers throughout the project cycle. For example, data is generated when building applications, after compiling the program and giving new entries, when the project has dependencies, when executing automatic tests, when the final product is released into the market, and finally when the product reaches the end-users (Capizzi et al. 2020). Capizzi et al. (2020) identify what data that can be generated through each DevOps stage. In the "planning "stage, clients and developers meet up to discuss what service must be implemented. In this first stage, the software is designed by accounting for users' first requirements. From here, the first documentation is generated. Generally, this documentation includes user stories, tasks, events, backlogs, and statistics. The following phase is the "coding" stage. The interaction with clients is minimal, if not absent. During the coding stage, information is generated from scripts, upgrades of previous software versions, thus, all the information that supports the comparison between old and new prototypes. The "building" stage follows the coding. In this part, programs are compiled, and files executed, data generated in this phase include successes, alerts, errors in compilation, and failures. The following "testing" phase generates and collects data from the testing of the software. This might include information about the type of algorithms used for testing, log data with tests that passed and that failed, thus reports of quality, and other documentation. The "releasing" stage generates data about the product launches, so to highlight the changes in features, files, packages, logs, and metrics between old and recent versions. The "deploying" stage that

follows generates data about the successes and failures, thus error messages, and warnings. The “operating” stage informs about the performance of software, hardware, tools, and systems where the service is operated. Finally, the “monitoring” stage generates data from users’ feedback, so provides a set of data such as logs and statistics (Capizzi et al. 2020).

3.1 Monitoring business initiatives with data

Data is a strategic asset for achieving successful DevOps procedures. As DevOps generates a huge amount of data, management of these data can be challenging for organizations that do not employ enough resources or do not have sufficient business capabilities (Capizzi et al. 2020). Therefore, organizations need to prioritize what are the most valuable data in terms of strategic development. To prioritize, one should identify the implications of these data (Table 1).

To ensure a successful strategy, the management must store information about strategy, rules, policies, objectives, plans, processes, procedures, documentation, and resources. It must also track services, finances, opportunities, threats, and customer feedback. These data support the governance to manage business initiatives (Sahid et al. 2017). Furthermore, records should be stored about initiatives, resources, and budgets in support of business planning. Companies must keep track of how initiatives impact the business and evaluate progress and results. Information regarding operations, system functions, business projects, activities, maintenance, and administration can ensure further service quality (Sahid et al. 2017).

The health of business IT infrastructures should be monitored as they affect organizational processes. The tracking must account for the load of CPU, allocation of RAM, statistics on network traffic, memory consumption, disk space availability. If certain events occur, automated tools generate reports and send notifications. Thus, it is possible to respond immediately in case of negative events and mitigate the impact (Ebert et al. 2016). To achieve better system performance, incidents must be reduced. Incidents happen due to poor management of problems, the limited capability of system infrastructure, reduced availability of service and organization, as well as unmanaged operations, and poor system configuration (Sahid et al. 2017). A way to successfully manage applications is to record log data. These data include real-time information such as failures, errors, warns, debugs, and

other info gathered from applications, platforms, and servers. According to the environments' complexity, log data can be either collected through cloud tools or standard frameworks (Ebert et al. 2016).

User event data are crucial in support of service development. These data are generated from users' activities such as user registration, website visits, and clicks, access to email, or completed purchases. When stored, these data must have a timestamp, event name, type, and details of conditions and causes must be explicit too (Atwal 2020). User satisfaction must be recorded too, as it serves the company to gain customer value, this data could be gathered through surveys or customer feedback (Sahid et al. 2017). This is achieved through feedback loops. Feedback loops help assess the quality and health of a product, whether it responds to users' requirements but also serves to assess the health of the team and quality of teamwork. Reports are generated from processing feedbacks, identifying errors in data quality, measuring the accuracy of model predictions, or carrying out experiments (Atwal 2020).

Table 1. Business domain, data generated, and their managerial implications

Domain	Categories	Data	Managerial implications
Business governance	Functions	Tactics, policies, programs, operations, processes, initiatives, services, budgets, costs and finances, opportunities, threats, customer feedback, impacts, progress, results, analytics, reports, maintenance, administration	Support for long-term plans and strategies, open to new business models, support for business planning, provide insights on how to implement the business and operations
Business IT management	Objects	Codes, modules, projects, system functions, infrastructures, tools, technologies, experimentation, iteration, platforms	Improve project management and planning, manage assets and resources, support decision-making
	Infrastructures	Load of CPU, allocation of RAM, statistics on network traffic, memory consumption, disk space availability	Knowledge management, decision-making, manage accidents, security, improved relationships with stakeholders
	Events	Timestamp, name, type, event conditions, customer	To manage applications, find faulty points as soon as possible,

		registrations, website visits, clicks, email opens, and sales transactions, fatalities, errors, warn, info, trace, debug	improve service and relationship with stakeholders, keep track for handling future events
Stakeholders' relationship management	Communication	Emails, chats and other internal and external messages, meetings, feedbacks loops, surveys, reports	Improves relationship with stakeholders, management of human resources, track of activities, improve quality, perform analytics
	Documentation	Rules, policies, legislations, contracts, frameworks, agreements, procedures, negotiations	To organize the activities, set responsibilities, ensure that legal requirements are followed, ensure customer satisfaction, maintain a reference in case of accidents or when handling complex situations, such as conflict resolution, negotiation, or problem-solving
	Stakeholders	IDs, roles and responsibilities, authority level, competencies, contribution, satisfaction, engagement, collaboration, relationship, ownership	To improve workflow, provide autonomy, create work teams, manage conflicts and communication, solve problems, improve process delivery

3.2 Achieving business objectives with data

Understanding the impacts and uses of data at the management level of organizations, is crucial for understanding how to use these data to achieve certain business objectives. Through data, it is possible to measure customer satisfaction, manage resources and monitor their performance, and, overall, improve the product quality (see an overview in Table 2).

To improve customer satisfaction, it is important to account for the time needed to deliver the service, whether deadlines are met, what level of information customers have, how risks will be mitigated, what the tasks needed to complete a project are, how much time is spent on fixing problems. To these are included, lead and cycle times, progress level and percentage allocation of each task, number of tasks started but not completed, total work in progress, due date and performance compared with the success rate, number of items and

tasks that are blocked or delayed, elements and causes that block the workflow, rework rates due to errors or miscommunication, and periodical throughput (Atwal 2020).

To optimize resource utilization, and minimize costs, data that track the performance of resources and applications can be gathered in the cloud or data centers. Information can include log events, real-time performance, budget available and spending, usage, and workload. These metrics help at tackling problems and resolving them promptly. Having such control helps to manage budget and workloads and keep a minimal expenditure. Notifications can be set to inform developers and managers when certain limits have been achieved, but also to identify opportunities for process optimization and increase returns in investments (Atwal 2020). To monitor the efficiency of the CI/CD pipeline, data can indicate the frequency of deployment, changes in lead time, the time needed for a new feature or bug fix to be officially released, failure rates, and average time to recover from a failure (Atwal 2020).

To improve employees' performance, teams should be based on tools and skillsets available. Thus, data should help managers not duplicate resources across teams and allocate talents to projects as soon as they are available. Information should be provided also about individuals and teams' work-in-progress state and at which stage of the process the project is, but also, items that are blocked, the time they have been blocked, location of the block in the DevOps process, and causes of blockage (Atwal 2020).

In support of teams' performance, employees must be held responsible and share their duties and skillset with their mates. Team members must be able to cooperate, for example, to perform a root-cause analysis to deeply understand why and where a problem occurs. Thus, data about collective work-in-progress, collective contribution to single item delivery, achievement of customer requirements, lead time, and quality percentage should be visible (Atwal 2020).

Table 2. Achieving business objectives and metrics

Business objectives	Data generated/collected/needed
Achieve customer satisfaction	Time to deliver, deadlines met, risk mitigation, tasks mix, time spent on fixing problems, lead times, cycle times, percentage allocation and progress for each work type, total work in progress, due date, success rate, items blocked or delayed, blockers, rework rates due to errors or miscommunication, weekly or monthly throughput, the satisfaction rate of service delivery (customer is satisfied with the

	service), produce meets business objectives (customer satisfaction evident through with data)
Utilize resources optimally	Log events, real-time performance, budget allocation, spending and workloads for each resource, exceeding thresholds, frequency of deployment, change in lead time, time for a new feature or bug fix to be released, production deployment failure rate, and average time to failure recovery
Improve employees' performance	Skillset, tools in use, time availability, contribution to the project, completion rate, work in progress for the individual and for the team (waiting to access the system, for review, for testing, and so on), blocked items, time spent since they are blocked, wherein the process the block occurs, internal and external causes
Improve teams' performance	Responsibilities and skillset of the team and single individuals, collective work-in-progress, collective contribution to single item delivery, achievement of customer requirements, lead time, quality percentage, the health of team and processes (team's concerns about the service), the health of product (team's concern about data)

3.3 How to share data

The goal of a DevOps pipeline is to reduce the time to deliver the service, ensure continuity, improve collaboration, ease conflict resolution, and reduce miscommunication problems. With such a management methodology, the organization can foster a culture of development, quality control, and optimal operation flow (Ebert et al. 2016). Staff members must have granted independent access to data and infrastructures, the relationship between software engineers and data scientists must be closed, as an active discussion should follow on understanding what functionalities must be built for a service and what data should be available for reuse. Data sharing in support of DevOps practices aims at improving the cooperation between stakeholders: the ones responsible for managing data, such as data engineers and data scientists, and the ones consuming data, such as managers, developers, and end-users (Atwal 2020).

To improve sustainability in the approach of managing DevOps, the optimal data sharing practice must promote a pull-based workflow: only when a team has completed a task, and the latest priority and value of information have been accounted for, it can be finally decided whether the team can move on to the next task. As a pool of waiting tasks ranked by priority is generated, the pull system provides more options to choose from, increasing the degree of freedom in the decision-making process (Atwal 2020).

The process of data sharing generates data analytics and reports. This might include producing tables and reports about historical events, describing what has happened and displaying it on a dashboard, and using a tracking system supported by meaningful Key Performance Indicators. Data mining and visualization techniques can help at finding patterns and relationships between data and describe the roots of certain events. Experiments could be carried out so that predictive models and statistics could identify customers' behaviour and actions. Finally, with process automation, it could be possible to recommend solutions, minimize negative impact, and support decision-making (Atwal 2020).

A dashboard helps stakeholders to access reports, as they can use metrics as filters to select reports considering KPIs, tags, or target data, and gain better visualization of what happens throughout the pipeline. This tool helps with planning and decision-making. Through the user interface, company goals can be easily evaluated, and processes to improve are quickly identified (Sahid et al. 2017).

Automation and virtualization tools ensure the achievement of quality standards in the shortest time possible and make sure that developers work in a consistent environment (Ebert et al. 2016). For these purposes, cloud systems would be the best environments where to perform data analytics, as these ensure the development of centralized platforms for DataOps and provide flexibility and ease when designing architecture and processes that are based on heterogeneous data and multiple use cases (Atwal 2020).

4 Analysis - Optimizing DevOps

Software development can turn into an unmanageable mess due to relationships between the client and the development team. Multiple factors may affect the success of a project. One of these factors is communication. Successful software development processes require communication between client and development team as often these projects are under strict timelines and budgets. Limited resources require efficiency, otherwise, there could be significant delays or risk of going over budget. What causes even more problems is often the client's misunderstood role in the process. Lack of communication might result in a client not understanding the software under development or lead to weird or even impossible requirements. DevOps has the goal of improving relationships by setting out processes and tools that should enable streamlined methods of communication.

Workers are often stressed by strict schedules and overtime. Software development companies should strive for balance in their working life to prevent excessive stress for the employees. Bad management of employees leads to reduced communication with customers and therefore could directly affect the success of the project. There should be a compromise where the workplace has balance but not without sacrificing customer relationships. Therefore, a balance must be found where customer relationship nor employee wellbeing is being sacrificed. DevOps provides a set of tools for each aspect of software development. These tools can be utilized to bring more value to the customer or improve current development processes. Data provided by these tools should be utilized to improve business practices. As companies are often limited in their time and resources there should be prioritization in their adoption of DevOps. Therefore, in this research, we set out to explore how DevOps data could be used to improve businesses and have taken these restrictions into account by prioritizing the data that we display. Our goal is to build a dashboard that would quickly show the status of the software based on the metrics we have chosen.

DevOps plays a supporting role in software development. To create value for developers we must consider what aspects are the most important for them to do their job as well as possible. Software development teams need to be able to track metrics instantly. In case something happens, the DevOps team needs to respond as fast as possible. The Dashboard enables development teams to monitor their product and respond in real-time. There is inherent value in seeing the instant state of the software through a dashboard. DevOps dashboard is a reporting tool that aggregates metrics from multiple tools to create simple interface that

development team can use to monitor their product. This way DevOps dashboard enables developers to track down time or any sort of service disruption through this dashboard and therefore improving the efficiency as developers can respond in real-time. Displayed data could include things such as up and downtimes, response times, load times, errors, development work items, etc. Each metric's importance to the company varies case by case therefore our prioritization doesn't necessarily apply to all companies. In this research, our dashboard will be an example of how this sort of data tool looks like and how it could be used by companies to improve their businesses. We will also provide a prioritization matrix as a tool to conduct this data prioritization work.

A prioritization matrix is formed out of information that was discovered in the literature about DevOps. The gathered information covers all the most important stages of software development. To use this prioritization matrix, the company must define its priorities and come up with its assessment criteria. Within the context of this research, we were guided by EfiCode about what they think the most important aspects of DevOps are, and based on that interview this prioritization matrix was drawn.

Table 3. Prioritization matrix

			1	2	3	4	5	6	7	8
			Alternatives							
	Assessment criteria	Weight	Plan	Code	Build	Test	Release	Deploy	Operate	Monitor
1	Requires dialogue with customers	3	**5**	**0**	**0**	**5**	**4**	**2**	**5**	**5**
2	Complex process for management to monitor	1	**4**	**1**	**1**	**5**	**4**	**3**	**4**	**3**
3	Conflicts arise due to mis-communication	3	**5**	**1**	**4**	**3**	**0**	**0**	**2**	**3**
4	Generates a considerable amount of data	2	**4**	**5**	**5**	**5**	**3**	**5**	**5**	**5**
		Totals	42	14	23	39	22	19	35	37
		Rank	1	8	5	2	6	7	4	3

Based on the matrix above, we can conclude that the most important stages are the planning, testing, and monitoring phases. This means that the data we display in the dashboard should

reflect these stages shown in the prioritization matrix. Each stage will house its data metrics. Examples of what data are associated with each DevOps stage are listed below:

- Plan: design of the software, user requirements and documentation, information about the project environment such as user stories, tasks, events, backlog, and statistics.
- Code: all information generated from coding, version upgrades, old and new prototypes, lines of scripts, differences in versions, and other parameters.
- Build: information generated from the development stage, such as lists of files executed, packages, logs, and measures that include successes, alerts, errors in compilation, failures, and so forth.
- Test: test automation algorithms, logs with data about failed and passed tests, results, quality reports, and other documentation written throughout the software development.
- Release: information about new product launches, for instance, features, version, files, packages, logs, and metrics.
- Deploy: information about successes, failures, errors, and warnings stored in files, scripts, and logs.
- Operate: data generated from physical and virtual servers that informs how software, tools, and systems operate.
- Monitor: information extracted from users' feedback and monitoring tools that provide data such as logs and statistics (Capizzi et al. 2020).

Based on the matrix, the most important phases concerning collected data can be identified. In the case of EfiCode, these include the planning phase, the testing phase, and the monitoring phase, respectively. The planning phase is all about project management. Data that is utilized should be related to how and what the development team is working on. Good metrics include things such as development velocity or sprint capacity estimations. The testing phase is about validating and verifying software under development. Data related to testing often includes things such as unit test pass rates, integration test results, and logs about failing aspects of the software. The monitoring phase can be seen as the phase that enables planning. Often, released software runs into defects and users tend to give feedback on how the software could be improved. This way project gains list of things to work on and

these issues can be prioritized in the planning phase. In the monitoring phase software is carefully examined for these defects and improvements.

To visualize the above-mentioned points as an example a mock-up data set was created in MS Excel focusing on the planning, testing, and monitoring phases as identified in Table 3. The data set was extracted to Power BI. Here the data relationships were modeled, and data cleaned using Power Query. New measures were also created in Power Query where needed. Then data is then visualized with the help of various cards, graphs, pictures, and charts into the Planning, Testing, and Monitoring dashboards, respectively.

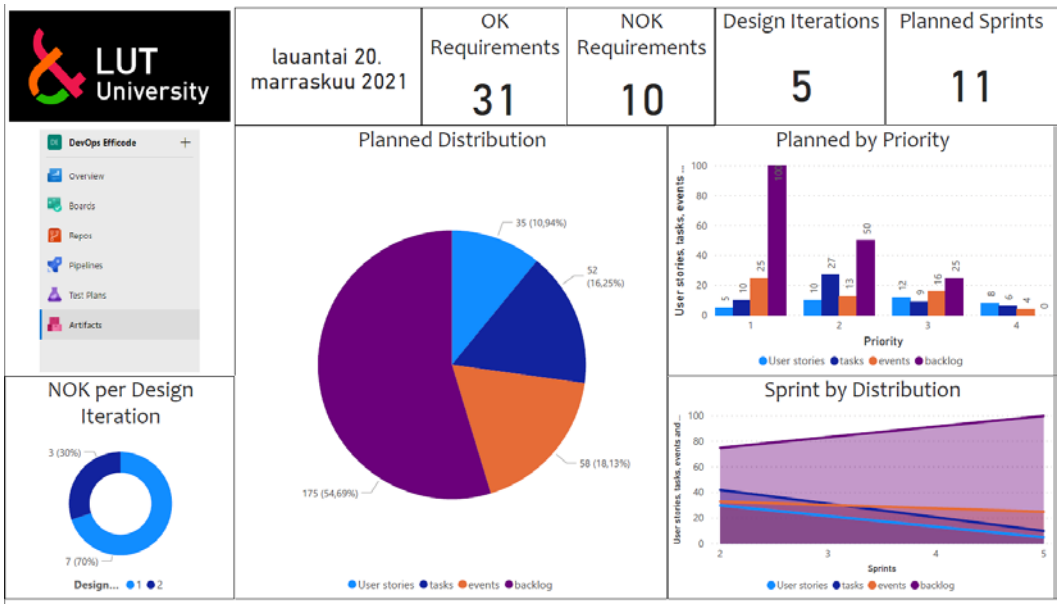

Figure 3. Planning dashboard with mock-up data in Power BI

The planning information is displayed in the Figure 3 dashboard such as planned sprints, design iterations for software, and the OK and NOK requirements. Furthermore, the different distributions of user stories, tasks, events, and product backlog by sprint and priority are displayed.

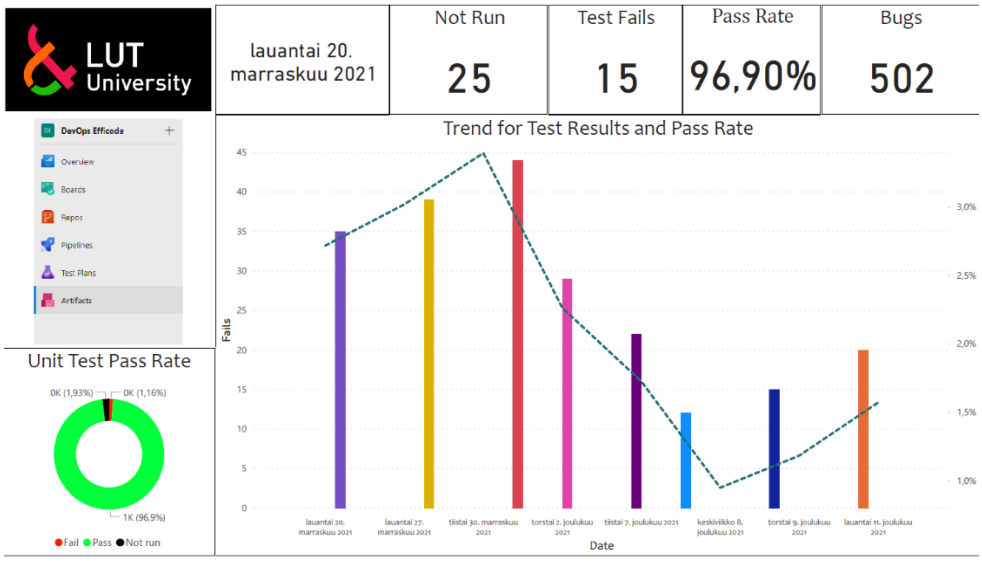

Figure 4. Testing dashboard with mock-up data in Power BI

The testing information displayed in Figure 4, indicates information such as the number of tests not run, the test fails, bugs found and the pass rate in percentage for the tests conducted. Also, a trend graph displays the trend for test results and pass rate over a certain period. A donut chart visualizes the percentages of tests that fail, pass, and do not run as a whole.

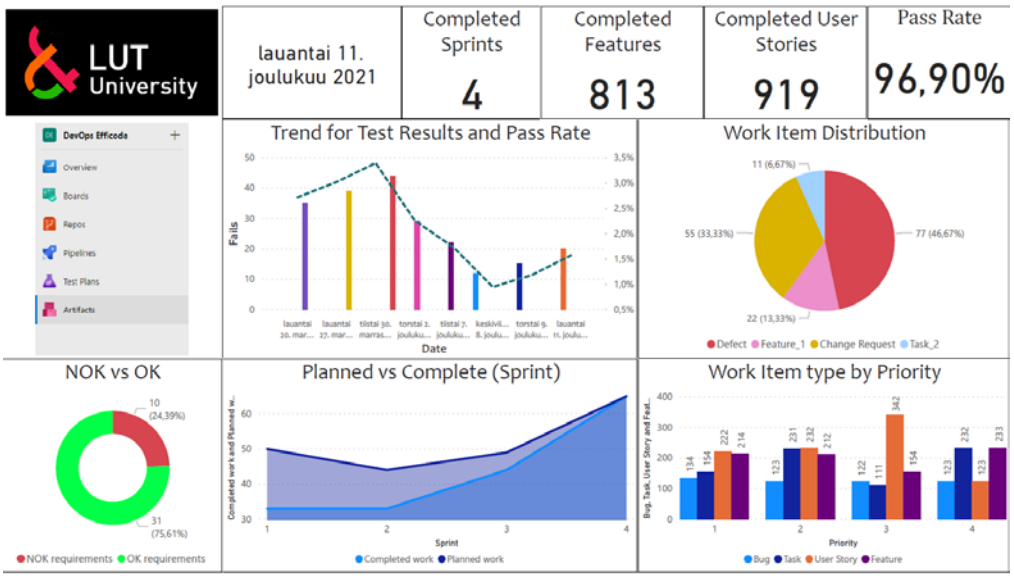

Figure 5. Monitoring dashboard with mock-up data in Power BI

The monitoring dashboard in Figure 5, an overview of DevOps status in an ongoing software development phase is displayed by the mock-up data. Here, completed sprints, features, user stories, pass rate, test results trends, NOK vs OK requirements, and various other statistics are visualized.

5 Conclusion

DevOps is becoming a more and more important aspect in the continuous deployment paradigm in both practical and academic areas. However, DevOps brings besides many opportunities also challenges that need to be considered when implemented. DevOps practices impact processes, products and services, organizational structures, associated technologies, and business practices and opportunities. Furthermore, the changes can result in organizational and business stresses. Consequently, adopting DevOps is not easy for companies. It may require that the organization introduces the process, personnel, and technological changes and innovations. Furthermore, to have a successful DevOps adoption it should be customized to the organization. Therefore, this paper identifies the main important aspects for successful DevOps in an organization.

Firstly, we identified the concept of DevOps and DataOps. For successful product development, the process of data analytics must be integrated within the organization. Thus, for managers and developers, it must be paramount to approach the business holistically. Consequently, to guarantee the efficiency of data pipelines as to ensure supply of data within multiple business units. The final goal of DevOps is to use automation to discover, access, clean, and store data. For this reason, data must be scalable and have a certain degree of accuracy.

For DevOps to work effectively and give the results, multiple challenges must be addressed. Because of the high amount of data and limited resources of companies, prioritization must be made about which data is most important. This can be done by using the conceptual framework discussed in the paper. Companies need to identify which data can bring the most value to their organization. To assist with this challenge, a prioritization matrix is established. The prioritization matrix is a tool for organizations to identify which data collection and transformation they need to prioritize to optimize DevOps within their organization. Based on the matrix, the most important phases of collected data can be identified. In the case of EfiCode, these include the planning phase, the testing phase, and the monitoring phase respectively.

It is important to note that there are also some limitations that should be considered. This paper touches the surface of the implementation and adoption process of DevOps. There are many other elements to consider when adopting DevOps, such as the culture, or

management. However, in case of needing more information about DataOps and which data is required for optimization of DevOps, this paper provides a clear overview and suitable guidelines.

References

AltexSoft. (2019). Legacy System Modernization: How to Transform the Enterprise for Digital Future. [online] Available at: https://www.altexsoft.com/whitepapers/legacy-system-modernization-how-to-transform-the-enterprise-for-digital-future/ [Accessed 23 Dec. 2021].

Atlassian (n.d.). History of DevOps. [online] Atlassian. Available at: https://www.atlassian.com/devops/what-is-devops/history-of-devops.

Atwal, H. (2020). Practical DataOps: Delivering Agile Data Science at Scale. Apress, Berkeley, CA.

Banica, L. Radulescu, M., Rosca, D. & Hagiu, A. (2017). Is DevOps another Project Management Methodology? Informatica Economică. 21(3).

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

Capizzi, A., DiStefano, S. & Mazzara, M. (2020). From DevOps to DevDataOps: Data Management in DevOps Processes. In book: Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment. pp.52-62.

Claps, G.G., Berntsson Svensson, R., Aurum, A. (2015). On the Journey to Continuous Deployment: Technical and Social Challenges Along the Way. Information and Software Technology 57, 21–31.

Ebert, C., Gallardo, G., Hernantes, J. & Serrano, N. (2016). DevOps. Software Technology. IEEE Computer Society 33(3), pp. 94-100.

Happonen, A., Siljander, V. (2020), Gainsharing in logistics outsourcing: trust leads to success in the digital era, International Journal of Collaborative Enterprise, Vol. 6, No. 2, pp. 150-175.

Leite, L., Rocha, C., Kon, F., Milojicic, D. and Meirelles, P. (2020). A Survey of DevOps Concepts and Challenges. ACM Computing Surveys, 52(6), pp.1–35.

Lwakatare, L. E. (2017). DevOps adoption and implementation in software development practice. Concept, practices, benefits and challenges. Ph. D. dissertation.

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015). Dimensions of devops. In International conference on agile software development (pp. 212-217). Springer, Cham.

Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare, L. E., Tiihonen, J., & Männistö, T. (2016). DevOps adoption benefits and challenges in practice: a case study. In International conference on product-focused software process improvement (pp. 590-597). Springer, Cham.

Ross, J.W. and Weill, P. (2002). Six IT Decisions Your IT People Shouldn't Make The Idea in Brief The Idea in Practice.

Sahid, A., Maleh, Y. & Belaissaoui, M. (2017). An Agile Framework for ITS Management in Organizations. A Case Study Based on DevOps. ACM International Conference Proceedings Series of International Conference on Computing.

Salmela, E., Santos, C., Happonen, A. (2013), Formalisation of front end innovation in supply network collaboration, International Journal of Innovation and Regional Development, Vol. 5, No. 1, pp. 91-111.

Santti, U., Eskelinen, T., Rajahonka, M., Villman, R., Happonen, A. (2017), Effects of Business Model Development Projects on Organizational Culture: A Multiple Case Study of SMEs, Technology Innovation Management Review, Vol. 7, Iss. 8, pp. 15-26.