Investigate the benefits and challenges of adopting the DevOps culture

Abdo Taleb

Dept. Computer Science & Engineering
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfilment of the requirements for the bachelor's degree in software engineering. The thesis is equivalent to 10 weeks of full-time studies

Contact Information:
Author(s):
Abdo Taleb
E-mail: abdo.taleb1@hotmail.com

University advisor:
Michel Nass
Dept. Computer Science & Engineering

Dept. Computer Science & Engineering
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Companies follow different approaches in the life cycle of software, but usually activities are divided into different teams, the most important teams are the development team and the operations team. Often the goals of the operations team and the development team are different, which results in an escalating relationship between the two teams, and a conflict occurs between teams due to the blame of responsibility on each other, which in turn leads to long delivery periods and ineffective delivery methods.

To solve these problems, the DevOps approach emerged, which is an acronym for the two words development and operations. This approach aims to align both the development team and the operations team and break the barriers between them by working as one team towards a single goal that shares everyone the responsibility. Thus, the speed of the program delivery process increases and the company provides better services to its customers.

Since DevOps is a new approach and many companies tend to adopt this approach, this study aims to investigate the benefits of the DevOps approach and the challenges involved in adopting this approach. This study summarized that the adoption of DevOps approach has an impact on the companies in several aspects. The research presents the most important challenges that the DevOps attempts to overcome, DevOps tools that helps to achieve the value of DevOps and the benefits of the DevOps approach.

Keywords: DevOps, Development and Operations, Software development life cycle, SDLC

Contents

Chapter 1

Introduction

The concept of devops is a set of principles and methodologies that in the end streamlines planning,development etc process. The main idea being to create a set of rules and guidelines that specifically focuses on delivering objectives and goals.

In order to achieve the streamlined process and fulfil objective goals, introduced are concepts such as continuous integration and planning. One of the focuses of DevOps is automation on a larger scale. By utilizing specialized processes such as CI/CD pipelines further improvements can be made. Culturally and socially teams such as the operation team and the development which typically are isolated, are introduced and collaboration between the teams is set into motion [30].

Throughout history, there was a gap between software developers and IT professionals, and this gap was causing problems for the owners of the two fields, so the practical translation of DevOps technology was a profound cultural shift in the technical field, where the fruit of the idea was for software developers and IT professionals to cooperate daily to bridge the distances between them and decode the codes for each of them, and without this cooperation the idea will not succeed [11].

1.1 Problem formulation and research purpose

The purpose of this study is to find out the important advantages of adopting DevOps, examine the difficulties that companies face while changing the way of work to DevOps Culture, identification of the most important tools of the DevOps approach that help in realizing the value of DevOps. Define the practices that must be followed in order to overcome the difficulties that will be encountered by both the Development team and operations team.

1.2 Research Questions

RQ1: What are the benefits of adopting DevOps?
RQ2: What are the practices that companies needs to incorporate while adopting DevOps?
RQ3: What are the challenges in DevOps Adoption?
RQ4: What are the ways to overcome DevOps challenges?

1.3 Motive and Value

In RQ1 DevOps companies get several advantages that compete with other companies that use legacy systems. By creating a single team that brings together cross-functional members who all work in collaboration, organizations that rely on DevOps can speed up the project completion and get more time to innovate. A study conducted by experts at Grand View Research showed that DevOps adoption increased by 17% in 2018 compared to 2017, the increase was 10%. Companies that adopted DevOps culture in software development significantly improved by 63% in the quality of software deployments [12].

In RQ2 there is a need to think about what are precisely the practices that an association needs to incorporate as a piece of the DevOps selection measure. There is no appropriate direction in existing writing about the practices and standards of DevOps. This examination question targets recognizing the practices that an association needs to incorporate while embracing DevOps.

In RQ3 the meeting of the development team and the operating and operations team in one place, it is very difficult to bring together two different areas, such as the areas of development and implementation, and because each team is accustomed to being alone independently of the other, differences may arise between them in the distribution of tasks and how to manage the project [13].

In RQ4 recognizing the systems that will help to overcome many difficulties that may face the team after adopting DevOps as a method of work. Knowing how to deal with and solve these problems may prevent the team from facing problems in the future. Consequently, it is necessary to study the methodologies that must be followed to solve or avoid problems after adopting DevOps.

Chapter 2

Theory

Foreword : The aim of this chapter is to provide the reader with an understanding of the software life cycle and DevOps approach.

2.1 Software development life cycle

Almost sixty years prior, in 1956, the principal portrayal of an SDLC model was introduced by Herbert Benington. From that point forward, SDLC models have made considerable progress [14].

SDLC is basically an abbreviation of Software Development Life Cycle (SDLC). It is considered as a process or method that covers the stages from the raw idea of a software to the arrival of the eventual outcome or so-called final software product. It is a method for planning, creating, testing as well as deploying the software. The process of software development measure comprises of time sensitive stages. In this way, the software can be created in a planned manner. These stages are handled in a ceaseless cycle, as the requirements of software functions are continually changing and growing. You can move to and fro at any phase of the cycle [15].

SDLC is viewed as the foundation for all product methodologies of software development with different activities related with each level. SDLC typically starts with deciding client business needs, trailed by execution and testing. The cycle closes with the satisfaction, all requirements considered and completed. Due to these reasons SDLC is widely used in order for development of new software product or system [16].

The software development life cycle basically comprises of seven distinct phases that are elaborated and represented as:

2.1.1 Planning Phase:

This is basically the initial stage termed as planning stage that will decide project objectives and build up a high-level arrangement for the expected task. It is considered as most essential and basic hierarchical stage [17]. The three primary activities associated with the planning phase are as per the following:

- Identification of the framework for improvement.
- Feasibility evaluation.
- Creation of project plan.

2.1.2 Analysis Phase:

End-client business requirement analysis happens during this stage. In this phase basically the goals of the project are converted into the characterized framework works that the organization plans to create [17]. The three primary activities engaged with the investigation stage are as per the following:

- Gathering business requirements.
- Creating measure charts.
- Performing a detailed analysis or investigation.

2.1.3 Design Phase:

In the design stage, we portray that features that we require and tasks of the system. This stage incorporates business rules, pseudo-code, screen formats, and other vital documentation. The two primary activities engaged with the design stage are as per the following:

- Designing the IT infrastructure.
- Designing the system model.

The IT framework ought to have strong establishments to keep away from any accident, breakdown, or decrease in execution. In this stage, the expert suggests the customers and workers needed on an expense and time premise and the framework's specialized attainability. The association additionally makes client connection interfaces, information models, and element relationship graphs (ERDs) in this stage [17].

2.1.4 Development Phase:

The development stage is the point at which all documents from the past stages transfer into the system. The primary activities engaged with the development stage are as per the following:

- Development of IT framework or infrastructure.
- Development of data set and code.

During the design stage, simply the IT framework outline is given. Conversely, the organization buys and introduces the installs the respective software and hardware to help the IT infrastructure during the development stage. Following this, creating the Database and genuine code can start to finish the system as per the particulars [17].

2.1.5 Testing Phase:

All pieces of code are incorporated during the testing stage and conveyed in the testing environment. Testers then, at that point work through SDLC activities in order to check the system for bugs, errors, faults, defects etc to confirm the system's functionalities performs as expected [17]. The two primary activities associated with the testing stage are as per the following:

- Writing test cases.
- Execution of test cases.

Testing is a basic piece of the product improvement life cycle. To give quality programming, an organization should deliberately perform testing. Subsequent to writing test cases, the tester executes them. They contrast the actual outcome and an expected outcome to confirm the framework and guarantee it works accurately. Composing test cases and physically performing them is a serious errand for any organization but will succeed whenever executed appropriately [17].

2.1.6 Deployment Phase:

During this next stage, the system is sent to a reality (the customer's) environment where actual clients can start using the system. All information and parts are available in the production environment. This stage is likewise called 'delivery' [17].

2.1.7 Maintenance Phase:

Any vital upgrades, revisions, modifications and changes are made during the maintenance stage to guarantee the system proceeds to work and remain updated in order to meet business objectives. It is important to upgrade as well as to maintain the system time to time in order to adapt future needs. The three primary activities associated with the support stage are as per the following:

- Support the system clients.
- Maintenance of System.
- System adjustments and change.

2.2 Waterfall model

Dr. Winston Royce presented the Waterfall Model in 1970. The waterfall model, that was documented in 1970 by Royce was the principal life cycle model that was ever documented publicly. This model is a famous adaptation of the SDLC for software engineering [18].

A waterfall model is that software model where steps are performed in an order sequence. Each process or step relies upon the completion of the past one. It is the least complex cycle and requires cautious bit by bit information following. This SDLC model is the most established and generally clear. With this model, we finish one stage and afterward start the following. Waterfall model is a sequential life cycle model that does not overlap with one another phase. Each stage has its own mini plan and each stage "waterfalls" into the next. The developer should finish each stage before the next stage starts. This model is named "Waterfall Model" since its diagrammatic representation looks like a cascade of waterfalls. It basically serves as a baseline for many other models [16].

2.3 The spiral model

The spiral model was characterized by the American mathematician and software engineer lecturer named as Barry Boehm. Subsequent to introducing his idea in 1986 for the advancement of complex applications, he published his model in 1988 in a more far-reaching structure in his article "A spiral model of software advancement and improvement"[1].

Some portion of this 1988 publication portrayed the spiral model graphically, appearing completely what the development process of software resembles in a

twisting or spiral design and upheld by cycles. In his publication, Boehm depicted it in contrast to the recently developed waterfall model, which also served as the reason for his profession, he considered the spiral model as a viable option [1].

The spiral model was not quick to examine repeating development, but rather it was the primary model to clarify why iteration is significant. As initially planned, it has been focused on everywhere, complex projects whose iterations regularly range from a half year to 2 years [1].

IEEE characterizes the spiral-model as a model that can be used in software development process in which the constituent tasks or activities, such as typical requirements assessment or analysis, preliminary and thorough configuration, integration, testing as well as coding are performed out repeatedly until the software is finished or the final result is ready to deploy [2].

For large and high-risk projects, spiral model is one of the most used SDLC models. The complete development process is separated into four stages, and the entire process is depicted in a spiral diagram, which is referred to as the spiral model. It is also known to be based on evolutionary development since the primary and critical elements are designed, developed, and deployed initially, followed by the addition of other required features as the project progresses through the spiral or phases. Because the process of setting objectives, analyzing risks, developing software, and planning is done separately for each phase of the project, the spiral model of software development is noted for being extremely adaptable. With each iteration of the project, it also allows for improvements. Furthermore, it is extremely risk-driven, as the project's success is contingent on effective risk assessments. As a result, having specialized risk analysis knowledge at each phase is critical [3], [4].

Basically, the development process in the SDLC Spiral model begins with a small set of requirements and progresses through every development phase for those requirements or set of criteria. In ever-increasing spirals, the project team that is software engineering team here adds functionality for that increased requirement until the application would get for production or development phase [5].

2.4 Agile

In February 2001, 17 senior software development engineers gathered at a resort in Snowbird, Utah, USA. These engineers sought to find ways to quickly create and deliver work programs to users. The meeting participants came up with the formulation of what is known as an "Agile Manifesto", which is considered a

paradigm shift in the field of software management and production. The values and principles of "agile" have strengthened reliance on the human ability to learn from their experiences and benefit from the collective intelligence of the team to achieve efficiency and effectiveness in production [19].

2.4.1 The Agile Principles, as stated in [19]:

- Gaining customer satisfaction by providing an effective and usable program (product) and adhering to delivery times.
- Acceptance of changes in requirements by the customer, even if it is at an advanced stage of development.
- Delivery of usable software in the shortest possible time at regular intervals.
- Programmers and technicians should work closely and with one another daily throughout the life of the project.
- Face-to-face conversations between team members are the best and fastest way to transfer information between the team (usually team members meet daily in the morning Stand-up Meeting for 10-15 minutes).
- Project construction should be based on enthusiastic individuals. Individuals who are provided with the appropriate environment and the necessary support and give them confidence to accomplish the required work with passion and enthusiasm away from the routine of traditional work.
- The only way to measure the progress of any project is only to have software, applications and tools that can be used by users, not just performance reports, holding periodic meetings, preparing technical or administrative documents, or the like.
- Agile encourages the continuous development of individuals and their skills and knowledge. Developers should be able to always maintain a steady progression rate.
- Constant attention to excellence and quality in technical development and design.
- Simplicity is an essential, vital, and important part of agile, meaning minimizing unnecessary and unimportant work.
- Self-organizing teams, offering the best requirements, structure, and design.
- Teams assess and monitor their work to be more effective, then adjust errors and behaviour at regular intervals.