GRADUATE SCHOOL OF
SOCIAL SCIENCES

From Agile to DevOps , Holistic Approach for Faster and Efficient Software Product Release Management

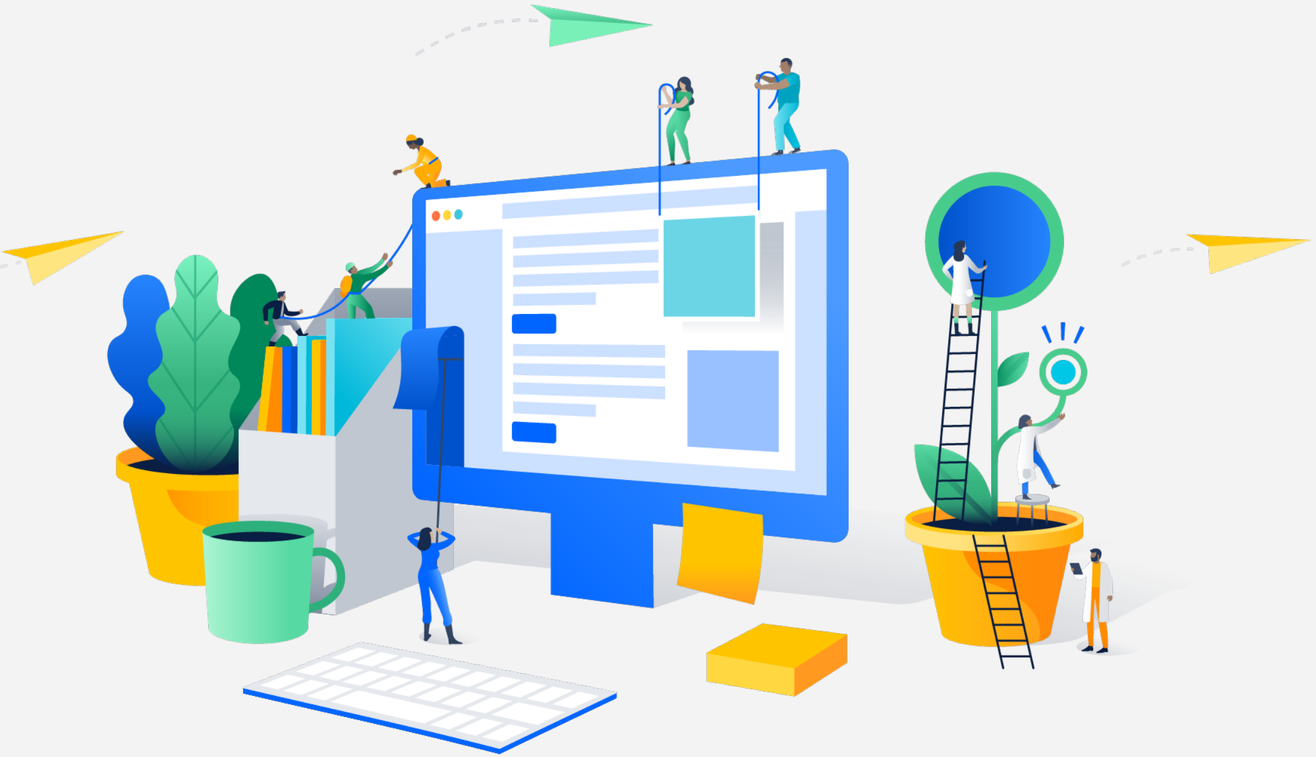


Table of Content

Introduction

Release management has an important place in the total development, test, analysis, and builds stages in software projects. Effective release management ensures that all these stages and the product to be released are faster, more effective, accurate, and manageable. To achieve this, many new technologies, tools, philosophies and practices continue to be researched and developed, especially in recent years.

Faster

The speed of release management depends on the manageability of the project and processes.

Efficient

The effectiveness of release management ensures that the software project is timely, efficient, and meeting the requirements.

Sustainable

Sustainable, integrated, reusable, sustainable software release management is important for software in the component model.

Easy to Use & Adopt

The fact that release management is easy to use and adopt is important for the integration of new developers, teams, and users.

Current Studies & Literature

We can gather studies conducted in the field under these headings in general.

1 Agile to DevOps Release Management

From Agile Development to DevOps: Going Towards Faster Releases at High Quality – Experiences from an Industrial Context (Elberzhager, F., Arif, T., Naab, M., Süß, I., & Koban, S. 2017, January).

2 DevOps Release Management

Towards Definitions for Release Engineering and DevOps (Dyck, A., Penners, R., & Lichter, H. , 2015)
DevOps: Advances in Release Management and Automation (Azoff, M. , 2011).

3 Change & Release Management

Methods and systems for software release management (Barsevksy, 2004) Change and release management system (Carroll, 2006)

Current Studies & Literature

We can gather studies conducted in the field under these headings in general.

4

Release Planning in Agile

Agile Release Planning: Dealing with Uncertainty in Development Time and Business Value (Logue, McDaid, 2008)

5

Software Development Lifecycle

Evolving a New Software Development Life Cycle Model (SDLC) incorporated with Release Management (Massey, 2012) Software development lifecycle models (Ruparelia, 2010)

6

Release Management

Release engineering processes, their faults and failures (Wright, 2012) Software Release Management Evolution - Comparative Analysis across Agile and DevOps Continuous Delivery (Mohamed, 2016)

General Summary of the Literature

When we look at the literature, we see that the studies in this field generally focus on the computer science category and related categories. We see that all of the 25 articles scanned by the web of science under the common title were written after 2014. In a research area where quite new fields and technologies are common, we can think that this is quite normal.

Besides, when we take a general look at the literature, we can say that the studies conducted in this field generally reveal these technologies, examine their application practices, examine their studies in different sectoral disciplines, and draw a general framework.

We can summarize these researches as follows;
DevOps culture extends the agile methodology to rapidly create applications and deliver them across the environment in an automated manner to improve performance and quality assurance. Continuous Integration (CI) and Continuous Delivery (CD) has emerged as a boon for traditional application development and release management practices to provide the capability to release quality artifacts continuously to customers with continuously integrated feedback. (Soni, 2015)

Problem

- What Model Should Be Established for Faster and Efficient Software Product Release Management?
- What are the Elements of the Model?
- How to Express the Practical Application of the Model from End to End?

Goals & Objectives

The main purpose of this research is to present a model that predicts how to do release management, which is a very serious step and is the principle of software product management, in an effective, correct and sustainable manner.

6

An Efficient Model Suggestion

To present an effective model in this field and to present this model from a general perspective.

Introducing Latest Technology and Methods

To summarize relatively new practices such as DevOps, Agile, CI / CD and their use in release management.

Developing a Holistic Approach

To demonstrate a holistic and end-to-end approach

Researching Applicable Practices

To provide grounds for future studies by researching applicable practices

Software Development Life Cycle

What is?

There are several ways to develop systems. The classical approach is called the Software Development Life Cycle (SDLC). This consists of the stages: Software Requirement Analysis (with Prototyping), Design, Coding, Testing, and Maintenance. (Jana, 2014)

Processes

Different software development life cycle (SDLC) methodologies follow different processes, the most important ones are detailing functional requirements, system design, implementation, testing, deployment, and maintenance. (Vemula, 2017)

Different Models

Software development follows the software development life cycle (SDLC). SDLC models include Waterfall, Spiral, Agile, and RAD. (Chapple, 2017)

Change and Release Management

Release Management: A formal release process for nonemergency corrective, perfective, and adaptive projects. (Lewis, 2017)

Change management is broader, tracking changes across an entire software development program. (Conrad, Misenar, Feldman, 2012) Change management is important because it's critical to keep track of the version of the software and related documents that are being tested.

Latest Methods & Principles

New technologies, approaches and their implications for effective release management in the Release Management process

1

Agile

Agile projects release their most valuable features first and release new versions frequently, which dramatically increases value. (Shore, 2007)

2

DevOps

DevOps includes the release cycle, so quality assurance (QA) aspects are important in this area as well. (Rossberg, 2014)

3

CI/CD

Continuous Integration then needs Continuous Deployment, a culmination of practices and steps which enable us to release working software any time, any place, with as little effort as possible. (Duvall, 2007)

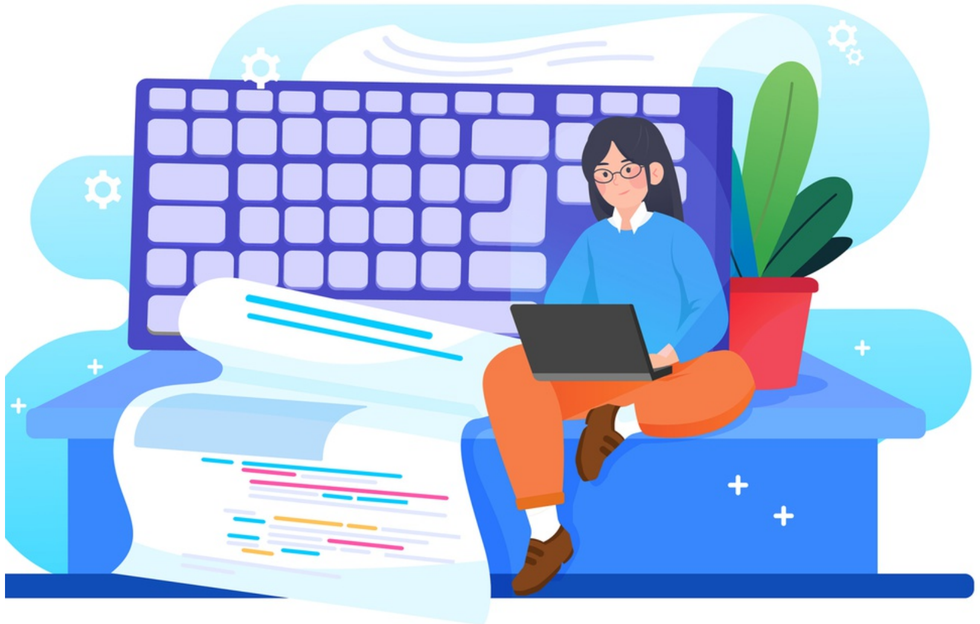
Swiftness & Efficiency in Agile

Agile provides advantages to software teams in terms of less document management, more agile processes, shorter versions and speed and interaction. Agile information systems and software methods are characterized by nimbleness to rapid changes, multiple incremental iterations and a fast development pace (Abrahamsson et al., 2003). DevOps contributes to an appropriate release process in terms of both speed and efficiency. One of the core goals of DevOps is to achieve maximum efficiency with an application delivery pipeline by optimizing the pipeline. (Sharma, 2017)

Flow Of Model Suggestion

In the field of software, we see that Agile methods are becoming widespread in almost all projects and sectors due to reasons such as frequent publication management, iterative development, and efficiency in small-sized projects.

Besides, we see that the effect of end-to-end and continuous automation is increasing with the increase in cloud computing, computing power, and the emergence of new tools and technologies, especially of CI / CD and DevOps practices. The largest technology companies and tools today are concentrated in this area. Example: Google Cloud, AWS, Microsoft Azure, etc.



Flow Of Model Suggestion

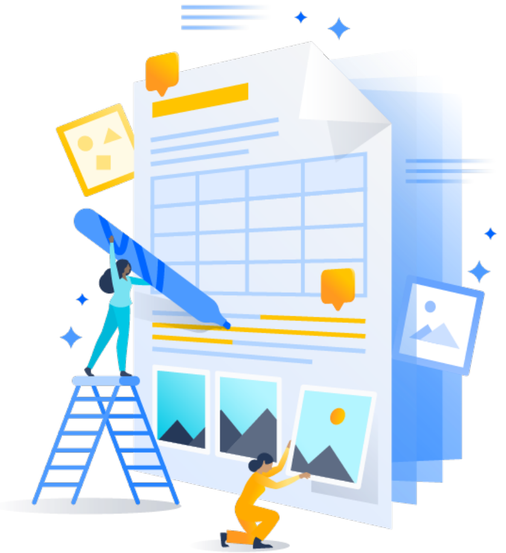
Practices in this area should mainly include:

* Application of Agile Methods:
Agile methods should be applied to improve the publication management in a healthy way, especially in projects that require continuous improvement, uncertain scale, team interaction, and customer focus.

* Automation and DevOps:
DevOps, which is a philosophy rather than a practice, should be applied with all its practices. Because DevOps makes the release management more effective, more manageable, scalable, fast, and sustainable. It also has a significant impact on productivity.

* Change Management, Build, Integration, and Processes:

Effective change management, configuration, and process management underlie the efficiency of release management. Besides, quality and test activities must be defined in place and correctly, and that they are set up following Agile and DevOps practices. Thus, end-to-end, automated release management with established practices emerges. While problems such as bugs, bugs, version errors are avoided, efficiency problems are reduced.

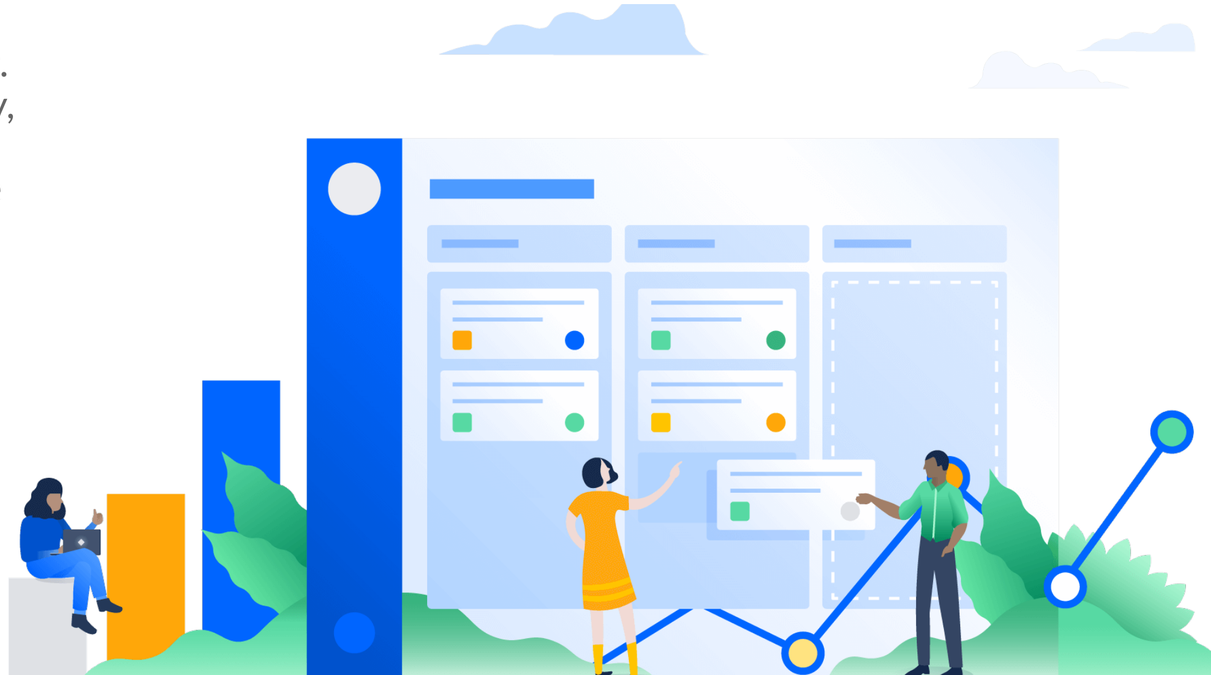


Important Notes for Model

We can express the important points for our model as follows. The most important expression of the model is agile software development, which is a creative method of software development, DevOps, and release management, which offer an automated and holistic development and operation approach, together. Therefore, an end-to-end process must be designed.

The most important advantages of this process are sustainable quality, iterative development, adaptation, speed, competitiveness, fast problem solving, flexibility, efficiency, and low turnover, as planning and execution from agile software development are together.

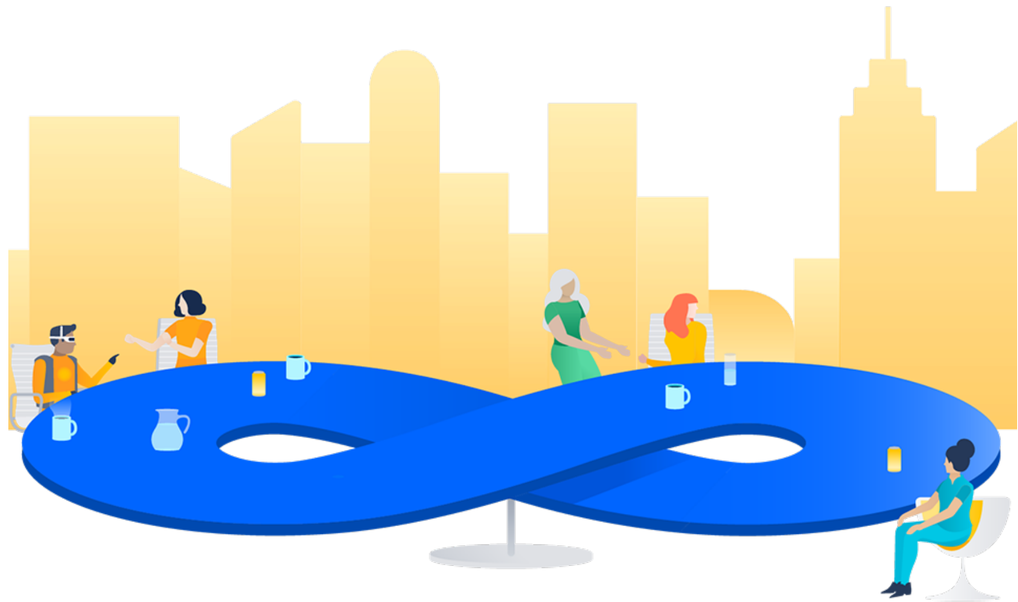
Also, the model basically includes the advantages of DevOps development. These advantages can be listed as speed, fast delivery, reliability, scalability, collaboration, and security. All these provide efficiency to the model especially in certain sectors (SaaS, Startup, Informatics, Low-Medium Scale Software Projects).



Verbal Expression's of The Model

We can express the model verbally as follows. You can also find its visual expression on the next slide:

1- System requirements are collected from users and customers. These requirements are kept integrated with requirement management systems.
2- Business and software analysis processes are improved, business requirements evolve into functional requirements.
3- Sprint plans are started about the requirements, issues are designed in story and epic-sized backlog.
4- CRs are created for relevant updates and defects.
5- It is determined that the CRs regarding the project manager, product manager, scrum master meetings will be approved and developed.
6- The system is coded and developed.
7- Developed codes are tested.
8- Authorization of QA, Configuration, and other stakeholders is provided.
9- The code is deployed to the relevant environment. Integration of CI / CD and DevOps processes is important at these stages.
10- Post-implementation reviews are made, certain audits are made.
11- Relevant publication and CRs are closed, the release is completed and the release is released. Since this version leaves DevOps processes, it is integrated, tested, and open to traceability.



Flow Of Model Suggestion

Acknowledgements & Future Studies

We can gather what has been learned from our study and what will be done in future studies under the following headings and focal points.

The Impact of the Model is Researchable

The efficiency and effects of the model can be discussed in accordance with certain scales and impact analysis methods.

Different Applications of the Model Can Be Investigated

Approaches including similar models, agile applications and devops practices in different sectors can be put forward.

Similar models and their efficiency can be scaled

The effectiveness of similar models can be scaled, these activities can be demonstrated and tested with scales.

Different models and comparisons can be examined

The differences, similarities and effects of the model with different models can be compared and revealed.

References

- Soni, M. (2015, November). End to end automation on cloud with build pipeline: the case for DevOps in insurance industry, continuous integration, continuous testing, and continuous delivery. In 2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM) (pp. 85-89). IEEE.
- Azoff, M. (2011). DevOps: Advances in release management and automation.
- Dyck, A., Penners, R., & Lichter, H. (2015, May). Towards definitions for release engineering and DevOps. In 2015 IEEE/ACM 3rd International Workshop on Release Engineering (pp. 3-3). IEEE.
- Mohamed, S. I. (2016). Software Release Management Evolution-Comparative Analysis across Agile and DevOps Continuous Delivery. International Journal of Advanced Engineering Research and Science, 3(6), 236745.
- Barshefsky, A., Ramakrishnan, M., & Xiao, Q. (2005). U.S. Patent Application No. 10/810,207.
- Massey, V., & Satao, K. J. (2012). Evolving a new software development life cycle model (SDLC) incorporated with release management. International Journal of Engineering and Advanced Technology (IJEAT), 1(4), 25-31.
- Mohammad, S. M. (2019). DevOps Automation Advances IT Sectors with the Strategy of Release Management. International Journal of Computer Trends and Technology (IJCTT)–Volume, 67.
- Shore, J. (2007). The Art of Agile Development: Pragmatic guide to agile software development. " O'Reilly Media, Inc.".
- Conrad, E., Misenar, S., & Feldman, J. (2012). CISSP study guide. Newnes.
- Duvall, P. M., Matyas, S., & Glover, A. (2007). Continuous integration: improving software quality and reducing risk. Pearson Education.
- Van Der Hoek, A., Hall, R. S., Heimbigner, D., & Wolf, A. L. (1997). Software release management. ACM SIGSOFT Software Engineering Notes, 22(6), 159-175.