IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

Volume 3, Issue 2, February 2023

Impact Factor: 7.301

Adopting Continuous Integration Practices to Achieve Quality in DevOps

Kavita Dhakad
Assistant Professor
Indira College of Commerce and Science, Pune, Maharashtra, India
kavita.dhakad1@gmail.com

Abstract: *DevOps phenomenon is gaining popularity through its ability to support continuous value delivery. Every software process transition, technical practices has its own challenges so as DevOps. The aim of this study is to systematically review and analyse challenges confronted and practices adopted in continuous integration practice of software development to improve quality of software in DevOps. We have done systematic literature review of 44 papers. We conclude 26 challenges and 28 practices which are majorly in the continuous practices, automation, tools, monitoring and pipeline. These practices are having evidences of improving quality through faster release, monitoring performance, reduced risk, reduced testing time and efforts, improved security, fast feedback loop. In further research we have to measure the quality factors by using case study methods on selected software applications to quantify impact of presented practices.*

Keywords: DevOps, Continuous Integration, Continuous Practices, Continuous Software Engineering

I. INTRODUCTION

In the software development process, software process models plays very vital role. The process models are implemented to manage various concerns associated with cost, time, and quality and changing requirements of client's etc. The Agile is now the leading method used today for software development. The key characteristics of agile admiration are adapting change, rapid delivery and constant user involvement is presented by (Haraty & Hu, 2018). (*Business 4.0., 2019)*, Organisation on the road to Business 4.0 have found that adopting agile methodologies gives them quick wins that evidence the further transformation. Agile software development adopts Agile Manifesto presented by (Kent, 2001) and his team, agile method generally value individuals and interactions over processes and tools, deals with working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a software delivery plan.

In the Harvard Business Review(Darrell, 2018), witnesses that by scaling up agile brings values and principles to business operations, support functions, leads to greater efficiency and productivity, better financial results, greater customer loyalty and employee engagement. Agile is widely embraced due to its success factors in the category of people and organisation as customer satisfaction, customer collaboration, customer commitment, decision time, corporate culture, control, personal characteristics, societal culture, and training and learning, (Misra et al., 2009).In his survey (Kurapati et al., 2012) discovered that 89% respondents agreed that agile practices increased productivity, 90% strongly agreed that customer has given rapid feedback and 83% are satisfied with the output through frequent deliveries. Agile transformation observed reduction average cycle time per story, increase in average team throughput and improved team efficiency these results are presented in research article (Randolph, 2019).

Agile software development has broken down some of the silos between requirements analysis, testing and development. But deployment, operations and maintenance are other activities which have suffered a similar separation from the rest of the software development process. The DevOps movement is aimed at removing these silos and encouraging collaboration between development and operations. DevOps provides a pragmatic extension for the current agile activities. Agile methods can be considered as enablers to adopt DevOps thinking. The term "DevOps" was first introduced in 2009 when Patrick Debois launched the "DevOps days" event in Ghen, Belgium. The constantly changing business needs and the requirement for faster time to market with software of present day has created a paradigm shift towards a 3rd generation Software Development philosophy called DevOps. DevOps has continued to grow and in

2014 we saw the increased expansion of DevOps into enterprise environments marked by the launch of the DevOps Enterprise Summit, included in article presented by (Christopher & Sean, 2019).The DevOps phenomenon is gaining popularity through its ability to support continuous value delivery and ready accommodation of change. Agile can support DevOps by encouraging collaboration between team members, automation of build, deployment and test, measurement and metrics of cost, value and processes, knowledge sharing and tools.

DevOps combines agile methodologies with a purpose of creating seamless workflow from development to operations using continuous integration (CI), continuous deployment (CD), continuous delivery (CDE) and continuous feedback mechanisms. Continuous practices are expected to provide several benefits such as: (1) getting more and quick feedback from the software development process and customers; (2) having frequent and reliable releases, which lead to improved customer satisfaction and product quality; (3) through CD, the connection between development and operations teams is strengthened and manual tasks can be eliminated discussed in the article (Leppänen et al.,2015 and Chen, 2015)

Agile software development principles, values and practices are required forsuccessful adoption of DevOps eventually includes ability to release software quickly, frequently andwith improved quality(Lwakatare et al., 2016). Agile product management practices had positive impact on both software delivery performance and organizational performance(Nicole et al., 2018).

Automation is a cornerstone of the DevOps movement and facilitates collaboration(Perera et al., 2016). Automating Continuous Integration, Deployment and Delivery (CIDD) for adapting to DevOps culture has its own advantages to the business development process still very few companies automated partially or fully to the CIDD practice due to the lack of labours and knowledge of tools and environment is perceived in the research (Poornalinga & Rajkumar, 2016). Since 2019, companies has started implementing a programmatic DevOps approach to accelerate the development and deployment of software products Manual DevOps is time-consuming, less efficient, and error-prone. (Danave, 2019), predicted that in 2019, CI/CD automation will become central in the DevOps practice. In the report (Nicole et al., 2018) witnessed those technical practices like continuous delivery reduces the risk and cost of performing release which will be referred as the roadmap to achieving higher software delivery performance.

DevOps builds quality into the entire software delivery chain by laying emphasis on communication, collaboration, and integration among various stakeholders in the software development process, i.e. development, QA, and operations. John Willis and Damon Edward introduces CAMS model states that Culture, Automation, Measurement and Sharing are depicted as four pillars of DevOps.The researcher (Perera et al., 2017), identified that quality of the software gets improved when practice DevOps by following CAMS (Culture, Automation, Measurement, Sharing) framework. (Elliot, 2014), Suggested DevOps teams should consider the business metrics to communicate success, in this he is explaining the quality metric as improved availability, deeper requirements analysis, early business stakeholder support and involvement, security and compliance risk reduction, and identifying issues earlier through continuous testing and integration

Every software process transition, technical practices has its own challenges so as DevOps. These challenges are identified by various researchers categorised in different capacities as technical (CI, CD, Quality Assurance, Security etc.), team, organisational and social. Researchers also suggested practices, models, framework and pipeline to overcome these challenges. Further there are some research papers guiding on DevOps Metrics to measure the DevOps success are studied for literature review and systematically presented in this article.

Due to the growing importance of continuous practices, an increasing amount of literature describing approaches, tools, practices, and challenges has been published in the literature. The aim of this study is to systematically review and analyse challenges confronted and practices adopted in continuous integration practice to improve quality of software in DevOps. The research divided in four sections: introduction, Research Method, Discussion and last section is Conclusion.

II. RESEARCH METHOD

Researcher used Systematic Literature Review (SLR) that is one of the most widely used research methods for Software Engineering. SLR purposes a well-defined process for identifying, evaluating, and interpreting all available evidence relevant to a particular research question (Kitchenham & Charters ,2007). The SLR research method involves three main stages: defining a review protocol, conducting a review, and reporting a review. In this research we are following the SLR guidelines reported in (Kitchenham & Charters ,2007), our review procedure consisted of: (i) research questions (ii) search strategy (iii) inclusion and exclusion criteria (iv) study selection (v) data extraction and synthesis. We discuss these steps in the following sections:

2.1 Research Questions

This study aimed at summarizing the current research on continuous integration, challenges confronted and practices adopted in software development to improve quality of software. We formulated a set of research questions (RQs) to be answered through this report. Table 1summarizes the research questions. The answers to these research questions can be directly linked to the objective of this SLR: Identifying the challenges in CI (RQ1), further identify best practices to overcome these challenges (RQ2)and find the best practices to improve quality of software in CI process of DevOps(RQ3).

The results of these research questions would enable researchers to identify the research gaps in this area and practitioners to consider the evidence-based information about continuous integration.

In this SLR, we consider process, practices, techniqueand challenges as a technical and formalized approach to facilitate in software development process and support continuous integration.

Table 1: Research Questions and Its Motivation in the Study

RQ#	Research Question	Motivation
RQ1	What are the challenges in Continuous Integration (CI) process of DevOps?	To ascertain challenges in CI
RQ2	What are the best practices to overcome these challenges in Continuous Integration (CI) process of DevOps?	Use of best practices to overcome these challenges in CI
RQ3	Which practices are considerable to improve quality of software in CI process of DevOps?	To adopt CI best practices to improve Quality of software

2.2 Search Strategy

In order to retrieve as many as possible relevant studies, we defined a search strategy. The search strategy used for this review is designed to consist of the following components:

A. Search Term

We formulated our search terms with the help of guidelines provided in (Kitchenham & Charters ,2007). The resulting search terms were composed of the synonyms and related terms about continuous integration, DevOps, challenges and practices. After running a series of pilot searches and verifying the inclusion of the papers that we were aware of, we utilized the final search string as presented in the following. It should be noted that the search terms were used to match with paper titles, keywords, and abstracts in the digital libraries.

Scopus Search Key

TITLE-ABS-KEY ((**"DevOps"** AND **"Continuous Integration"** OR **"CI"**) AND (**"Challenges"** OR **"Practices"**))

Web of science Search Key

TS=((("DevOps" and "continuous integration" or "CI")and("Challenges" or "Practices")))

III. DATA SOURCES

We executed search query on digital libraries, Scopus, Web of Science and Google Scholar for retrieving the relevant papers. These are the primary sources of literature for potentially relevant studies on software and software engineering. For all these libraries, we ran our search terms based on title, keywords and abstract. It is worth noting that Google Scholar was selected as data source because of consideration of company experience report, article and conference papers. We found enormous articles in preliminary scanning which is presented in Table 2.

Table 2: Number of articles Obtained after preliminary scanning

Digital Library	#papers
Scopus	92
Web of Science	53
Other Sources	40
Total	185
(-)Duplicates	52
Total	**133**

3.1 Inclusion and Exclusion Criteria

Table 3presents the inclusion and exclusion criteria, which were applied to all studies retrieved from digital libraries. We chose a specific time frame year 2014 to 2020 of the search period.DevOps has continued to grow and in 2014 we saw the increased expansion of DevOps into enterprise environments marked by the launch of the DevOps Enterprise Summit, included in article presented by (Christopher & Sean, 2019). The database search results shows 98% of the articles are published 2014 onwards.

Only peer-reviewed papers were included, and we excluded editorials, tutorial summaries, panel discussions and non-English studies. We selected only those papers that have reported the DevOps or CI and challenges or practices using empirical research methods such as case study, experience report, industrial reports and review articles. In cases where we found two papers addressing the same topic and have been published in different venues (e.g., in a conference and a journal), the less mature one was excluded. We eliminated duplicate studies retrieved from different digital libraries.

Table 3: Inclusion and Exclusion for this SLR

Inclusion Criteria	
I1	A study that is peer-reviewed and available in full-text.
I2	A study that presents and challenges practices associated with CI practice of DevOps
I3	Empirical study: a study that evaluates, validates, or investigates the proposed practices through qualitative and quantitative research methods
I4	A Systematic Review papers which presents challenges and practices of Continuous integration in DevOps
I5	A study published in Research Journals, Industry reports , experience reports and Conference proceeding is considered
I6	A study published from year 2014 to 2020
Exclusion Criteria	
E1	A Study which is not in English
E2	Non peer-reviewed papers such as editorials, position papers, keynotes, tutorial summaries, and panel discussions.
E3	A study which is not related to software development process

3.2 Study Selection

The Following steps show the number of studies retrieved at each stage of this SLR. The inclusion and exclusion criteria were used to filter the papers in the following way:

Step 0: We ran the search string on the digital libraries namely Scopus, Web of scienceand Google Scholar. We finally found 133 potential papers.

Step 1: We filtered the papers by reading title and keywords. When there were any doubts about the retrieved papers and it was not possible to determine the papers by reading the titles and keywords, these papers were transferred to the next round of selection for further investigation.

Step 2: We looked at the abstracts and conclusions of the retrieved articles to ensure that all of them were related to the objective of our SLR.

Step 3: In the last (third) selection round, we read the full text of the selected studies from second phase and if a paper met all the inclusion criteria, this paper was selected for inclusion in this SLR. Finally, we selected 44for this review.

Research Method	**#Count**
Validation	16
SLR	15
Experience Report	5
Survey	7
Framework	1
Total	**44**

Table 4: Research Methods Used in SLR Paper

3.3 Data Extraction and Synthesis

Distribution of Article according to Research Methodology

The selected research article for SLR used various research methodologies as:

1. Validation of method, process, framework through case studies.
2. Peer reviewed SLR papers from reputed Journals
3. Experience Report from their work experiences
4. Survey method to collect quantitative data
5. Framework to present the research work

The major contributed research articles were used validation, SLR and Survey as the research methodology in their research.

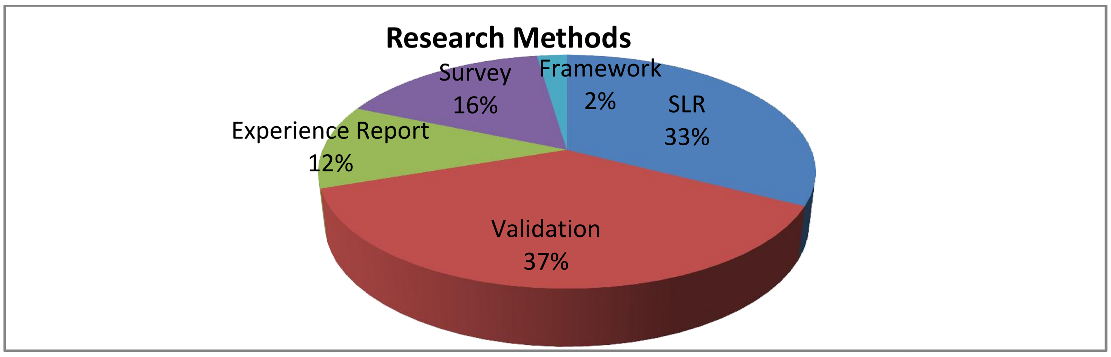

Figure 1: Research Methodology used in SLR Papers.

Research article contribution in Research Questions

There are 44 research article selected for study. The Table 6 shows contribution of each article as per research questions.

Table 5 : Research article showing contribution in proposed research

Code	**Title**	**RQ1**	**RQ2**	**RQ3**
S1	Roche, J. (2013)		✓	✓
S2	Ståhl, D., & Bosch, J. (2014)		✓	✓
S3	Fitzgerald, B., & Stol, K. J. (2014)		✓	✓
S4	Eck, A., Uebernickel, F., & Brenner, W. (2014)		✓	
S5	Fitzgerald, B., & Stol, K. J. (2015)	✓	✓	
S6	Rathod, N., & Surve, A. (2015)		✓	✓
S7	Gottesheim, W. (2015)		✓	✓
S8	Lai, S. T., & Leu, F. Y. (2016)		✓	✓
S9	Poornalinga, K. S., & Rajkumar, P. (2016)		✓	✓
S10	Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016)		✓	✓
S11	Perera, P., Bandara, M., & Perera, I. (2016)		✓	✓

S12	Hilton, M., Tunnell, T., Huang, K., Marinov, D., & Dig, D. (2016)	✓	✓	✓
S13	Zhu, L., Bass, L., & Champlin-Scharff, G. (2016)	✓	✓	✓
S14	Kumar, D., & Mishra, K. K. (2016)	✓	✓	✓
S15	Rahman, A. A. U., & Williams, L. (2016)		✓	✓
S16	Shahin, M., Babar, M. A., & Zhu, L. (2017)	✓	✓	✓
S17	Ståhl, D., Hallén, K., & Bosch, J. (2017)	✓	✓	✓
S18	Elberzhager, F., Arif, T., Naab, M., Süß, I., & Koban, S. (2017)		✓	✓
S19	Perera, P., Silva, R., & Perera, I. (2017)	✓	✓	✓
S20	Bou Ghantous, G., & Gill, A. (2017)		✓	✓
S21	Heinrich, R., van Hoorn, A., Knoche, H., Li, F., Lwakatare, L. E., Pahl, C., ... & Wettinger, J. (2017).	✓		
S22	Gupta, V., Kapur, P. K., & Kumar, D. (2017)		✓	✓
S23	Karvonen, T., Behutiye, W., Oivo, M., & Kuvaja, P. (2017)		✓	✓
S24	Vasanthapriyan, S. (2018)		✓	
S25	Arachchi, S. A. I. B. S., & Perera, I. (2018)		✓	✓
S26	Senapathi, M., Buchan, J., & Osman, H. (2018*)*	✓	✓	✓
S27	Laukkanen, E., Paasivaara, M., Itkonen, J., & Lassenius, C. (2018)	✓	✓	✓
S28	Poth, A., Werner, M., & Lei, X. (2018)		✓	✓
S29	Herbst, N., Bauer, A., Kounev, S., Oikonomou, G., Eyk, E. V., Kousiouris, G., ... & Iosup, A. (Eds.). (2018)	✓		
S30	Haghighatkhah, A., Mäntylä, M., Oivo, M., & Kuvaja, P. (2018)		✓	✓
S31	Agarwal, A., Gupta, S., & Choudhury, T. (2018)	✓	✓	
S32	Wikström, A. (2019)	✓	✓	✓
S33	Kowzan, M., & Pietrzak, P. (2019)	✓	✓	✓
S34	Tegeler, T., Gossen, F., & Steffen, B. (2019)	✓	✓	✓
S35	Rütz, Martin. (2019)		✓	✓
S36	Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2019)		✓	
S37	Luz, W. P., Pinto, G., & Bonifácio, R. (2019)		✓	✓
S38	Imtiaz, J., Sherin, S., Khan, M. U., & Iqbal, M. Z. (2019)	✓	✓	✓
S39	Ibrahim, M. M. A., Syed-Mohamad, S. M., & Husin, M. H. (2019)	✓	✓	✓
S40	Y. Wang, M. Pyhäjärvi and M. V. Mäntylä.(2020)		✓	✓
S41	Lima, J. A. P., & Vergilio, S. R. (2020)	✓	✓	✓
S42	Khan, M. O., Jumani, A. K., & Farhan, W. A. (2020)	✓	✓	✓
S43	Mishra, A., & Otaiwi, Z. (2020)		✓	✓
S44	Gokarna, M. (2020)	✓	✓	

The researcher have analysed from table 6 that total 19 articles reported challenges in Continuous Integration (CI) process of DevOps, practices to address these challenges is discussed in 41 articles and the practices to improve quality is addressed in 36 articles. Whereas 15 articles discussed all the questions as challenges, practices in continuous integration and practices which can improve quality in DevOps.

IV. DISCUSSION

The objective of this research is to identify challenges, best practices in Continuous Integration phase and study which of these practices will improve quality of the software in DevOps. Continuous integration is imperative part of DevOps and improving quality is the integral of DevOps lifecycle.

4.1 DevOps Lifecycle

DevOps lifecycle is a software development lifecycle which comprises with set of continuous software engineering activities required for software development. The term Continuous software engineering is introduced by Fitzgerald & Stol and explained 13 continuous software engineering activities(Fitzgerald & Stol, 2014).
Further, many researchers and professionals come up with the numerous continuous software engineering activities used in DevOps. For this discussion we will consider seven continuous software engineering activities as DevOps lifecycle phases(Amol, 2020).

4.2 DevOps lifecycle phases as given below:

A. Continuous Development

The first phase of the DevOps lifecycle is where the planning and software coding takes place. The planning involves understanding the vision of the project and foreseeingsoftware based on those perceptions. Planning doesn't involve any major tools, but maintaining the code entails the use of a range of tools.
Developing the source code for application begins by choosing from the different programming languages.The process of maintaining the code is called Source Code Management (SCM), where version control tools such as Git, TFS, GitLab such others, are used.
With the help of a version control tool, a stable version of the application code is built in the continuous development phase. Developers can also package the code into executable files by using tools.

B. Continuous Integration

The developer modifies source code several times, and these changes happen frequently on a weekly or a daily basis. Code integration phase, is the core of the entire DevOps lifecycle. In continuous integration, new codes that support add-on functionalities are built and integrated into the existing code.
In this phase, bugs are detected early in the source code. To generate new code that brings more functionality to the application, developers run tools for unit testing, code review, integration testing, compilation, and packaging.
The continuous integration of this new code into the existing source code helps reflect the changes that end-users would experience with the updated code.
Jenkins is popularly used as a reliable DevOps tool for procuring the updated source code and constructing the build into an executable format. These transitions occur seamlessly, and the updated code is packaged and continued to the next phase, which is either the production server or the testing server.

C. Continuous Testing

Sometimes developerspractice to carry out the continuous testing phase prior to the continuous integration phase. This phase can be repositioned around the continuous integration phase basedin the DevOps lifecycle on the updates in the application code.
Now, the developed software is continuously tested to identify bugs. A test environment is simulated with the use of Docker containers. Use of automated testing saves Developers effort and time. The reports generated by automated testing improve the test evaluation process. It becomes easy to analyse the failed test cases. After completing UAT (User Acceptance Testing) process, the resulting test suite is bug-free. The tools can be used to arrange test-case execution in a pre-set timeline.
In the end, the tested code is re-sent to the continuous integration phase for updating the source code to ensure the flawless functionality.

D. Continuous Monitoring

Monitoring the performance of an application is of key importance for application developers. In this phase, developers record data by using application and continuously monitor each function.
Continuous monitoring endures the availability of services in the application. It also determines the threats and root causes of recurring system errors. Security issues get resolved and problems are automatically detected and fixed.

E. Continuous Feedback

Continuous testing and continuous integration are the two crucial phases that ensure consistent improvements in the application code. These improvements are analysed in Continuous feedback phase.
Developers can measure the outcome of these modifications on the final product. Most importantly, customers who tested these applications can share their experiences in this phase. The feedback is assessed promptly and developers begin working on the new changes. Soon, there is a positive response in customer feedback, which paves the way for releasing new versions of the software application.

F. Continuous Deployment

Conventionally, the phase of continuous deployment takes place before continuous monitoring.
In this phase, the finalized application code is deployed to the production servers. Configuration Management is a key process in this phase, and it carries out the precise deployment of application code on all servers. Consistency in the application's performance and functional conditions is established and curated. Code is released to the servers, updates are scheduled for all servers, and these configurations are kept consistent throughout the production process.
Containerization tools are used to achieve continuous deployment through the Configuration Management process. These tools nullify all sorts of production failures and system errors by replicating and packaging the software couplings from testing, staging, and development phases. Ultimately, the application runs smoothly on different computers.

G. Continuous Operations

The purpose of continuous operation is to automate the process of releasing the application and the subsequent updates. Development cycles in continuous operations are shorter, allowing developers to accelerate the time-to-market for the application.
This study gives focus on the continuous integration challenges, practices to overcome the challenges and relate how these practices improves quality of the software.

H. Continuous Integration Challenges

Developers and DevOps professionals while adopting DevOps continuous practices came with various challenges. Researcher reported 26 challenges in their research. Many of these challenges have evidences of validation in real life through case studies.
Continuous integration (CI) is the process where software is built and initial tests are completed. The technical goal of CI is to establish a consistent and automated way to build, package, and test applications. With consistency in the integration process in place, teams are more likely to commit code changes more frequently, which lead to better collaboration and software quality.

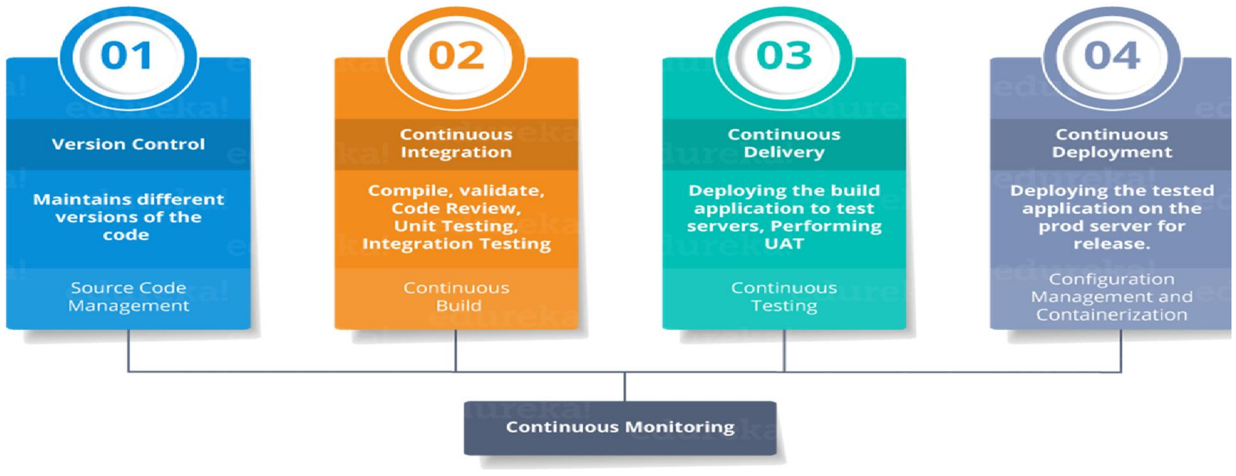
01
Version Control
Maintains different versions of the code
Source Code Management
02
Continuous Integration
Compile, validate, Code Review, Unit Testing, Integration Testing
Continuous Build
03
Continuous Delivery
Deploying the build application to test servers, Performing UAT
Continuous Testing
04
Continuous Deployment
Deploying the tested application on the prod server for release.
Configuration Management and Containerization
Continuous Monitoring

Fig 2 : describes the activities covered in the continuous integration.

I. Continuous Integration Activities

Commit (Version Control)

A code commit stage is otherwise known as version control. A **commit** is an operation that sends the latest changes written by a developer to the repository. Every version of the code written by a developer is stored indefinitely. After a discussion and review of the changes with collaborators, developers will write the code and commit once the software requirements, feature enhancements, bug fixes, or change requests are completed. The repository where the edits & commit changes are managed is called Source Code Management (SCM tool). After the developer commits the code (code Push Request), the code changes are merged into the base code branch stored at the central repository like GitHub.

Build

The Continuous Integration process's goal is to take the regular code commits and continuously build binary artefacts. The continuous integration process helps to find bugs more quickly by checking if the new module that is added plays well with the existing modules. This helps reduce the time to verify a new code change. The build tools help in compiling and creating executable files or packages (.exe,.dll, .jar, etc.) depending on the programming language used to write the source code. During the build, the SQL scripts are also generated and then tested along with infrastructure configuration files. In a nutshell, the build stage is where your applications are compiled. Other sub-activities that are a part of the Build process are Artefactory Storage, Build Verification, and Unit Tests.
The build and commit steps faces some challenges in this SLR researcher reported following 3 challenges

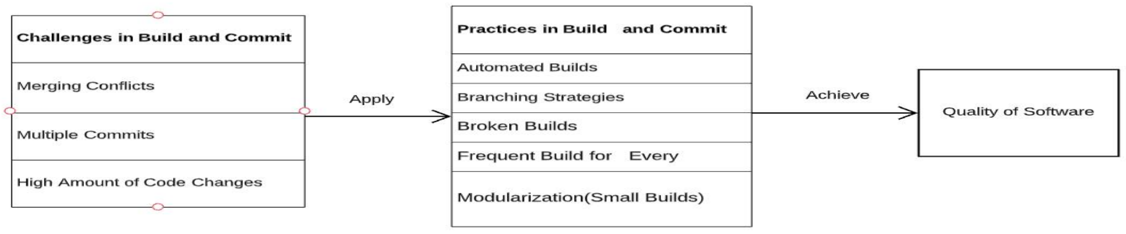

Fig 3: Challeges and Practics in Build and Commit

Test

Post a build process a series of automated tests validates the code veracity. This stage helps errors from reaching the production. Depending on the size of the build this check can last from seconds to hours. For large organizations where codes are committed and built from multiple teams, these checks are run in a parallel environment to save precious time and notify developers of bugs as early as possible.

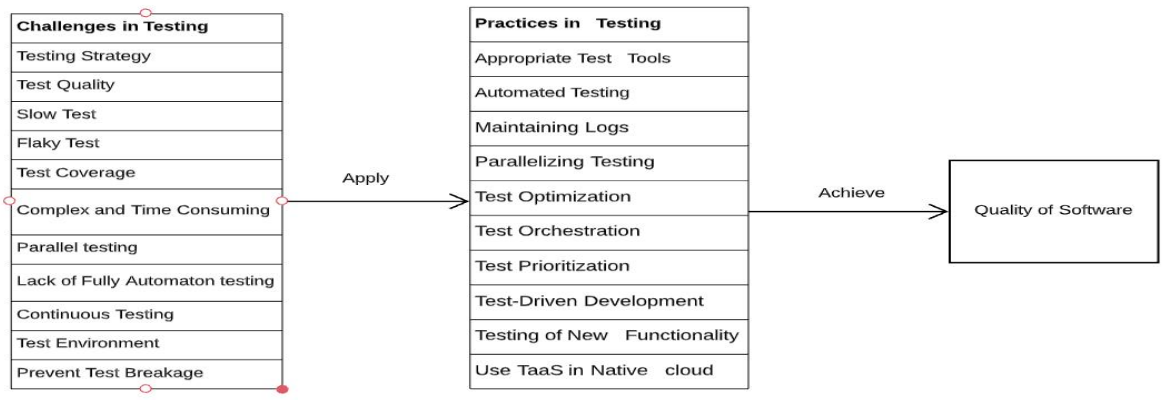

Fig 4: Challeges and Practics in Testing

These automated tests are set up by testers (or known as QA engineers) who have set up test cases and scenarios based on user stories. They perform regression analysis, stress tests to check deviations from the expected output. Activities

that are involved in testing are Sanity tests, Integration tests, Stress tests. This is a much-advanced level of testing that happens. Here we find issues that were probably unknown to the developer developing the code.
Testing is the important step towards quality software, in this SLR researcher could notice following 11 testing challenges.

Role of Tools in Automation

DevOps is implemented through a combination of people, process and tooling. There is need to understand effectiveness of various tools used at each phase of DevOps.
DevOps automation is the addition of technology that performs tasks with reduced human assistance to processes that facilitate feedback loops between operations and development teams so that iterative updates can be deployed faster to applications in production.
These tools are used to design, build, deploy, test, monitor, manage and operate software and systems connected as one integrated pipeline. Tools are broadly classified as Commercial and Open Source tools.
In Continuous Development, process of maintaining the code is called Source Code Management (SCM), where version control tools such as Git, TFS, GitLab such others, are used(Amol, 2020). In the SCM process, Git is a preferred tool (Gokarna, 2020) that enables a distributed version control. The large projects, where a vast number of collaborators are involved in the development activity, Git establishes reliable communication between the teams through the Commit messages.
Continuous Integration, process should be automated and for the automation of continuous integration GitHub is preferred by the practitioners (Wikström, 2019 and Hilton et al., 2016). Jenkins is an open source Continuous Integration and automation server which works on plugins-based architecture and has the capability to integrate variety of tools enabling Continuous Integration(Gokarna, 2020)Jenkins is popularly used as a reliable DevOps tool for procuring the updated source code and constructing the build into an executable format.(Amol, 2020)
Continues automated testing uses TestNG, Selenium, and JUnit are some of the DevOps tools (Amol, 2020 and Gokarna, 2020). Quality assurance engineers (QAs) can use these tools for parallel testing of several other code-bases.
Sensu, ELK Stack, NewRelic, Splunk, and Nagios are the key DevOps tools used in continuous monitoring (Amol, 2020 and Gokarna, 2020). These tools enable complete control the performance management of the system, the production server, and the application. The operations team can actively engage in increasing the reliability and productivity of the applications with the help of these tools.
Vagrant, a containerization tool, develops consistency from development and testing to staging and production(Amol, 2020). The scalability of continuous deployment is handled by tools like Docker (Amol, 2020 and Gokarna, 2020). Ansible, Puppet, and Chef are some of the effective DevOps tools used for Configuration Management, where they frequently execute the quick and continuous deployment of new code(Amol, 2020 and Gokarna, 2020).
Using Automation Tools professionals faces challenges, through this SLR researcher could present 3 challenges as:

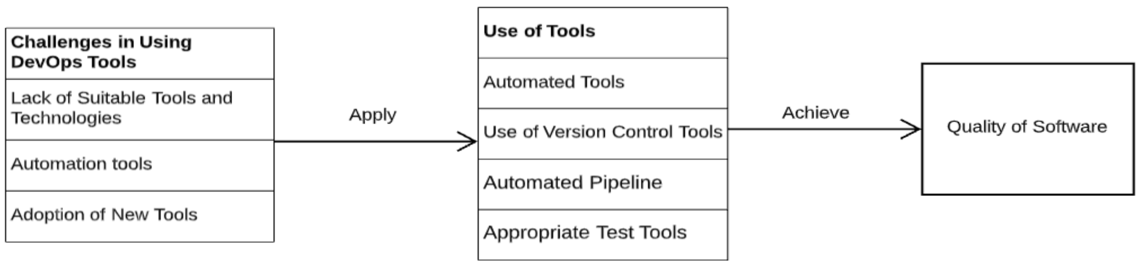

Fig 5: Challeges and Practics in Using DevOps Tools

CI/CD Pipeline

A series of steps that include all the stages from the outset of the CI/CD process and is responsible for creating automated and seamless software delivery is called a CI/CD pipeline workflow. With a CI/CD pipeline, a software release artefact can move and progress through the pipeline right from the code check-in stage through the test, build, deploy, and production stages. This concept is powerful because once a pipeline has been specified, parts or all of it can be automated, speeding the process and reducing errors.

When application is going through CI/CD pipeline some of these issues are reported by researchers are:

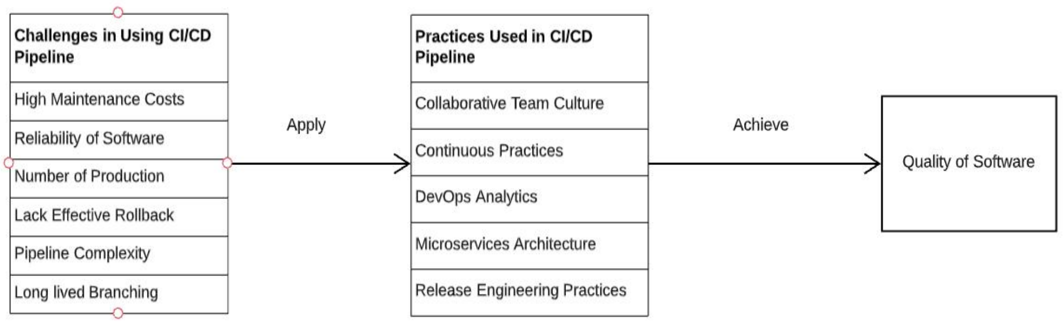

Fig6: Challeges and Practics in Using CI/CD Pipeline

Continuous Monitoring

Continuous Monitoring automates and optimizes the ability to monitor and manage the performance and availability of applications and infrastructure continuously. It tells how good my systems are performing and whether it needs any correction.

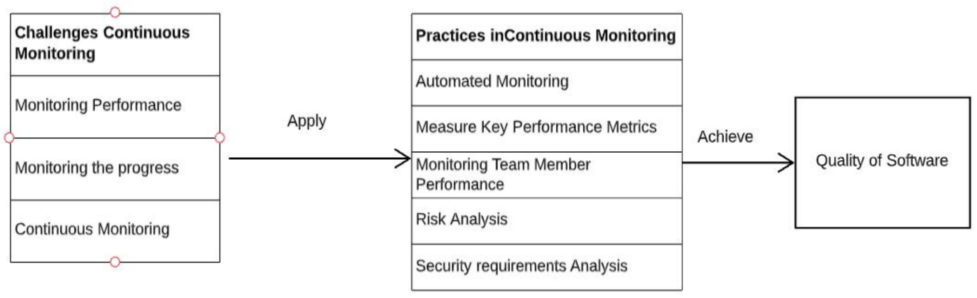
Challenges Continuous Monitoring
Monitoring Performance
Monitoring the progress
Continuous Monitoring
Apply
Practices inContinuous Monitoring
Automated Monitoring
Measure Key Performance Metrics
Monitoring Team Member Performance
Risk Analysis
Security requirements Analysis
Achieve
Quality of Software

Fig 7: Challeges and Practics in Continuous Monitoring

CI Practices to improve Quality

Research articles considered for SLR are 37% validation and 16% survey. Research articles have evidences of validation for 28 practices presented in this article and these practices are effective for improving quality of software. Software quality in DevOps can be achieve by numerous factors listed in the Table no 6

Table 6 : List of Quality Factors in DevOps

Quality Factors Achieved
Fast Release
Reduced Risk
Improved Release frequency
Technical and Cultural Transformations.
Monitoring Perpormance
Improved security
Faster Feedback
Customer Satisfaction
Increased Release Velocity
Faster Defect Detection
Optimizes Mean Time to Recover (MTTR)

V. CONCLUSION

DevOps phenomenon is gaining popularity through its ability to support continuous value delivery and ready accommodation of change. DevOps builds quality into the entire software delivery chain by laying emphasis on communication, collaboration, and integration among various stakeholders in the software development process.CAMS model states that Culture, Automation, Measurement and Sharing are depicted as four pillars of DevOps. DevOps

professionals face some challenges and incorporate practical solutions to these challenges. The aim of this study is to systematically review and analyse challenges confronted and practices adopted in continuous integration practice of software development to improve quality of software in DevOps.
We have done systematic literature review of 44 papers. We conclude 26 challenges and 28 practices which are majorly in the continuous practices, automation, tools, monitoring and pipeline. These practices are having evidences of improving quality through faster release, monitoring performance, reduced risk, reduced testing time and efforts, improved security, fast feedback loop.
In further research we are planning to verify with quantitative and qualitative research techniques and measure the quality factors on selected case studies projects to quantify impact of presented practices.

REFERENCES

[1]. Kent, B. (2001).Manifesto for Agile Software Development. Retrieved From http://agilemanifesto.org [Last accessed: Dec 2018].
[2]. Misra, S. C., Kumar, V., & Kumar, U. (2009). Identifying some important success factors in adopting agile software development practices. Journal of Systems and Software, 82(11), 1869-1890.
[3]. Kurapati, N., Manyam, V. S. C., & Petersen, K. (2012, May). Agile software development practice adoption survey. In International Conference on Agile Software Development (pp. 16-30). Springer, Berlin, Heidelberg.
[4]. Elliot, S. (2014). DevOps and the cost of downtime: Fortune 1000 best practice metrics quantified. International Data Corporation (IDC).
[5]. Perera, P., Bandara, M., & Perera, I. (2016, September). Evaluating the impact of DevOps practice in Sri Lankan software development organizations. In 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 281-287). IEEE.
[6]. Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2016, November). Relationship of DevOps to agile, lean and continuous deployment. In International Conference on Product-Focused Software Process Improvement (pp. 399-415). Springer, Cham.
[7]. Poornalinga, K. S., & Rajkumar, P. (2016). Survey on Continuous Integration, Deployment and Delivery in Agile and DevOps Practices. International Journal of Computer Sciences and Engineering, 4(4), 213-216.
[8]. Perera, P., Silva, R., & Perera, I. (2017, September). Improve software quality through practicing DevOps. In 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 1-6). IEEE
[9]. Haraty, R. A., & Hu, G. (2018). Software process models. International Journal of Engineering & Technology, 7 (2.28) (2018) 390-397
[10]. Darrell. (2018).Agile at Scale. Harvard Business Review (pp. 88–96).
[11]. Nicole, F. Jez, H. & Gene, K. (2018). Accelerate: State of DevOps 2018 Strategies for a New Economy. Retrieved from https://cloudplatformonline.com/rs/248-TPC-286/images/DORA-State%20of%20DevOps.pdf [Last accessed: Nov 2019]
[12]. Business 4.0. (2019).Winning in business 4.0 world TCS. Retrieved from https://www.business4.tcs.com/content/dam/tcs_b4/pdf/winning-in-a-business-4-0-world.pdf [Last accessed: Jan 2020]
[13]. Randolph W. (2019).Transforming Mind sets to accelerate an Agile Transformation. XP2019 conference experience report. Retrieved from https://www.agilealliance.org/resources/experience-reports/transforming-mindsets-to-accelerate-an-agile-transformation/ [Last accessed: Dec 2019].
[14]. Danave, S.(2019).2019 Tech Predictions. Retrieved from https://dzone.com/articles/msys-technologies-2019-tech-predictions-smart-stor [Last accessed: Dec 2019].
[15]. Christopher L,Sean M. (2019). Amplify Agile With DevOps. Retireved From https://dzone.com/articles/amplify-agile-with-devops [Last Accessed: Jan 2020].
[16]. Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering.

[17]. Kitchenham, B. A., Dyba, T., & Jorgensen, M. (2004). Evidence-based software engineering. In Proceedings. 26th International Conference on Software Engineering (pp. 273-281). IEEE.

[18]. Leppänen, M., Mäkinen, S., Pagels, M., Eloranta, V. P., Itkonen, J., Mäntylä, M. V., & Männistö, T. (2015). The highways and country roads to continuous deployment. Ieee software, 32(2), 64-72.

[19]. Chen, L. (2015). Continuous delivery: Huge benefits, but challenges too. IEEE Software, 32(2), 50-54.

[20]. Roche, J. (2013). Adopting DevOps practices in quality assurance. Communications of the ACM, 56(11), 38-43.

[21]. Ståhl, D., & Bosch, J. (2014). Modeling continuous integration practice differences in industry software development. Journal of Systems and Software, 87, 48-59.

[22]. Fitzgerald, B., & Stol, K. J. (2014). Continuous software engineering and beyond: trends and challenges. In Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering (pp. 1-9). ACM.

[23]. Eck, A., Uebernickel, F., & Brenner, W. (2014). Fit for continuous integration: How organizations assimilate an agile practice.

[24]. Fitzgerald, B., & Stol, K. J. (2015). Continuous software engineering: A roadmap and agenda. Journal of Systems and Software, 123, 176-189.

[25]. Rathod, N., & Surve, A. (2015, January). Test orchestration a framework for continuous integration and continuous deployment. In 2015 international conference on pervasive computing (ICPC) (pp. 1-5). IEEE.

[26]. Gottesheim, W. (2015, February). Challenges, benefits and best practices of performance focused DevOps. In Proceedings of the 4th International Workshop on Large-Scale Testing (pp. 3-3).

[27]. Lai, S. T., & Leu, F. Y. (2016, July). A Version Control-based Continuous Testing Frame for Improving the IID Process Efficiency and Quality. In 2016 10th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) (pp. 464-469). IEEE.

[28]. Poornalinga, K. S., & Rajkumar, P. (2016). Survey on Continuous Integration, Deployment and Delivery in Agile and DevOps Practices.International Journal of Computer Sciences and Engineering Vol.-4(4), PP(213-216) April 2016.

[29]. Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016, May). What is devops?: A systematic mapping study on definitions and practices. In Proceedings of the Scientific Workshop Proceedings of XP2016 (p. 12). ACM.

[30]. Perera, P., Bandara, M., & Perera, I. (2016, September). Evaluating the impact of DevOps practice in Sri Lankan software development organizations. In 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 281-287). IEEE.

[31]. Hilton, M., Tunnell, T., Huang, K., Marinov, D., & Dig, D. (2016, September). Usage, costs, and benefits of continuous integration in open-source projects. In 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 426-437). IEEE.

[32]. Zhu, L., Bass, L., & Champlin-Scharff, G. (2016). DevOps and its practices. IEEE Software, 33(3), 32-34.

[33]. Kumar, D., & Mishra, K. K. (2016). The impacts of test automation on software's cost, quality and time to market. Procedia Computer Science, 79, 8-15.

[34]. Rahman, A. A. U., & Williams, L. (2016, May). Software security in devops: Synthesizing practitioners' perceptions and practices. In 2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED) (pp. 70-76). IEEE.

[35]. Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE Access, 5, 3909-3943.

[36]. Ståhl, D., Hallén, K., & Bosch, J. (2017). Achieving traceability in large scale continuous integration and delivery deployment, usage and validation of the eiffel framework. Spinger, Empirical Software Engineering, 22(3), 967-995.

[37]. Elberzhager, F., Arif, T., Naab, M., Süß, I., & Koban, S. (2017, January). From agile development to devops: going towards faster releases at high quality–experiences from an industrial context. In International Conference on Software Quality (pp. 33-44). Springer, Cham.

[38]. Perera, P., Silva, R., & Perera, I. (2017, September). Improve software quality through practicing DevOps. In 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 1-6). IEEE.

[39]. Bou Ghantous, G., & Gill, A. (2017). DevOps: Concepts, practices, tools, benefits and challenges. PACIS2017.

[40]. Heinrich, R., van Hoorn, A., Knoche, H., Li, F., Lwakatare, L. E., Pahl, C., ... & Wettinger, J. (2017, April). Performance engineering for microservices: research challenges and directions. In Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion (pp. 223-226).

[41]. Gupta, V., Kapur, P. K., & Kumar, D. (2017). Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling. Information and software technology, 92, 75-91.

[42]. Karvonen, T., Behutiye, W., Oivo, M., & Kuvaja, P. (2017). Systematic literature review on the impacts of agile release engineering practices. Information and Software Technology, 86, 87-100.

[43]. Vasanthapriyan, S. (2018). Achieving continuous integration excellence in Agile software development,Proceedings of 8th International Symposium-2018, SEUSL

[44]. Arachchi, S. A. I. B. S., & Perera, I. (2018). Continuous Integration and Continuous Delivery Pipeline Automation for Agile Software Project Management. In 2018 Moratuwa Engineering Research Conference (MERCon) (pp. 156-161). IEEE.

[45]. Senapathi, M., Buchan, J., & Osman, H. (2018, June). DevOps capabilities, practices, and challenges: Insights from a case study. In Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018 (pp. 57-67). ACM.

[46]. Laukkanen, E., Paasivaara, M., Itkonen, J., & Lassenius, C. (2018). Comparison of release engineering practices in a large mature company and a startup. Empirical Software Engineering, 23(6), 3535-3577

[47]. Poth, A., Werner, M., & Lei, X. (2018, September). How to Deliver Faster with CI/CD Integrated Testing Services?. In European Conference on Software Process Improvement (pp. 401-409). Springer, Cham.

[48]. Herbst, N., Bauer, A., Kounev, S., Oikonomou, G., Eyk, E. V., Kousiouris, G., ... & Iosup, A. (Eds.). (2018). Quantifying cloud performance and dependability: Taxonomy, metric design, and emerging challenges. ACM Transactions on Modeling and Performance Evaluation of Computing Systems (ToMPECS), 3(4), 1-36.

[49]. Haghighatkhah, A., Mäntylä, M., Oivo, M., & Kuvaja, P. (2018). Test prioritization in continuous integration environments. Journal of Systems and Software, 146, 80-98.

[50]. Agarwal, A., Gupta, S., & Choudhury, T. (2018, June). Continuous and Integrated Software Development using DevOps. In 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE) (pp. 290-293). IEEE.

[51]. Wikström, A. (2019). Benefits and challenges of Continuous Integration and Delivery-A Case Study. Computer Science, 33, 1.

[52]. Kowzan, M., & Pietrzak, P. (2019). Continuous integration in validation of modern, complex, embedded systems. In Proceedings of the International Conference on Software and System Processes (pp. 160-164). IEEE Press.

[53]. Tegeler, T., Gossen, F., & Steffen, B. (2019, January). A Model-driven Approach to Continuous Practices for Modern Cloud-based Web Applications. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 1-6). IEEE.

[54]. Rütz, Martin. (2019). DEVOPS: A SYSTEMATIC LITERATURE REVIEW.Seminar Paper, IT Management Paper, 2019, Wedel, Germany

[55]. Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2019). A Survey of DevOps Concepts and Challenges. ACM Computing Surveys (CSUR), 52(6), 1-35.

[56]. Luz, W. P., Pinto, G., & Bonifácio, R. (2019). Adopting DevOps in the real world: A theory, a model, and a case study. Journal of Systems and Software, 157, 110384.

[57]. Imtiaz, J., Sherin, S., Khan, M. U., & Iqbal, M. Z. (2019). A systematic literature review of test breakage prevention and repair techniques. Information and Software Technology, 113, 1-19.

[58]. Ibrahim, M. M. A., Syed-Mohamad, S. M., & Husin, M. H. (2019, February). Managing Quality Assurance Challenges of DevOps through Analytics. In Proceedings of the 2019 8th International Conference on Software and Computer Applications (pp. 194-198).

[59]. Y. Wang, M. Pyhäjärvi and M. V. Mäntylä. (2020). "Test Automation Process Improvement in a DevOps Team: Experience Report," 2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Porto, Portugal, pp. 314-321, doi: 10.1109/ICSTW50294.2020.00057.

[60]. Lima, J. A. P., & Vergilio, S. R. (2020). Test Case Prioritization in Continuous Integration environments: A systematic mapping study. Information and Software Technology, 121, 106268.

[61]. Khan, M. O., Jumani, A. K., & Farhan, W. A. (2020). Fast Delivery, Continuously Build, Testing and Deployment with DevOps Pipeline Techniques on Cloud. INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY, 13(05), 552-575.

[62]. Mishra, A., & Otaiwi, Z. (2020). DevOps and software quality: A systematic mapping. Computer Science Review, 38, 100308.

[63]. Gokarna, M. (2020). DevOps phases across Software Development Lifecycle.IBM India Pvt Ltd.

[64]. Amol, M. (2020). What is DevOps Lifecycle and How to Manage Yours. Retrieved From https://dzone.com/articles/what-is-devops-lifecycle-how-to-manage-yours [Last accessed: May 2021].

APPENDIX I

Table 7 : Coding of Selected Paper for SLR

Code	Title
S1	Roche, J. (2013). Adopting DevOps practices in quality assurance. Communications of the ACM, 56(11), 38-43.
S2	Ståhl, D., & Bosch, J. (2014). Modeling continuous integration practice differences in industry software development. Journal of Systems and Software, 87, 48-59.
S3	Fitzgerald, B., & Stol, K. J. (2014). Continuous software engineering and beyond: trends and challenges. In Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering (pp. 1-9). ACM.
S4	Eck, A., Uebernickel, F., & Brenner, W. (2014). Fit for continuous integration: How organizations assimilate an agile practice.
S5	Fitzgerald, B., & Stol, K. J. (2015). Continuous software engineering: A roadmap and agenda. Journal of Systems and Software, 123, 176-189.
S6	Rathod, N., & Surve, A. (2015, January). Test orchestration a framework for continuous integration and continuous deployment. In 2015 international conference on pervasive computing (ICPC) (pp. 1-5). IEEE.
S7	Gottesheim, W. (2015, February). Challenges, benefits and best practices of performance focused DevOps. In Proceedings of the 4th International Workshop on Large-Scale Testing (pp. 3-3).
S8	Lai, S. T., & Leu, F. Y. (2016, July). A Version Control-based Continuous Testing Frame for Improving the IID Process Efficiency and Quality. In 2016 10th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) (pp. 464-469). IEEE.
S9	Poornalinga, K. S., & Rajkumar, P. (2016). Survey on Continuous Integration, Deployment and Delivery in Agile and DevOps Practices.International Journal of Computer Sciences and Engineering Vol.-4(4), PP(213-216) April 2016.
S10	Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016, May). What is devops?: A systematic mapping study on definitions and practices. In Proceedings of the Scientific Workshop Proceedings of XP2016 (p. 12). ACM.
S11	Perera, P., Bandara, M., & Perera, I. (2016, September). Evaluating the impact of DevOps

	practice in Sri Lankan software development organizations. In 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 281-287). IEEE.
S12	Hilton, M., Tunnell, T., Huang, K., Marinov, D., & Dig, D. (2016, September). Usage, costs, and benefits of continuous integration in open-source projects. In 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 426-437). IEEE.
S13	Zhu, L., Bass, L., & Champlin-Scharff, G. (2016). DevOps and its practices. IEEE Software, 33(3), 32-34.
S14	Kumar, D., & Mishra, K. K. (2016). The impacts of test automation on software's cost, quality and time to market. Procedia Computer Science, 79, 8-15.
S15	Rahman, A. A. U., & Williams, L. (2016, May). Software security in devops: Synthesizing practitioners' perceptions and practices. In 2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED) (pp. 70-76). IEEE.
S16	Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE Access, 5, 3909-3943.
S17	Ståhl, D., Hallén, K., & Bosch, J. (2017). Achieving traceability in large scale continuous integration and delivery deployment, usage and validation of the eiffel framework. Spinger, Empirical Software Engineering, 22(3), 967-995.
S18	Elberzhager, F., Arif, T., Naab, M., Süß, I., & Koban, S. (2017, January). From agile development to devops: going towards faster releases at high quality–experiences from an industrial context. In International Conference on Software Quality (pp. 33-44). Springer, Cham.
S19	Perera, P., Silva, R., & Perera, I. (2017, September). Improve software quality through practicing DevOps. In 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 1-6). IEEE.
S20	Bou Ghantous, G., & Gill, A. (2017). DevOps: Concepts, practices, tools, benefits and challenges. PACIS2017.
S21	Heinrich, R., van Hoorn, A., Knoche, H., Li, F., Lwakatare, L. E., Pahl, C., ... & Wettinger, J. (2017, April). Performance engineering for microservices: research challenges and directions. In Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion (pp. 223-226).
S22	Gupta, V., Kapur, P. K., & Kumar, D. (2017). Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling. Information and software technology, 92, 75-91.
S23	Karvonen, T., Behutiye, W., Oivo, M., & Kuvaja, P. (2017). Systematic literature review on the impacts of agile release engineering practices. Information and Software Technology, 86, 87-100.
S24	Vasanthapriyan, S. (2018). Achieving continuous integration excellence in Agile software development,Proceedings of 8th International Symposium-2018, SEUSL
S25	Arachchi, S. A. I. B. S., & Perera, I. (2018). Continuous Integration and Continuous Delivery Pipeline Automation for Agile Software Project Management. In 2018 Moratuwa Engineering Research Conference (MERCon) (pp. 156-161). IEEE.
S26	Senapathi, M., Buchan, J., & Osman, H. (2018, June). DevOps capabilities, practices, and challenges: Insights from a case study. In Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018 (pp. 57-67). ACM.
S27	Laukkanen, E., Paasivaara, M., Itkonen, J., & Lassenius, C. (2018). Comparison of release engineering practices in a large mature company and a startup. Empirical Software Engineering, 23(6), 3535-3577
S28	Poth, A., Werner, M., & Lei, X. (2018, September). How to Deliver Faster with CI/CD Integrated Testing Services?. In European Conference on Software Process Improvement (pp.

	401-409). Springer, Cham.
S29	Herbst, N., Bauer, A., Kounev, S., Oikonomou, G., Eyk, E. V., Kousiouris, G., ... & Iosup, A. (Eds.). (2018). Quantifying cloud performance and dependability: Taxonomy, metric design, and emerging challenges. ACM Transactions on Modeling and Performance Evaluation of Computing Systems (ToMPECS), 3(4), 1-36.
S30	Haghighatkhah, A., Mäntylä, M., Oivo, M., & Kuvaja, P. (2018). Test prioritization in continuous integration environments. Journal of Systems and Software, 146, 80-98.
S31	Agarwal, A., Gupta, S., & Choudhury, T. (2018, June). Continuous and Integrated Software Development using DevOps. In *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)* (pp. 290-293). IEEE.
S32	Wikström, A. (2019). Benefits and challenges of Continuous Integration and Delivery-A Case Study. Computer Science, 33, 1.
S33	Kowzan, M., & Pietrzak, P. (2019). Continuous integration in validation of modern, complex, embedded systems. In Proceedings of the International Conference on Software and System Processes (pp. 160-164). IEEE Press.
S34	Tegeler, T., Gossen, F., & Steffen, B. (2019, January). A Model-driven Approach to Continuous Practices for Modern Cloud-based Web Applications. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 1-6). IEEE.
S35	Rütz, Martin. (2019). DEVOPS: A SYSTEMATIC LITERATURE REVIEW.Seminar Paper, IT Management Paper, 2019, Wedel, Germany
S36	Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2019). A Survey of DevOps Concepts and Challenges. ACM Computing Surveys (CSUR), 52(6), 1-35.
S37	Luz, W. P., Pinto, G., & Bonifácio, R. (2019). Adopting DevOps in the real world: A theory, a model, and a case study. Journal of Systems and Software, 157, 110384.
S38	Imtiaz, J., Sherin, S., Khan, M. U., & Iqbal, M. Z. (2019). A systematic literature review of test breakage prevention and repair techniques. Information and Software Technology, 113, 1-19.
S39	Ibrahim, M. M. A., Syed-Mohamad, S. M., & Husin, M. H. (2019, February). Managing Quality Assurance Challenges of DevOps through Analytics. In Proceedings of the 2019 8th International Conference on Software and Computer Applications (pp. 194-198).
S40	Y. Wang, M. Pyhäjärvi and M. V. Mäntylä, "Test Automation Process Improvement in a DevOps Team: Experience Report," 2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Porto, Portugal, 2020, pp. 314-321, doi: 10.1109/ICSTW50294.2020.00057.
S41	Lima, J. A. P., & Vergilio, S. R. (2020). Test Case Prioritization in Continuous Integration environments: A systematic mapping study. Information and Software Technology, 121, 106268.
S42	Khan, M. O., Jumani, A. K., & Farhan, W. A. (2020). Fast Delivery, Continuously Build, Testing and Deployment with DevOps Pipeline Techniques on Cloud. INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY, 13(05), 552-575.
S43	Mishra, A., & Otaiwi, Z. (2020). DevOps and software quality: A systematic mapping. Computer Science Review, 38, 100308.
S44	Gokarna, M. (2020). DevOps phases across Software Development Lifecycle.IBM India Pvt Ltd.

Table 8: List of challenges found through this systematic Literature Review

Challenges	Paper Addressed	#
Testing		
Testing Strategy	[S16],[S21]	2
Test Quality	[S16],[S41]	2
Slow Test	[S32]	1

Flaky Test	[S32] ,[S41]	2
Test Coverage	[S32], [S39]	2
Complex and Time Consuming	[S41]	1
Parallel testing	[S41]	1
Lack of Fully Automaton testing	[S19], [S31],[S39],[S42]	4
Continuous Testing	[S5]	1
Test Environment	[S33], [S34]	2
Prevent Test Breakage	[S38]	1
Monitoring		
Monitoring Performance	[S13],[S21],[S29]	3
Monitoring the progress	[S17]	1
Continuous Monitoring	[S5]	1
Tools		
Lack of Suitable Tools and Technologies	[S16], [S27],[S33],[S42]	4
Automation tools	[S19]	1
Adoption of New Tools	[S26]	1
Build		
Merging Conflicts	[S16]	1
High Amount of Code Changes	[S33], [S34]	2
Pipeline		
High Maintenance Costs	[S12]	1
Reliability of Software	[S13]	1
Number of Production Environments	[S27]	1
Lack Effective Rollback Mechanism	[S31],[S39]	2
Pipeline Complexity	[S12],[S32]	2
Multiple Commits	[S41]	1
Long lived Branching	[S31]	1

Table 9: List of Practices and count of research article addressed in SLR

Practices	Paper Addressed	#
Appropriate Test Tools	[S7],[S10],[S13],[S15],[S17],[S38],[S40], [S42]	8
Automated Monitoring	[S20],[S26],[S37],[S43]	4
Automated Builds	[S9],[S19],[S26],[S43]	4
Automated Pipeline	[S34],[S39],[S42],[S43]	4
Automated Testing	[S4],[S9],[S10],[S14],[S15],[S18],[S19],[S20], [S22],[S23],[S26],[S36],[S37],[S40],[S42],[S43]	16
Automated Tools	[S9],[S12],[S19],[S20],[S22],[S25],[S26],[S32], [S35], [S37],[S39],[S42],[S43]	13
Branching Strategies	[S4],[S16],[S22]	3
Broken Builds	[S2],[S32]	2
Collaborative Team Culture	[S35],[S37]	2
Continuous Practices	[S3],[S7],[S8],[S10],[S19],[S22],[S26]	7
DevOps Analytics	[S39]	1
Frequent Build for Every Chang	[S24],[S33]	2
Maintaining Logs	[S24]	1
Measure Key Performance Metrics	[S2],[S7],[S18],[S19],[S36],[S37],[S38], [S40],[S43]	9

Micro-services Architecture	[S13],[S36],[S43]	3
Modularization(Small Builds)	[S2],[S16]	2
Monitoring Team Member Performance	[S11]	1
Parallelizing Testing	[S32],[S36]	2
Release Engineering Practices	[S27]	1
Risk Analysis	[S15]	1
Security requirements Analysis	[S15]	1
Test Optimization	[S4]	1
Test Orchestration	[S6]	1
Test Prioritization	[S1],[S30],[S32],[S41]	4
Test-Driven Development	[S4],[S23]	2
Testing of New Functionality	[S2]	1
Use of Version Control Tools	[S24]	1
Use TaaS in Native cloud	[S28]	1