# **REPORT ON CONTINUOUS INTEGRATION ENVIORNMENT**

ANUSHA MOGILI -9706268167

SRAVANI BEJAWADA-9705259563

RAHUL DANDULURI -9505281353

KARTHIK THUPALLE -9409160133

**ABSTRACT**

Software industry is known for team work and group management, to achieve this one needs to do a lot of hard work. It may be the project manager, engineer, stakeholder or anyone, but everyone needs to do the same amount of hard work. While doing a team work, there are many operations to be performed even for a small task like updating, deleting...etc. Doing this individually and sharing with everyone needs a lot of work and it is a time-consuming process. For this type of problems in software engineering there's a solution that needs to be implemented. And that is known as continuous integration which is an automated process for compile and test when a commit is pushed by a team member and it is pulled by every member of the team to update the work done. This can be achieved by using continuous integration tool and version control server and software.

**INTRODUCTION**

In today's world, automated process towards agile methodology is the most researched topic and it is improving day by day in software industry. To improve agile methodology the automation process plays an important role by helping in time, space management and allows work to complete faster.

While coding for a project there's a lot of work to be done by team members and to keep updated frequently is difficult task but it is a very useful task and helpful in finding errors at early stage. Most of the organizations that are running presently, uses the continuous integration [1]tools to achieve the automation process.

To achieve automation process, a continuous integration (CI) environment that can execute tests should be built. This can be done from combining different types of software's for coding and testing and showing the results.

The continuous integration can be done by installing software that can work in fast build environment. some of the CI tools are Jenkins, GitLab, team city, Travis CI. In CI the work that is done is: after installing the software the configuration is done and the project is uploaded from the GitHub and it is executed to get the results. It shows how many projects are committed and pull is made, and test results will be shown in console output. Unit test performed using Jenkins as it gives the better results with every detail possible to give in output.

A version control software is intermediate software between CI and GitHub. Where every user can organize and perform operations and console the project work individual by updating the data by pull operation.

A version control server is a software tool that help a team to maintain changes and manage up to date. In version control software the new code is uploaded and managed in version control software and executed in continuous integration.

An automated build tool that connects to the continuous integration tool and builds the project to perform testing. For example, maven, nowadays IDE like eclipse are having inbuilt plugins where we must download them directly and install.

An integrated development environment like eclipse, which plays the most important part that is where a source code is implemented, and test are performed with the help of Junit for unit testing or other testing tools like XUnit, eye automate can be used to produce unit and integrated test.

Using these a complete continuous integration (CI) is built to perform simple to complex jobs easily.

**DESCRIPTION**

**Continuous Integration (CI)** [2]

Continuous integration (CI) environment is built using all the above tools and is made perfectly correct for project use by the developers. Mainly a continuous integration means a team of developers make changes to a repository and integrate them several times a day and all the team members pulls the push and commit changes done to the source codes. All the integrated work is then automated and tested depending on the project requirements.

By using the continuous integration errors can be identified quickly and correct them accurately before it's too late to change the whole code in worst case. This can be achieved in continuous integration because, small changes are made in repository more frequently in CI and every change can be identified, executed and corrected in case of bugs. It is better than correcting the whole source code at once.

In software development the continuous integration played an important role and it is evolved day by day and contains three main aspects of integration those are:

revision control is same as version control, build automation is used to do the work automatically like maven and automated testing where integrated code is tested by using testing tools.

continuous integration has its own key principals and its own best-practices to produce along with Continuous Deployment and Continuous Delivery, which helps in your application deployment and keeping updating the changes and delivery the execution results.

## **TECHNICAL CHOICES OF TOOLS:**

**Eclipse**: which is known as a Software Development Kit (SDK) for integrated development environment (IDE) for java projects with developed in eclipse platform. Where a million of products are developed with suitable plugins from frameworks to testing desires.[3]

**Technical overview for eclipse platform to support is:**

-Variety of software tools are integrated to perform the desired projects and to build the product.

-Arbitrary tools can be easily managed and manipulated like html, java, GIF.

-GUI and non-GUI applications are supported to develop.

-Eclipse is supported on any operating system in different sizes like windows, Linux.

-Gives importance to the java language in developing products.

Nowadays eclipse is most used IDE and the facilities provided in eclipse can be efforted in another platform like NetBeans for examples apache can be integrated with the eclipse and can be executed the output is shown in eclipse console with the help of server and inbuilt browser accessibility.

**Gitkraken:** is a version control software where it is easy to operate the changes in the repository and keep track of it.[4]

**Technical overview for** **Gitkraken are:**

-Atomic commits where it guarantees the changes that are performed.

-File renames can be performed without fail and can merge the rename files.

-Symbolic links for revision control and for security purpose

-Pre/post event hooks for trigger commands such as action, commit, act.

-Tags and Unicode filename are supported for specific regions of the file and encodings for different characters.

These are the technical aspects that are useful in version control software.

**GitHub:** is a version control server for hosting a web server. Which is used to integrate the version control software and continuous integration tools. Where the action, commit, push operations are performed to the repository.[5]

**Technical overviews of GitHub are:**

-README files and documentation can be obtained without fail after commit and push a project.

-Tracking of issues and requests pull for push is organized.

-History of commits, graphs, directory od integrations, notification in emails are generated.

-Publicly operated websites can be used and perform operations in GitHub pages.

-Pdf viewer and subscribe to the specific project team work can integrated.

**Jenkins:** Continuous Integration (CI) tool for automated built and testing.[5]

**Technical features for Jenkins:**

-Jenkins can Jenkins work not only as continuous integration tool but also continuous deployment and delivery depending on product requirement.

-Easy download and installation for any type of OS.

-Improves performance for builds and tests with deploy feature

-Different types of plugins are available for different type of requirements like user interface, platforms.

-Server is configured easily for web interfaces.

**Maven:** automated build tool for connecting to CI tool is mostly used for java projects and it has two purposes to accomplish like product built and describe the dependencies between them.[6]

**Technical aspects of Maven:**

-New developers has consistency on all projects and time consumption is avoided.

-Can operate more no of projects simultaneously.

-Adding or extending new plugins depending on requirements.

-Setup for new projects are easy and can be completed in seconds of time.

-Many number of repository metadata can be easily managed.

**CHALLENGES THAT ARE FACED WHILE IMPLEMENTING THEM**

-In eclipse integrating the different plugins and importing them is the serious problem that will be faced while building the project.

-Version control software like GitKraken, in this the stored files are unorganized and needs technical understanding and search of the directory files to be installed files.

-Version control server like GitHub, the operations that are performed to the code repository are bit difficult and sharing the link and access the team members work needs a lot of work.

-Continuous integration like Jenkins, while installation server problems are faced by many and the exact problem in configuration is not explained correctly, the time of execution log takes time to display.

-Maven takes time to integrate plugins and other resources while implementing java programs and doesn’t differentiate at a time which makes it difficult to understand.

-Unit and integration test can obtain number if test cases in false and true assumptions without main functionalities which difficult to identify.

Solutions to the challenges that are faced:

-Importing and integration should be done in a specific ways step by step process mainly while importing from one work space to another one. It can be obtained from the lot of practise.

-Gitkraken has saved files folder where each file can be opened and viewed. Can also organize data with a little understanding of the technical aspects

-Accessing the repository by every member of the team with invitation each is not necessary which is helpful, and operations are confusing that can be removed by using version control software.

-CI problems like apache server configuration can be done in alternative ways to achieve and the time taken for log display is inbuilt and it takes time for updating the test results and in the mean time we can execute our next process.

-Maven makes every project different setup and path but while using extra plugins the path gets confused and fused, it can be removed from configuring build path and cleared.

-Unit and integration tests are performed to test the functionalities by assuming different test cases for one function and testing which case is true and which is false. This should be done in a meaningful way and test the code.

**RESULT**

Using CI environment, the automated builds and tests can execute frequently which helps in team work with less effort and reducing time consumption, also plays an important role in finding errors quickly and managing the changes up to the date.

**CONCLUSION**

CI plays an important role in today's software industry and it helps to improve the performance like speed, time and space. which is the most important in agile methodology by using automated process for built and tests like unit and integration tests where the functionality of the cod is tested.

**REFERENCES**

[1] S. Guckenheimer, ‘What is Continuous Integration? | DevOps’, *Visual Studio*. [Online]. Available: https://www.visualstudio.com/learn/what-is-continuous-integration/. [Accessed: 16-Mar-2018].

[2] C. Inc, ‘Continuous Integration: What is CI? Testing, Software & Process Tutorial’, *Continuous Integration Essentials | Codeship*. [Online]. Available: http://cms.codeship.com/continuous-integration-essentials. [Accessed: 16-Mar-2018].

[3] ‘Eclipse Platform Technical Overview’. [Online]. Available: http://www.eclipse.org/articles/Whitepaper-Platform-3.1/eclipse-platform-whitepaper.html. [Accessed: 16-Mar-2018].

[4] ‘GitKraken Current | GitKraken Documentation’. [Online]. Available: https://support.gitkraken.com/release-notes/current. [Accessed: 16-Mar-2018].

[5] ‘Better Integration Between Jenkins and GitHub (with the GitHub Jenkins Plugin)’, *CloudBees*, 11-Jan-2012. [Online]. Available: https://www.cloudbees.com/blog/better-integration-between-jenkins-and-github-github-jenkins-plugin. [Accessed: 16-Mar-2018].

[6] ‘Maven – Maven Features’. [Online]. Available: https://maven.apache.org/maven-features.html. [Accessed: 16-Mar-2018].