Written Assessment: Continuous Integration Tools Research

Continuous Integration Overview

Continuous integration is a style of development when all blocks of code is merging together several times per day (as often as program blocks change).

The main benefit of this approach is that all problems of integration are revealing bugs on the early stage. Let's imagine the team which has to build a really fancy lego house. Each team member is working on his part. The first approach could be like this: each team member has to finish completely his part of work and then and only then all parts of the house are getting together. If on the final stage some parts don't fit other parts of the house it can be hard to find the mistakes. Another approach - continuous integration - when the parts of the house are putting together before they completely finished. In this case, if one room doesn't fit the rest of the house the mistake can be found and fixed immediately. Also, with this approach, each team member sees the house, so the process of developing becomes more clear, which has a positive effect on the productivity of the team.

Of course, any approach takes some resources. For implementing a continuous integration a team needs an instrument which would make the process of integration as fast and as smooth as possible. There are a lot of solutions which could help developers. The instruments depend on parameters of the project - team size, programming language, the complexity of the project etc.

Even though CI would take some resources (new tools, a new approach, reconfiguration of the team, writing tests etc) as shown above CI can seriously increase the productivity of the team and decrease the time for delivering the product to a customer. It is a step to the future of software development and a lot of big corporations already made this step.

Continuous Integration, Continuous Delivery and Continuous Deployment

Continues delivery is the next step which is following continuous integration. Using this approach at any moment the team has a product which can be delivered to a customer. However, there is still a human involved in the process of delivering the final version of the app. Continues deployment means that every change of code (in case if it passed the tests) is automatically become a part of the product which delivers to customer straight away.

Continuous integration is the first step towards the continuous deployment. The continuous delivery is a second step and the continuous deployment is the final one. It is possible to use CI approach without CD, but impossible to have CD without CI.

Continuous Integration Practices

✓ **Commit changes as fast as possible** - this is an essential part of the process. The more often changes are committed the more easy will be bug fixing after merging, the faster the product can be delivered.

✓ **Write reliable unit and integration test suites** - another important thing - test the code as often as possible (after each new build). Tests have to have 100% coverage and one mistake in the test code can cost a lot of resources, so tests should be reliable.

✓ **Easy come back to the previous version of the product** - in case if some of the tests is failed after a new build, developers need an easy way to come back to previous "good" version of the product.

The Role of Testing in Continuous Integration

Testing in CI (and even more in CD) plays one of the most important roles. After merging all codes of blocks tests are allowed to find and fix mistakes in the code as fast as possible. Automatic tests are running after each build, maybe a few times per day, so tests should have 100% of coverage of the code and be reliable.

Continuous Integration Tools Analysis and Comparison

There are a lot of tools for implementing CI approach.

"**GitLab** supports development teams with a well-documented installation and configuration processes, an easy-to-follow UI, and a flexible per-seat pricing model that supports self service. GitLab's vision is to serve enterprise-scale, integrated software development teams that want to spend more time writing code and less time maintaining their tool chain."
- Forrester CI Wave™

GitLab is trying to make CI as easy as possible, so developers would not have to make a lot of adjustments and learning before starting the work with the system.

"**Jenkins** is an open source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat. It supports version control tools, including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, TD/OMS, ClearCase and RTC, and can execute Apache Ant, Apache Maven and sbt based projects as well as arbitrary shell scripts and Windows batch commands." Wikipedia

This is one of the most popular systems. It is based on plugins, which is making this system very flexible.

"**TeamCity** is a Java-based build management and continuous integration server from JetBrains. It was first released on October 2, 2006.[2] TeamCity is commercial software and licensed under a proprietary license. A Freemium license for up to 100 build configurations and 3 free Build Agent licenses is available. Open Source projects can request a free license." Wikipedia

One of the features of this system is a great support for Visual Studio (framework testing, code analysis, coverage of the code - all included)

Specific Continuous Integration Tool Overview

Let's have a closer look on GitLab CI

GitLab Continuous Integration (CI) &
Continuous Delivery (CD)

Main features of the GitLab CI:

- ✓ **Stable:** your builds run on a different machine than GitLab.
- ✓ **Parallel builds:** GitLab CI/CD splits builds over multiple machines, for fast execution.
- ✓ **Realtime logging:** a link in the merge request takes you to the current build log that updates dynamically.
- ✓ **Flexible pipelines:** you can define multiple parallel jobs per stage and you can trigger other builds.

- ✓ GitLab is one application for the **entire DevOps lifecycle**
- ✓ Fully **integrated with GitLab**
- ✓ ...and more

Some of the terminology which are using in GitLabCI (from official website):

Omnibus is a way to package different services and tools required to run GitLab, so that most users can install it without laborious configuration.

GitLab Runner is the open source project that is used to run your jobs and send the results back to GitLab. It is used in conjunction with GitLab CI, the open-source continuous integration service included with GitLab that coordinates the jobs.

GitLab security dashboard - The Security Dashboard is a good place to get an overview of all the security vulnerabilities in your groups and projects. You can also drill down into a vulnerability and get extra information, see which project it comes from, the file it's in, and various metadata to help you analyze the risk. You can also action these vulnerabilities by creating an issue for them, or by dismissing them.

Cycle analytic - measures the time spent to go from an idea to production - also known as cycle time - for each of your projects. Cycle Analytics displays the median time for an idea to reach production, along with the time typically spent in each DevOps stage along the way.
Cycle Analytics is useful in order to quickly determine the velocity of a given project. It points to bottlenecks in the development process, enabling management to uncover, triage, and root-cause slowdowns in the software development life cycle.

Wiki - A separate system for documentation called Wiki, is built right into each GitLab project. It is enabled by default on all new projects and you can find it under Wikiin your project. Wikis are very convenient if you don't want to keep your documentation in your repository, but you do want to keep it in the same project where your code resides.

Learning Activity

Choose the right answer:

1. What stands behind abbreviation CI/CD?
   - ▢ Common Integration and Control Disk
   - ▢ Continuous Integration and Continuous Download
   - ▢ Continuous Integration Continuous Delivery and/or Continuous Deployment

2. Choose one or more tools for implementing CI/CD:
   - ▢ GitLab CI
   - ▢ NetBeans
   - ▢ Jenkins
   - ▢ Pacman
   - ▢ SkyCity

3. What is Omnibus?
   - ▢ A way to package different services and tools required to run GitLab
   - ▢ The funny city bus
   - ▢ One of the main packages in Jenkins
   - ▢ The name of the repository on GitHub

References

- [RUS]ИССЛЕДОВАНИЕ СОВРЕМЕННЫХ CI/CD ИНСТРУМЕНТОВ И ВНЕДРЕНИЕ ИХ В КРУПНЫЕ WEB-ПРОЕКТЫ НА ПРИМЕРЕ JENKINS Retrieved from https://sibac.info/studconf/tech/lxxiv/130991
- [RUS]Александр, &. (2018, April 04). Continuous Integration для новичков. Retrieved from https://habr.com/ru/post/352282/ Немытченко, &. (2018, April 11).
- [RUS] Для чего программисту Continuous Integration и с чего начинать. Retrieved from https://habr.com/ru/post/353194/

- GitLab Continuous Integration & Delivery. Retrieved from https://about.gitlab.com/product/continuous-integration/
- Jenkins. Retrieved from https://jenkins.io/
- Top Continuous Integration Tools: The 50 Best CI & Continuous Delivery Tools. (2019, June 19). Retrieved from https://stackify.com/top-continuous-integration-tools/
- TeamCity: The Hassle-Free CI and CD Server by JetBrains. Retrieved from https://www.jetbrains.com/teamcity/