Advances in Computing 2020, 10(1): 10-14
DOI: 10.5923/j.ac.20201001.02

Continuous Integration on Cloud Versus on Premise: A Review of Integration Tools

Vasanth K. Makam

Phoenix, AZ, United States of America

Abstract Continuous Integration (CI) aids in ensuring the software development process is reliable and provides fast feedback on tests and builds. It helps to detect errors quickly and corrects them before the system gets complicated. The CI process comprises of tools that help in identifying the correctness and provision of feedback for the software under process. The CI tools cleanse the data of errors through constant build and tests. The CI tools assist in transforming and mapping of the data before Integration and quick detection of errors. As cloud computing grows and gets embraced by businesses, the choice to host CI tools on premise or on cloud becomes a critical management decision. To make this decision, businesses ought to get acquainted with various CI tools and models they wish to implement, the security of their data, the capital to invest, and the reliability of the providers. Analyzing these variables will give business management an understanding of 1) the right CI tools for their businesses and 2) the best platform to host the integration tools. This review of CI on the cloud versus on premise revealed that it is imperative when determining the right CI tool for your need, to consider the right platform to host it to ensure synchronization.

Keywords Continuous Integration, DevOps, Continuous Delivery, Cloud Computing, Cloud Security, Agile

1. Introduction

1.1. Continuous Integration Overview

Globally, computing technologies are emergent. The environment of CI tools and where to host is receiving keen attention by businesses as they strive to offer efficient services to their customers. Whether to host CI tools on the cloud or on-premise has become a growing concern due to the emergent of various modern CI tools and cloud computing. Decisions on hosting the CI tools on-premise or cloud are complex in terms of interface and its infrastructure. [1]. They have turned to modern development practices, including continuous integration of the cloud infrastructure.

Continuous Integration commences with software integration. Software integration, as part of the development cycle, is the practice of linking components of software together to attain a single unified system [2]. Continuous Integration is, therefore, a process of software development, a software engineering concept where developers release their design codes more frequently (continuously) to identify and correct issues earlier as they occur [3]. Continuous Integration is one of the extreme programming (XP) practices that help to reduce efforts for code development. It leads to higher-quality software with delivery results that are predictable [2], [3]. The software market competition is increasing [4].

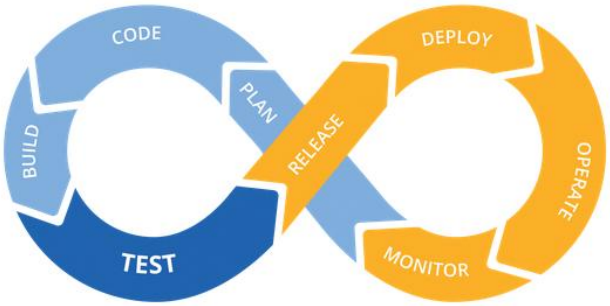

Figure 1. Architecture pattern [3]

1.2. How Continuous Integration Works

Continuous Integration has been in existence for a while in the software engineering industry. It was recognized in 1991 by a software engineer Grady Booch [5], and later adopted by XP with the modification of integrating regularly. In 2015, software configuration was still complex. Vincit, a software development company, took the initiative through their Need for Speed (N4S) project to bring tremendous developments and improvements in the features of software giving birth to continuous integration/delivery. Vincit brought out software build and development tools that are automated to support the complexity of the deployment and configurations to the software [6].

* Corresponding author:
vasanthmakam@gmail.com (Vasanth K. Makam)
Published online at http://journal.sapub.org/ac

Continuous Integration works when developers continually add parts of their system they are working on into a depository. The parts are automatically tested and built. The developer then receives immediate feedback from the network about the codes newly integrated and is capable of correcting them before they get complicated. This practice requires each developer to commit to frequently feed in the codes, to fix any broken code quickly, to write the automated tests, and not commit any broken code into the system [2]. The developers work with the newest codes sent to the system, tested, and built. When that process fails by any chance, then the developers stop working on their tasks until the failure is fixed.

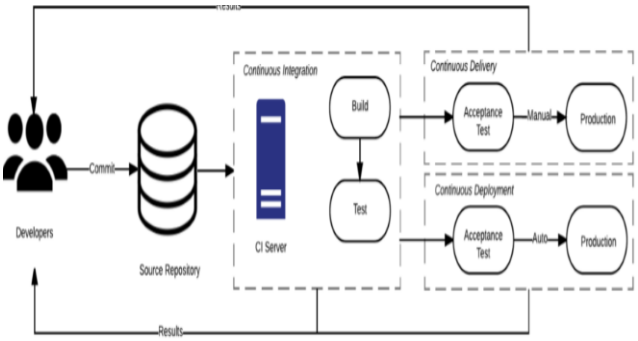

Figure 2. Continuous integration delivery [4]

Developers use various tests over time. 1) unit tests that verify individual behavior of functions, 2) integration tests that ensure multiple components run concurrently and correctly to integrate with other services. 3) Acceptance tests that are similar to integration tests but focus on business cases, 4) UI tests that ensure that the system applications are running correctly. The developers run tests whenever they detect change in the system. They monitor the repository system to detect any changes.

1.3. Continuous Integration on a Cloud vs. On-premises

CI tests and builds can run either on the Cloud providers or on-premise systems. Decisions on the deployment are dependent on various factors. Factors range from the CI tools used, the organizational needs, the security of the data, reliability of the providers, and hosting licenses. These factors vary with the main factor being the consideration of capital costs of set up [6].

Whether to host CI solutions, on-premise, or on cloud is a never-ending debate. Both options have their cons and pros, but it all builds upon the capital employed. The decisions organizations make on this debate are based on the cons and pros of both, the environmental differences of the cloud and premise (local data center), and the architecture involvement by the developers [7].

2. Background

Continuous Integration is founded based on software engineering and the constructs of CI, Continuous delivery (CDE), and Continuous Deployment (CD), as depicted in figure 2. Developers cannot implement CD without CI. Whereas CI has been described and is associated with frequent software building and testing, Continuous delivery (CDE) includes CI and the automated configuration and deployment. On the other hand, continuous deployment (CD) is involved with the corresponding production – it automatically sends software to production once committed. For the sake of this paper, the definitions and how they work together are defined as below.

Continuous Integration (CI)

A subset of Continuous delivery improves software development in terms of quality and speed by building and testing how projects are automated to eradicate errors from developers' manual environment set up. This means that software is periodically tested [6] [4].

Continuous Delivery (CDE)

It includes CI and automated end to end. Software testing and delivery are such that they are deployable to any production environment. It ensures that production is readily offering a reduced risk of deployment and increasing chances of delayed feedback. It requires CI practice to function. [5] [3].

Continuous Deployment (CD)

It includes both CI and CDE and automatically deploys software into production environments eradicating the need for manual production [5] [3].

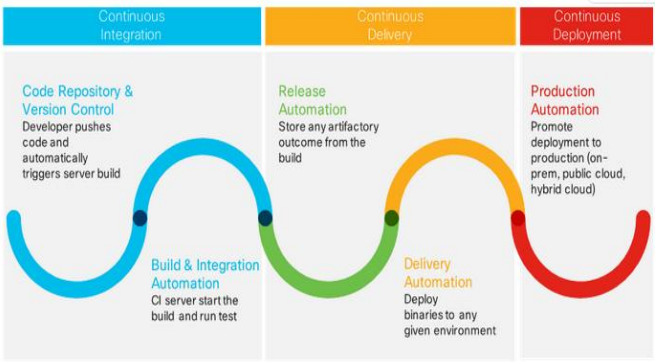

Figure 3. Visualization of CI/CDE/CD

2.1. Advantages of CI

Continuous Integration is favored for lowering the risks of the bugs that could easily break the application by introducing immediate feedback for smaller changes within the systems. Since CI uses agile methods and tools, it helps to get the software to the users fast. The users can get the features newly created, can interact easily and provide feedback within the system for any errors to be corrected quickly [8]. Each stage of CI configuration is meant to reduce the risk of failure. Therefore, it provides a pathway for quality checks and controls to avoid broken or unwanted features in the CI process.

CI provides a short feedback loop. CI provides feedback to users quickly; from the codes sent into it, features are

displayed to the users [9]. Developers get a sense of progress when deployment is made. They can see what is working in the environment via small changes.

The changes in conflicts when merging software are very common occurrences. CI tools help to ascertain that developers do not work on the same components or systems at the same time, thus reducing the chances of conflicts [2]. When developers create features at the same time, CI tools give the developers control through feature flags. CI tools send notifications to the developers.

Continuous integration tools impact businesses by reducing operational costs from fewer outages of their systems. Companies do not have to spend time and resources covering outages. [1] [10] CI cause automatic deployments in the software environment, which leads directly to production. Besides, since the software is delivered to the users faster, businesses use CI to build and improve their products and services for their customers.

2.2. Literature Review on CI integration

Many studies, reviews, and surveys have been conducted on software engineering, - continuous integration, continuous deployment, and continuous delivery. The studies are on software practices (CI, CDE, and CD) are mainly on approaches, tools, and practices for adoption and implementation. These have gained popular reviews. From 2004 to 2016, [4] have extracted 69 papers on continuous Integration. These numerous studies are a confirmation of the interest and attention that software engineering has attracted. Out of those published papers, 56.5 percent were done in the last three years. Only 32.2 percent of the reviewed papers were continuous integration tools. Between 2001 and 2014, 50 studies were on Continuous Deployment [11].

All these studies, mapping, and reviews reveal the emphasis on the need for researchers to look at the aspect of the framework process, integration tools, and deployability for continuous operations. Knowledge is lacking in the deployment of the CI tools either on the cloud or on the premise. Users require information on the kind of CI tools that would best suit their businesses and the benefits of deploying these tools for their businesses. Despite the growing interest and popularity of CI tools and how they are being used to impact businesses, there is very little information and evidence on the factors that influence the choice for cloud versus on-premise hosting of the CI tools.

Lack of this information to businesses and IT teams prevents them from giving solutions and decisions of CI tools. The questions left unanswered include if CI works better on the cloud or on-premise or do the factors for installing CI depend solely on the cost-effective analysis for setting up and deployment of CI. Without the knowledge of what CI tools work better on the cloud or on-premise, businesses and software developers are missing the opportunities and the advantages that come with CI production benefits. They lack the opportunities that CI would impact on the development of their products and services and miss to give their businesses the competitive edge that can be derived from implementing CI tools. Developers that choose to use CI are not utilizing it to its potential without the knowledge of what tool is suited for which environment.

This paper has looked at the studies conducted on continuous integration and reviewed the CI tools, suitable for cloud or on-premise. It has looked at the standard emergent tools in the market and how companies are implementing them to achieve a competitive edge by reviewing the advantages to the businesses.

2.3. Considerations for CI Tools on Cloud Versus on Premise

The future of the internet is the cloud. Cloud continues to be embraced as the most convenient and cost-effective way of managing extensive data [12]. After systems have been developed, the CI servers act to give feedback. The CI servers are the determinants of the problem for rectifications. To receive the right fit with the various CI tools, the management is tasked with the burden of determining the organization's needs. ÇI hostage on cloud software-as-a-service (SaaS) or on-premise (self-hosted) is a priority decision for management [7].

Integration tools fall under various categories; - vendor-supplied on-premise, a third-party on-premise, cloud-delivered, or API web services tools. These tools determine whether the data processes need to be on-premise or cloud-based platforms depending on the support tools requirements.

CI tools to be deployed

Some edge case features with some CI tools like GitHub, Bitbucket, Heroku is best deployed in the cloud as it requires SaaS solutions. The different models of CI tools are necessary for making the cloud versus on-premise decision [13].

Organization needs

Current and future needs of the organization will require different solutions. Unlike larger organizations with volumes of data processes, small enterprises require solutions that can manage few systems according to their operations [6].

Data Security

Organizations that are keen on data security and prevention of data breaches may consider on-premise servers that they can have adequate control over. SaaS allows organizations to focus on core products to maintain the system infrastructure offering flexibility [14]. There are potential data breaches that need to be observed and control measures taken in conjunction with compliance measures.

Cost

SaaS products (cloud) benefit from a lack of hardware or software management. Cloud is easy to set up using Version Control System tools (GitHub or BitBucket).

Businesses initializing and deploying CI on-premises find it time-intensive at the setup stage and CI system initialization. Additionally, apart from the initial capital for hardware and software, authentication, and authorization of users may require management [13] [12].

Flexibility

An on-premise hostage is not rigid. It can be flexed and extended according to company requirements. They can be customized to suit a particular functionality. For example, Jenkins CI tool that contains over 1000 plugins. Additionally, on-premise CI tools support larger development platforms and testing frameworks compared to SaaS – cloud. On-premise solutions have fewer limitations on configurations and favor the majority of CI tools [15].

Reliability of the provider

The primary factor when considering CI tools is the availability of the provider of the cloud. The decisions to be made include the ability to change providers should there be need. There could arise disagreements; providers could move out of business or shut down [15].

Table 1. Comparison table: CI tools on cloud vs on premise

Comparison Table: CI tools on Cloud Vs On premise		
	CI tools on Cloud	**CI tools on Premise**
Hosting (tools)	GitLab, AWS CodePipeline,	GitLab, Jenkins
Cost	Low cost of set up	Cost of set up is high
Data Security	Limited data security control	Better control of security matters
Data Management	Convenient and cost effective	Cost extensive,
Organization Need	Suitable for large projects, multispectral organizations	Suitable for small projects
Flexibility	Rigid	Can be extended to meet current needs of the organization
Reliability	Unreliable Providers	Reliable employees
Configuration & Architecture	Limited	Fewer limitations

2.4. Review of Continuous Integration Tools

For businesses to give their customers value, their systems need to be able to detect errors and correct them quickly and at speed. This can be achieved through CI tools. CI helps to eradicate errors in the system. When systems are tested, built, and ready for deployment, they are taken through a process. The most common CI tools being used in the market today include; GitLab, Jenkins, AWS CodePipeline, and Bitbucket pipelines, among others. These modern tools are enablers in software engineering deployment due to their speed of delivery and quality.

GitLab On Cloud or On-Premise hosting CI tool

GitLab can be hosted both on-premise and cloud. It offers increased productivity and quality. It acts on pull requests. GitLab is new in the market but has gained popularity and named as a leader in CI [16]. GitLab is easy to use, can be scaled up, integrated, and is innovative. It uses the YAML file to offer DevOps full experience to the engineers - it was created to improve the GitHub experience [17]. Apart from its CI function, it also offers monitoring (Kanban boards and time tracking features) within the applications. GitLab's one of its management features is the capability of fast feedback through its pipelines. Over 1500 companies use GitLab, Alibaba, Trivago, Avocode, Freelancer, Ticketmaster, among others.

Jenkins, On-Premise hosting CI tool

Jenkins is an on-premise hosted CI tool. It is driven by community updates and uses recent data. Jenkins is documented, can be customized with plugins, and has over 1000 available that allow all manner of functionalities [6] [17].

Jenkins's flexibility on the distribution of tests and builds makes it popular with most software engineers. It is advantaged in its use on multiple machines. [17]. This feature enables developers to set the CI environment. It is an open-source - free to use as it is licensed under MIT. Jenkins allows developers to set CI as it is written in java and supports tools like Git, Maven, and Mercurial. Jenkins has over 1 million users. It has over 147,000 active installations. Facebook, Netflix, LinkedIn, eBay are among the over 2500 companies that use Jenkins for their CI.

AWS CodePipeline, on cloud hosting CI tool

AWS Codepipeline integrated with third party services like GitLab or other custom plugins. It is fully cloud-hosted and integrated with Amazon web services. Its reliability and speed is a feature that ensures rapid delivery to its users. It has no repository servers, thus delivering software processes using an interface, making it easy to deploy applications during customization of its dependencies.

AWS CodePipeline can be customized for specific needs [1]. It is commonly used by companies based in the U.S. It is suitable for firms having 10,000 employees and over, and revenue of 1000 dollars. Hyatt Hotels Corporation, Asurion LLC, and Smartronix Inc. are among the few companies that use the AWS CodePipeline tool for Continuous Integration.

3. Conclusions

The software has become one of the differentiating factors for industries across the globe. Most of the businesses have seen and embraced the impact CI is creating by ensuring the continuous delivery of feedback to their users. The speed and distribution of the CI tools give businesses a competitive edge.

System developers should install CI tools by evaluating the one that is suitable for them dependent on how it is going to host. To do this, they require an evaluation of factors like price, flexibility, security, and the reliability of the providers. This review has concluded that there is a research gap and

potential for research on CI tools and if to host on cloud or on-premise. The industry lacks information and studies conducted on this aspect.

With the emergent of modern CI tools, coupled with emergent cloud computing issues like data breaches and cloud insecurity, to host or not to host CI tools on the cloud should be a critical factor in every management.

REFERENCES

[1] A. S. Delivery, "Practicing Continuous Integration and Continuous Delivery on AWS Accelerating Software Delivery with DevOps," June 2017.

[2] S. Hamdan and S. Alramouni, "A Quality Framework for Software Continuous Integration," *Procedia Manuf.*, vol. 3, No. 2015, pp. 2019–2025, 2015, doi: 10.1016/j.promfg.2015.07.249.

[3] T. Mikkonen, "Elements for a Cloud-Based Development Environment: Online Collaboration, Revision Control, and Continuous Integration," pp. 14–20, 2012.

[4] M. Shahin, M. Ali, and L. Zhu, "Continuous Integration, Delivery, and Deployment: A Systematic Review on Approaches, Tools, Challenges, and Practices," 2017.

[5] G. Booch, *Object-Oriented Analysis & Design With Application.* Pearson Education, 2006, 2006.

[6] H. Aleksi, D. Taibi, and K. Syst, "Towards Cloud Native Continuous Delivery: An Industrial Experience Report Towards Cloud Native Continuous Delivery: An Industrial Experience Report," no. December 2018, doi: 10.1109/UCC-Companion.2018.00074.

[7] C. Fisher, "Cloud versus On-Premise Computing," pp. 1991–2006, 2018, doi: 10.4236/ajibm.2018.89133.

[8] M. Hilton, T. Tunnell, K. Huang, D. Marinov, and D. Dig, "Usage, costs, and benefits of continuous integration in open-source projects," *ASE 2016 - Proc. 31st IEEE/ACM Int. Conf. Autom. Softw. Eng.*, pp. 426–437, 2016, doi: 10.1145/2970276.2970358.

[9] B. Varanasi and B. Varanasi, "Continuous Integration," in *Introducing Maven*, 2019.

[10] P. Rodríguez *et al.*, "Continuous deployment of software-intensive products and services: A systematic mapping study," *J. Syst. Softw.*, vol. 123, pp. 263–291, Jan. 2017, doi: 10.1016/J.JSS.2015.12.015.

[11] P. E. M.VernercMarkkuOivoa, "No TitleContinuous deployment of software-intensive products and services: A systematic mapping study," *J. Syst. Softw.*, vol. 123, pp. 263–291, 2017.

[12] M. De Donno, A. Giaretta, N. Dragoni, A. Bucchiarone, and M. Mazzara, "Cyber-Storms Come from Clouds: Security of Cloud Computing in the IoT Era," pp. 1–30, 2019, doi: 10.3390/fi11060127.

[13] H. Hai, "SaaS and Integration Best Practices," 2009.

[14] J. Sha, A. G. Ebadi, D. Mavaluru, M. Alshehri, O. Alfarraj, and L. Rajabion, "A method for virtual machine migration in cloud computing using a collective behavior-based metaheuristics algorithm," *Concurr. Comput.* , vol. 32, no. 2, 2020, doi: 10.1002/cpe.5441.

[15] P. Jamshidi, C. Pahl, and N. C. Mendonca, "Pattern-based Multi-Cloud Architecture Migration Pattern-based Multi-Cloud Architecture Migration," no. August 2016, doi: 10.1002/spe.

[16] The Forrester Wave, "Continuous Integration Tools, Q3 2017 report," 2017.

[17] B. Vasilescu, Y. Yu, H. Wang, P. Devanbu, and V. Filkov, "Quality and Productivity Outcomes Relating to Continuous Integration in GitHub."