A Literature Review On Continuous Delivery And Continuous Deployment: Problems, Causes, And Solutions

Hashim H. Alneami
College of Engineering
Embry-Riddle Aeronautical University
Daytona Beach, USA
alneamih@my.erau.edu

***Abstract*—Continuous Delivery (CD) is a software development discipline which facilitates the releasing of software at any time. Additionally, continuous deployment is an extension of CD in which the continuity is extended to the deployment to production stage. With respect to the current implementation of these disciplines, there were issues and challenges standing in the way of adapting CD and continuous deployment by software companies. In this paper, I conduct a literature review on continuous delivery and continuous deployment in which I define both disciplines and explained their differences. I also discuss the problems regarding them as well as the causes and introduce some of the suggested solutions proposed in previous research papers. I then give an analytical exposition on the existing solutions to CD problems.**

Keywords—continuous delivery, continuous deployment, software development

I. Introduction

Continuous delivery (CD) is a software engineering discipline that enables developers to continue producing usable software in short periods of time and make sure the software can be reliably released at any time [1]. Giving the potential benefits of applying CD as a development framework, some companies have adapted CD, and there are many software companies that look forward to use CD as their standard in software development.

Continuous deployment is considered as an add-on to CD in which the team build the software, test it, and deploy it to production automatically without the need of any manual steps between the developer commit and the deployment [2]. Continuous delivery is used to generate quick feedback from users after deployment, so defects can be fixed with reasonable cost [3]. Since the two frameworks are closely similar, it is noted that some people might refer to both of them as equivalent.

Although several companies have gained an interest in adapting a continuous scheme in development, they were faced with major challenges that stopped them from going further. Such problem raises an important case on whether the difficulty of adapting CD is much higher than the predicted value of CD, or the industry is still way behind adapting best practices [2]. According to Neely and Stolt [4], Rally Software, a software company, had to go through challenges regarding their build systems, automated testing frameworks, customer reviews, and communication internally. The issues that must be addressed exist across all of the different phases of the development process. Also, they extends to the production deployment since the continuity includes all of the stages involved in the project.

Almost all of the papers on CD I have read revolve around the problems hindering software companies from implementing the concept of continuous delivery and continuous deployment. In this paper, I discuss the problems that could result from adapting CD as well as continuous deployment along with their causes. I also mention some suggested solutions that could enable companies to apply CD to their development environments.

II. Continuous Delivery

A. What Is Continuous Delivery?

Continuous delivery (CD) is a software development discipline that allows for the release of the software to production at any time [5]. It is essential in CD that automation is applied to the release process in a reliable manner. In CD, several stages are used to ensure whether the software is in a releasable condition [2]. Neely and Stolt [4] states that the frequency of deploying the software is not the focus of attention but the ability to deploy it at will is what matters. Moreover, CD differs from other disciplines by having defining characteristics such as delivering valuable software, deploying in short cycles, releasing at any time, and providing reliable releases [6].

The reason of why continuous delivery is attracting the attention of many software companies, and is adopted by some companies is the benefits it bring to these companies and their overall business goals. One of the benefits is gaining more visibility on the development process. Developers and stakeholders can have a clear vision of what the software is going to carry to their customers. This also means more empowerment to stakeholders as deployment to production starts sooner than the usual duration in the traditional way [2]. Chen [1] also mentions that adopting CD improves customers' satisfaction since their feedback on the software is collected faster. Adopting CD guarantees more reliable releases of the software [6]. The repeated testing before deployment and the frequent releases due to the CD process have enabled for more reliable and better software.

However, companies that make the decision of adopting the continuous delivery approach are bound to work with several problems and challenges in order to have a successful transition to CD. In this paper, I used the findings made by Lakkanen et al. [2] as the base of determining the problems, causes, and solutions regarding adopting CD since these findings provide a thorough analysis of the current literature conducted on continuous delivery.

B. Problems of Continuous Delivery

There are many problems that prevent many software companies from adopting continuous delivery as their software development discipline. In their study, Laukkanen et al. [2] categorized CD problems into seven different themes. "Table I" lists the themes with some of their related problems. Those themes are build design problems, system design problems, integration problems, testing problems,

release problems, human and organizational problems, and resource problems [2]. The last two are concerned with the company's capability for adopting CD before the development process while the rest are concerned with the software development process.

TABLE I. PROBLEMS CATEGORIES AND RELATED PROBLEMS, ADAPTED FROM LAUKKANEN ET AL. [2].

Category	Related Problems
Build design	Inflexible build, complex build
System design	System modularization, unsuitable architecture, internal dependencies, database schema changes
Integration	Defected development flow, slow integration approval, merge conflicts, broken build
Testing	Flaky tests, ambiguous test results, time-consuming testing
Release	Documentation, feature discovery, marketing, more deployed bugs, deployment downtime
Human and organization	Lack of discipline, lack of motivation, lack of experience, more pressure, changing roles, team coordination, organizational structure
Resource	Insufficient hardware resources, network latencies, effort

Chen [1] mentions that the biggest challenge Paddy Power, a software company, has faced since adopting CD was organizational. Disagreements may occur among different departments of the company because of the possible differences in managing and pursued goals. That is why it is critical to establish a cooperative environment among the company's departments and work on ways to encourage the employees to work cooperatively in order to achieve the company's overall goals. It is also noted that getting the approval from all the management levels in the company to adopt CD can require a lot of time [6].

C. Causes of Continuous Delivery Problems

Laukkanen et al. [2] came up with a casual explanation to the problems faced when adopting continuous delivery based on their qualitative coding done on the extracted articles used in their study. When I searched for the articles involving continuous delivery, I found out that casual explanation shown in "Fig. 1" which was created by Laukkanen et al. [2] covers all the possible causes discussed in the related literature. The reported casual explanation draws the casual relationship between the found problems of CD across the different problem categories.

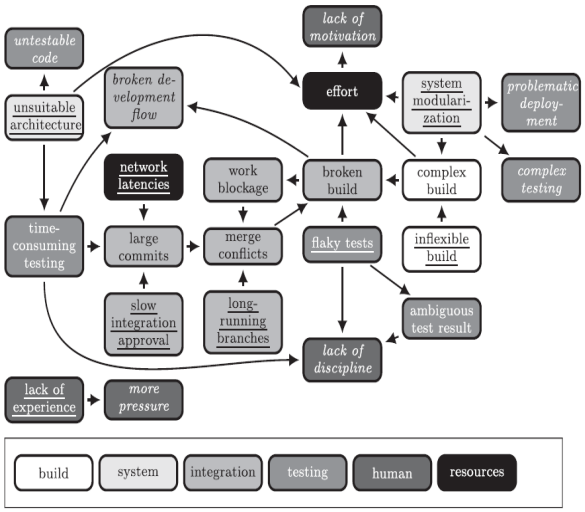

Fig. 1. All reported causal explanations. Roots that do not have any underlying causes are underlined and leafs that do not have any effects are in italics. Adopted from Lakkanen et al. [2].

D. Solutions to Continuous Delivery Problems

Laukkanen et al. [2] divided the proposed solutions into six categories shown in "Table II". The themes include system design, integration, testing, human and organizational, and resource. They are the same as the proposed problems themes except that it does not include the build design category as the research on the topic did not have sufficient explanation on build design solutions. Having said that, it can be possible to overcome the inflexibility in the design by establishing vastly used standards, designing open APIs, and creating a robust plug-in environment [1].

One of the proposed solutions to obtain faster migration to continuous delivery is to create a visual CD pipeline skeleton which can develop a more tangible sense towards CD by the development team as well as sustain their adaption momentum [6]. Furthermore, Neely and Stolt [4] stated that Rally Software was encouraged to invest more money on its GUI testing framework in order to prevent testing problems like flaky tests and slow-time testing.

TABLE II. SOLUTIONS CATEGORIES AND RELATED SOLUTIONS, ADAPTED FROM LAUKKANEN ET AL. [2].

Category	Solutions
System design	System modularization, hidden changes, rollback, redundancy
Integration	Reject bad commits, no branches, monitor build length
Testing	Test segmentation, test adaptation, simulator, test parallelization, database testing, testing tests, comprehensive testing, commit-by-commit tests
Release	Marketing blog, separate release processes
Human and organizational	Remove blockages, situational help, demonstration, collaboration, social rules, more planning, low learning curve, training, top-management strategy, communication
Resource	Tooling, provide hardware resources

III. CONTINUOUS DEPLOYMENT

A. What Is Continuous Deployment?

In continuous deployment, software is deployed continually at a fast pace. It includes the automated testing of software incremental changes and the frequent deployment of these changes to production ecosystems [7]. Such practice can lead to faster reach to customers as the product can be deployed within days or even hours. It differs from continuous delivery (CD) in its automatization to acceptance tests and deployment to production stages.

Additionally, it is clearly noted that companies have different deployment frequency rate per day thus the definition of continuous deployment might not be explanatory for all existing adoptees. However, Rahman et al. [8] suggested a more tailored definition of continuous deployment in order to establish a wider understanding of the topic. They define continuous deployment as "a software engineering process where incremental software changes are automatically tested, and frequently deployed to production environments" [8].

B. Problems of Continuous Deployment

One of the problems faced during the automated testing for a large-scale application such as Facebook is that it is not always clear if a problem has occurred [9]. Leppanen et al. [10] found that none of the fifteen companies, which they studied their implementation of continuous deployment, developed a complete automated deployment pipeline.

During their transition to continuous deployment, companies might also face other challenges regarding their network configuration, upgrade issues, unclarity of the deployment process [11]. In their research, Claps et al. [12] have categorized the potential problems companies might face when adopting continuous deployment into two main categories shown in "Fig. 2". The first category is related to the problems concerned with the technical side of the adaption, and the second category is concerned with the social adoption challenges.

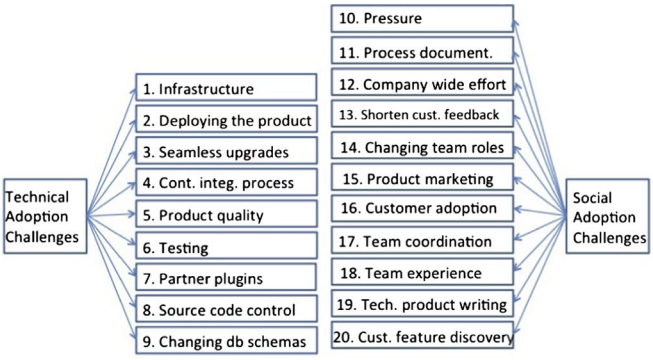

Fig. 2. Technical and social challenges. Adopted from Claps et al. [12].

C. Causes of Continuous Deployment Problems

One of the reasons that can lead to errors that are hard to debug is the continuous configuration changes [7]. Companies with a large codebase are required to conduct changes thousands of times a day which makes the tracking process so difficult to manage.

In addition, customers' distrust in more frequent releases can lead to customer adoption problems. Researches have stated that customers play an important factor on the social challenges side toward adopting continuous deployment [10]. Olsson et al. [11] mentioned that the customer's local and customized configuration can lead to complexity when continuous deployment is set to be adopted.

D. Solutions to Continuous Deployment Problems

As demonstrated in previous research, there are a number of suggested solutions and actions that could help software companies interested in adopting continuous deployment to overcome some of the most critical barriers. Including more organizational levels, especially in the management side, can mitigate some of the problems related to the social adoption side [11].

Companies must find a way to facilitate for more flexibility from their customers and educate them on the perceived benefits of adopting the discipline of continuous deployment. It seems that many customers object the concept of using more frequent releases.

IV. CONTINUOUS DELIVERY VS. CONTINUOUS DEPLOYMENT

Continuous delivery and continuous deployment are software development disciplines that are used to conduct software changes rapidly to customers. Humble and Farley [13] defined continuous deployment as a software process that releases software changes automatically to customers the required automated tests. Fowler [14] defines continuous delivery as the software engineering approach that builds software in such a way that it is releasable at any time. Also, he defines continuous deployment as the software process that actually releases software to production as soon as they are ready, resulting in many deployments to production every day [14]. These definitions of continuous deployment

appear to be alike however, Fowler does not emphasize the automated testing process.

V. Conclusion

When I investigated the existing examples of adopting continuous delivery and continuous deployment, it was highly noted that the barriers standing in the way of a continuous approach fall under one two themes, technical and social. If company decides to adopt a continuous development process, it will have to deal with several organizational problems involving development team adaption, management approval, and others. However, the company is expected to embrace the adopting of continuity as the perceived benefits are collected in the early stages.

Technically, most companies that adapted continuity face challenges regarding applying automated testing and complexity problems. Companies should focus more on improving their GUI testing framework as well as working toward creating standardized solutions. Specially at the beginning, they might need to enhance their monitoring process to ensure that the release is free from difficult to debug errors.

Furthermore, I noticed that the amount of research papers published on the topic is still lacking. We need more research studies conducted specifically for continuous delivery and continuous deployment. Also, publication bias can be seen through available work as the judgement established might indicate less problematic condition than in reality.

References

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

[2] Laukkanen, E., Itkonen, J., & Lassenius, C. (2017). Problems, causes and solutions when adopting continuous delivery—A systematic literature review. Information and Software Technology, 82, 55-79.

[3] Fitz, T. (2009). Continuous deployment.

[4] Neely, S., & Stolt, S. (2013, August). Continuous delivery? easy! just change everything (well, maybe it is not that easy). In 2013 Agile Conference (pp. 121-128). IEEE.

[5] Fowler, M. (2013). Continuous delivery. martinfowler. com.

[6] Chen, L. (2017). Continuous delivery: overcoming adoption challenges. Journal of Systems and Software, 128, 72-86.

[7] Parnin, C., Helms, E., Atlee, C., Boughton, H., Ghattas, M., Glover, A., ... & Stumm, M. (2017). The top 10 adages in continuous deployment. IEEE Software, 34(3), 86-95.

[8] Rahman, A. A. U., Helms, E., Williams, L., & Parnin, C. (2015, August). Synthesizing continuous deployment practices used in software development. In 2015 Agile Conference (pp. 1-10). IEEE.

[9] Feitelson, D. G., Frachtenberg, E., & Beck, K. L. (2013). Development and deployment at facebook. IEEE Internet Computing, 17(4), 8-17.

[10] 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.

[11] Olsson, H. H., Alahyari, H., & Bosch, J. (2012, September). Climbing the" Stairway to Heaven"--A Mulitiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software. In 2012 38th euromicro conference on software engineering and advanced applications (pp. 392-399). IEEE.

[12] Claps, G. G., Svensson, R. B., & Aurum, A. (2015). On the journey to continuous deployment: Technical and social challenges along the way. Information and Software technology, 57, 21-31.

[13] Humble, J., & Farley, D. (2010). Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader). Pearson Education.

[14] Fowler, M. (2010). Featuretoggle. October, 29, 1-4.