Enhancing Lean Software Development by using DevOps Practices

Ahmed Bahaa Farid
Information Systems Dept,
Faculty of Computers and Information, Helwan University,
Cairo, Egypt

Yehia Mostafa Helmy
Business Information Systems Dept,
Faculty of Commerce & Business Administration, Helwan University,
Cairo, Egypt

Mahmoud Mohamed Bahloul
Business Information Systems Dept,
Faculty of Commerce & Business Administration, Helwan University,
Cairo, Egypt

***Abstract*—Competition between companies has made a great pressure to produce new features continuously as fast as possible, subsequently successful software companies needs to learn more about customers and get new features out to them more rapidly. Lean software development cannot integrate between development and operation teams. DevOps enables this merge between them and creates operational parts as one part of the development process and made it up to date during the development phase, so reduced errors during the deployment. The purpose of this paper is to investigate how one can use devOps practices to improve the performance of lean software development production process and introduce a new framework that merge lean and devOps process. The research has been evaluated on a sample of 2 departments in Faculty of Commerce at Helwan University. The results of this work have led to reduce the response delivery time for customers and rapid feedback provides accurate expectations for customer needs that lead to lower levels of deployment pains and lower change fail rates.**

Keywords*—Lean software development; DevOps; development & IT operations; continuous delivery; monitoring; continuous integration

I. INTRODUCTION

Changing business needs always required providing products faster to market due to competition among software companies which puts an increasing pressure to produce new features extremely fast. Projects in software development field always faced risks or problems as bugs, failure, past deadline and poor quality etc. In recent years, software companies need way in which you manipulate problems such as long development life cycles and rapidly changing requirements from customer [1].

Lean methodology is designed to minimize the wastes of resources that do not add any customer value to products [2].

Lean depends mainly on continuous improvement and to achieve this any defect or problem that may occur in the delivery process must be detected to get feedback continuously [3].

DevOps is set of practices and principles that is trying to improve life-cycle as a whole through integration between development and operations teams to reduce the release cycles and increase number of software deliveries [4].

The paper is organized as follows: Section 2 presents the background and related research of Lean Software Development and DevOps. Section 3 describes the research problem. Sections 4 and 5 describe the research goals, approach, and assumptions. Section 6 shows the theoretical mapping of the two approaches. Section 7 shows overall results of the theoretical mapping. Section 8 describes the background of the analyzed study. Section 9 summarizes the assessment results from the study. In Section 10, the results of this study are discussed. The last section concludes the paper with the key findings, research limitations and future work.

II. OVERVIEW

This section consists of three parts. The first part presents an overview of Lean Software Development. The second part provides an overview of DevOps. The final part gives Related Work.

A. Lean Software Development

Lean software development provides a set of principles to minimize wastes and maximize the customer value in software processes. Mary and Tom Poppendieck [5] have formulated a set of principles for the application of Lean thinking into software development.

There are seven main principles in Lean development process as the following: Eliminate waste, Amplify Learning, Decide as Late as Possible, Deliver as Fast as Possible, Empower the Team, Build Integrity In and See the Whole [5].

Eliminating waste is the first principle that explains the Waste as any unnecessary activities that add cost or time without adding value to the customers [6, 3]. There are many wastes that transferred by Poppendieck and Poppendieck [5] from manufacturing to software development are: partially completed work, extra Features, extra processes, task switching, Handoffs (Motion), delays (waiting) and defects.

The way lean works is by creating more value for customers with fewer resources through remove wastes from activities and eliminating whenever possible those steps that do not create value to enhance quality products [7].

Take the right time to adding the real value that satisfied the customer through remove anything that doesn't either add customer value directly or add knowledge about how to deliver that value more effectively [8].

B. DevOps

Market needs are changing continuously. Therefore, there is always a need to adapt continuously to market needs and deliver quickly. DevOps appeared as a result of integration of development and operators team members to increase speed of new software releases and reduce time to respond to customer needs and changes [9].

Dubois [10] found that the developers had no knowledge of what was happening to the application on the deployment infrastructure, and that the operations team also did not care of the planning and priorities of projects.

Each of these teams has a different goal, development teams are interested for deliver new features and operations teams are interested for stability [11].

DevOps is approach that emerging to bridge this gap between these two teams and to achieve collaboration between them. The deployment process needs to be highly automated to enable continuous delivery of software so the DevOps provides a huge variety of practices to implement holistic deployment automation [12].

DevOps will improve productivity through accelerated customer feedback cycles and reduced overhead and rework in addition to provide a competitive advantage to a business through three dynamic capabilities. First a holistic collaborative work involving multiple stakeholders from business and software functions whereby speeding continuous planning and innovation of ideas [13]. Second continuously deploying of software builds through automating software delivery processes and eliminating wastes and this is known as continuous delivery [14]. Third Identify problems as early in the process and notify development teams as quickly as possible that means providing a feedback loop for continuous learning from customers by monitoring and optimizing the software driven innovation [15].

C. Related Work

There have been a number of publications focusing on the relationship between Lean Software Development and Agile, understanding of the combined use of agile and lean approaches in software development and investigate how agile and lean approaches have been combined in software development [16,17]. On the other hand, Shahid Mujtaba identify waste-related problems in a software product customization process by using value stream maps (VSM) [18] but they did not provide empirical evaluation of value stream maps in the software engineering. Pilar Rodríguez [19] presented some of challenges when applying Lean Software Development as achieving flow, transparency and creating a learning culture but unfortunately did not elaborate on ways to overcome these challenges. Henrik Jonsson [20] provided a framework for lean software development but he did not provide empirical evidences.

Finally, Pilar Rodríguez et al. [21] identified some bottlenecks in Lean Software Development as lack of collaboration between the hardware and software teams and short feedback loops from teams that led to not easy to involve business management to prioritize the backlog and defining the feature content but did not provide ways to overcome these challenges so this study is concentrated on determine challenges of lean software development to enhance lean process by using DevOps.

III. Proposed Definition

Business, competitive advantage, market and customer needs are forcing organizations to develop and deploy applications, products, and services at a fast rate. When talking about lean there are lack of coordination between different elements, tasks or features led to barriers in achieving process flow. Problems occurring in the integration between features or there is not enough time to apply these features and monitor them by operational teams. So it is needed to new approach to enable early integration between development and operation teams to enable merge between them and create operational parts as one part of the development process and made it up to date during the development phase, so reduced errors during the deployment.

IV. Research Approach

The goal of this research was to study how using DevOps practices to enhance lean software development process through identify reasons of lean wastes and DevOps role to overcome this reasons and provide framework that allows integration between them. The approach was developed in Three-phase model. The first phase was to determine reasons of lean wastes and the role of DevOps in overcoming them. The second phase was to provide framework that allow merging between two approaches with implement an empirical study that applied to a sample university to make sure that merging is possible. The Third phase has been conducted in order to measure the effect of this merge.

V. Research Assumptions

- Using lean software development process and DevOps practices together will be useful for the organizations to achieve better team productivity and predictability of problems.
- Rapid feedback provides accurate expectations for customer needs that lead to lower levels of deployment pains and lower change fail rates.

VI. Study Phase 1: The Challenges of Lean Software Development and the Role of DevOps to Overcoming Them

Any organization needs to find and fix issues early before they are available in production phase. For this reason, you need to team members work together as DevOps team, so the following section will explain the causes of the lean software development wastes and the role of DevOps in improving and addressing the following wastes:

LW1: Delays:

Reasons:

- Any worker in the system might be not understanding of the client's needs in a particular item partially, or

have not all of the knowledge that you need to complete your task, so delays work to be achieved.

- Approval on the work to transfer it from phase to another phase for example the completion of development work to beginning of deployment.
- Lack of trust between team members causes delays to the work.
- Delayed feedback lead to start with feature may be not needed by the customer.

DevOps practices applied to solve this reasons:

- Be sure to make the knowledge necessary available to execute the project and delivered on time by using a clear vision of the elements that will be implemented.
- Make sure the client's needs, which wants always through feedback iterations.

LW2: Extra Features:

Reasons:

- Misunderstanding of the customer expectations requirements.
- More features more testing that means quality assurance team will be busy with a lot of tests and this also will affect the developers.
- Developers add items that they believe will improve the product in the end, and it will add a better perspective of the product. This may have on increase features as a result of suggest other new features.
- Customers add items that they believe it will benefit the project in the end.

DevOps practices applied to solve this reasons:

- All extra items, features or codes must be tracked, compiled, integrated, tested and maintained so we need continuous improvement to delivery software processes.
- We need to more careful about what we produce and the real needs of the customer therefore it must take more interest about the product backlog. Adding item's functionality when it is necessary to meet a need of the customer. As a result, it helps to discover features that not added value to customer and will not developed.
- Continuous feedback as a DevOps practices enables rapid and continuous expectation of customer needs and give the ideal system, therefore eliminated extra features through tracking product easily and remove features that are not needed.

LW3: Re-Learning (Extra Processing):

Reasons:

- Passing knowledge from person to other person that required re-explain work to provide value and re-sharing it.
- Distributed tasks that lead to exchange between tasks, and thus loss of time to restore focus on the task to be performed.
- Captured poor knowledge lead to rediscovery of that same knowledge.

DevOps practices applied to solve this reasons:

- Sharing knowledge and meeting between team members and depend on experts that can provide the benefit of your project. Everyone in the team can face problem in his work so we can reduce rediscover something from another developer.

LW4: Motion:

Reasons:

- Need for more information because of a lack of understanding about task.
- Distributed work between team members lead to indirect communication between them.

DevOps practices applied to solve this reasons:

- Use visual boards to this can be helpful to reduce hand-off time.
- It's good to have cross-functional teams to create a single project team.
- Shorten feedback loops reduce the number of hand-offs.

LW5- Partially work done:

Reasons:

- Poor analysis for customer needs lead to problems in establishing proper requirements.
- Lack of active participation between environments as development and operations which leads to disruption work in production environment.
- The discovery of errors or detected later may lead to incomplete feature.

DevOps practices applied to solve this reasons:

- DevOps enables to see the whole system end to end view of system from inception to deployment the system to the customer through tracking work done. If feature "done" means "developers declare it to be done and deliver it into a production environment".
- Product backlog must be declared before started in execute features and determine the person who works on specific task and the time to do it so we need coordination between the team and the product owner.

LW6: Task Switching:

Reasons:

- There are tasks will deliver value to a different customer and every one of customers want realize value as

possible as and which results in switching between tasks to satisfy all customers.

- Delays may lead to task switching. If developer doesn't have the knowledge that needed to complete the task this will lead to switch to another task.

DevOps practices applied to solve this reasons:

- Make sure that you have a detailed knowledge well before the start of the task until does not happen disabled during execute the task.
- Determine the priority of the stories during planning phase. This prevents task switching.

LW7: Defects:

Reasons:

- Lack of proper the automation testing.
- Lack of understanding of the items clearly according to the standards that have been determined in advance.

DevOps practices applied to solve this reasons:

- Do some practices that enable team members to communicate with each other so easily give appropriate comments on the work before entering into began defect.
- In order to be successful project there must be a strong automated test which is a very important element to discovery defects early.
- Make sure you have a complete understanding about the item being implemented.

In short, DevOps can overcome the causes of lean software development wastes through using DevOps practices. This result can be displayed through table I.

VII. STUDY PHASE 2: EMPIRICAL STUDY AND INTEGRATED FRAMEWORK FOR LEAN SOFTWARE DEVELOPMENT AND DEVOPS

Once theoretical study were established through determine the causes of Lean Software Development wastes and DevOps role in addressing them, using framework to enhance Lean Process as shown in Fig. 1.

TABLE I. MATRIX OF LEAN SOFTWARE DEVELOPMENT WASTES AND DEVOPS PRACTICES

Lean Wastes/DevOps Practices	Continuous Planning	Continuous Feedback	Continuous Delivery	Continuous Integration	Continuous Testing	Continuous Monitoring
Waiting	√	√				
Extra Features	√	√	√		√	√
Extra Processing	√					
Motion	√	√		√		
Partially done work	√		√			
Task switching	√	√				
Defects	√				√	√

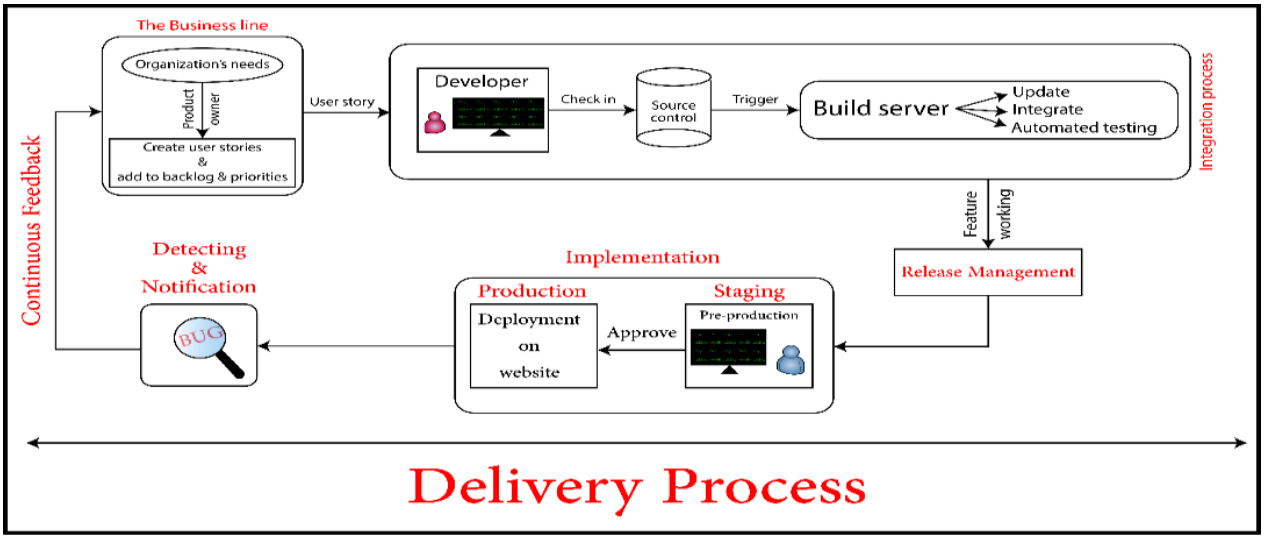

Fig. 1. Lean and DevOps framework.

A. Study description

Applying this framework through empirical study using Evaluation website to provide ability to students to participate in evaluation of the doctors who taught and the subjects studied in the previous term in Faculty of Commerce Helwan University, also used Visual Studio Team Services used to manage and track work items cross the team to address the entire software development lifecycle and Application Insights to help in diagnose issues and to understand what users actually do with your app. There are two programs in Faculty of Commerce: the first BIS program which has been used for lean development software and the second FMI program which has been used for DevOps process. The framework can be illustrated by the following steps:

1) The Business Line:

Any successful project or software the first step will be to plan and communicate continuously between the organization and the team members responsible for this work. Continuous planning of DevOps practices allows doing that by always having a product backlog and prioritizing each item. As shown in Fig. 2. The project is divided into a set of User Stories and each user story is divided into a group of Tasks.

Product owners guide the development through creating a clear and inspiring product vision together with the team, ensure that customer value is transparent for everyone and focus on the highest priority goal at any one time.

2) Integration Process:

Developers can now integrate code which allows checking at any time whether the product meets what the customer really values continuously. Continuous integration (CI) allows team members integrate their work frequently and verified by an automated build and testing to detect integration errors as quickly as possible as shown in Fig. 3.

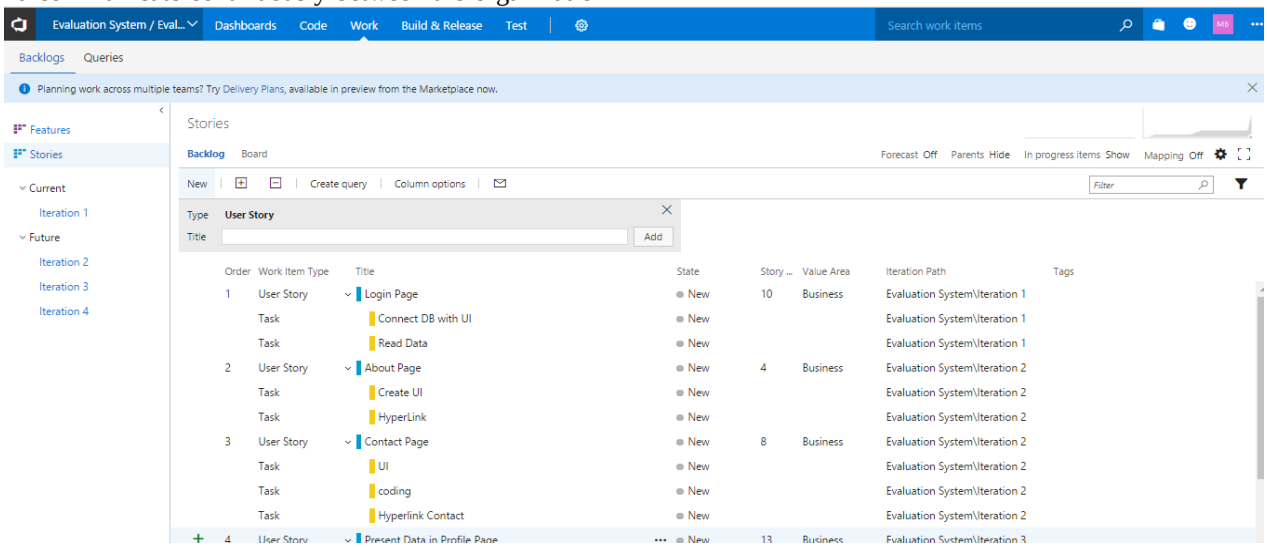

Fig. 2. User stories and Tasks I.

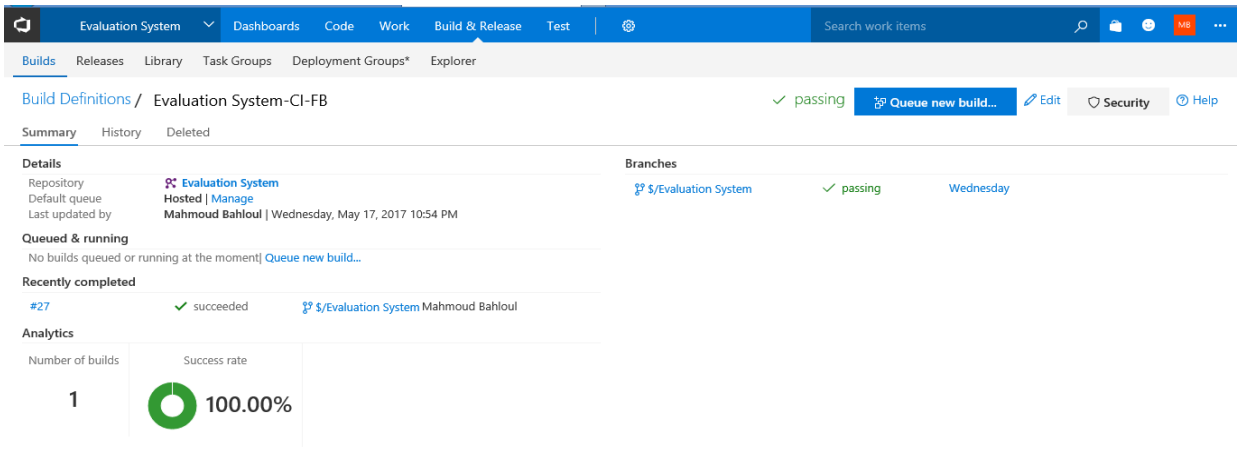

Fig. 3. Build process.

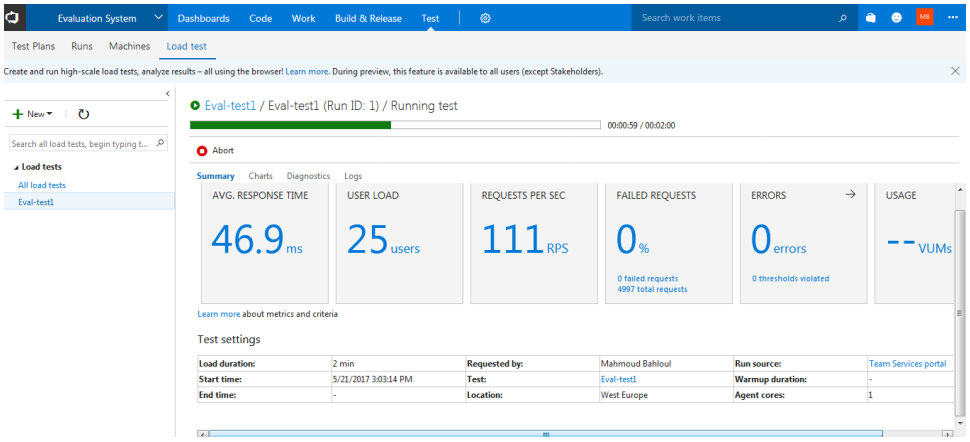

Fig. 4. Quality assurance environment.

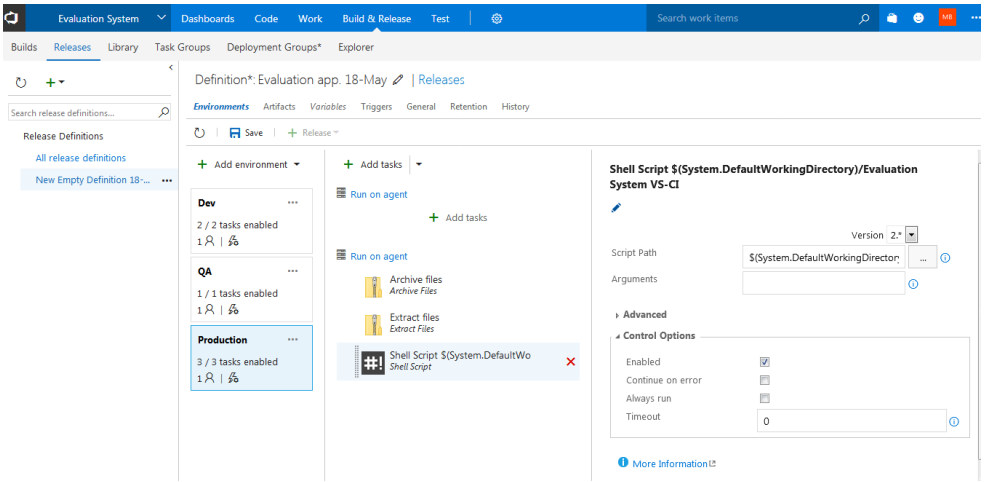

Fig. 5. Production environment.

The version control used to control changes in the source code and other software elements. The build server first check out the project from source control and executes targets from the build automation tool. It creates an integration build when any changes in the version control repository are made. Also as part of continuous integration process, we can do automated testing of the whole application along with code analysis to detect any issues or bugs and to analysis the code to obtain a quality build from the CI server.

When finish the developer from writing code and all unit tests is true, Quality Assurance (QA) team can only ensure that once a product is released it meets and matches all the quality criteria's and to build and ensure that the several testing and validation processes are improved continuously through automated testing which provides tests quickly as shown in Fig. 4.

3) Implementation:

The Release management is process responsible for planning, scheduling, controlling the build, testing and deploying release to increase numbers of successful releases through avoid unexpected outcomes. The quality assurance team gives the change its seal of approval. The change moves on to the staging server, where final acceptance testing commences. The staging or a pre-production environment is providing for the end-users to test the application. When the end-user accepts about this feature, the release to the production environment is performed as shown in Fig. 5.

4) Delivery Process:

The main focus of DevOps is all about delivering *value* very quickly to users and customers. Communication between team members can definitely make it easier to automate deployments. If you need the team who write the deployment scripts to collaborate with the people who manage the environments and run the scripts, you need to continuous delivery.

Emphasis on collaboration and feedback is very important to get successful working software, so DevOps focus on everyone involved in a project must communicate with each other constantly and to continuous delivery new values and releases for customers to be able to achieve a very short time to market as shown in Fig. 6.

5) Detecting, Notification & continuous Feedback:

Using DevOps and lean software development all customer expectations can be answered quickly. It helps us to get connected to the users and to act rapidly against them and

understand how they use their system. As shown in Fig. 7 it explains how trace data can detect errors at the same time.

This direct tracking of data enables me to know why and where the error occurred and how often specific events happen in my web app as student evaluated specific course. It has become very easy identify the features that are used constantly or not used. This continuous follow-up allow to know the most visited pages of the users as well as the pages do not care about as shown in Fig. 8.

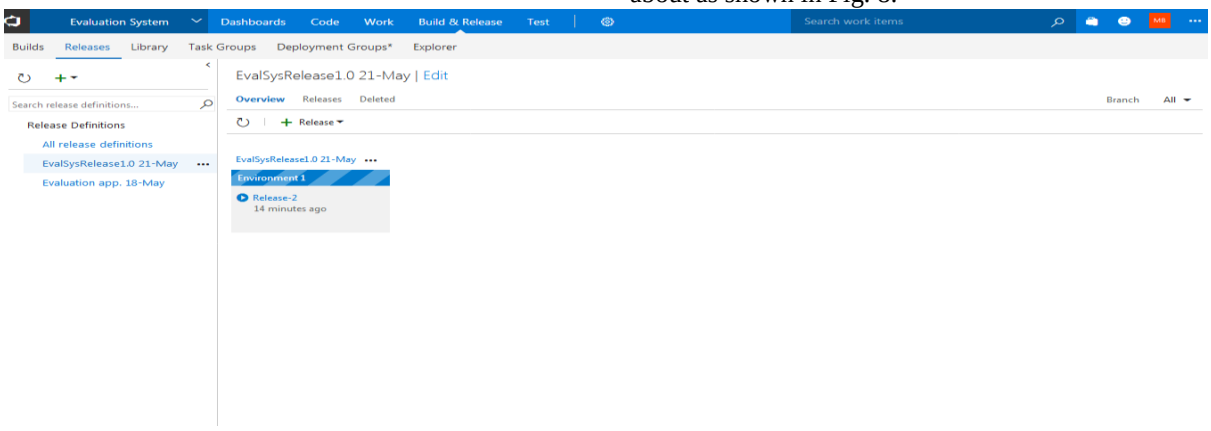

Fig. 6. Release delivery.

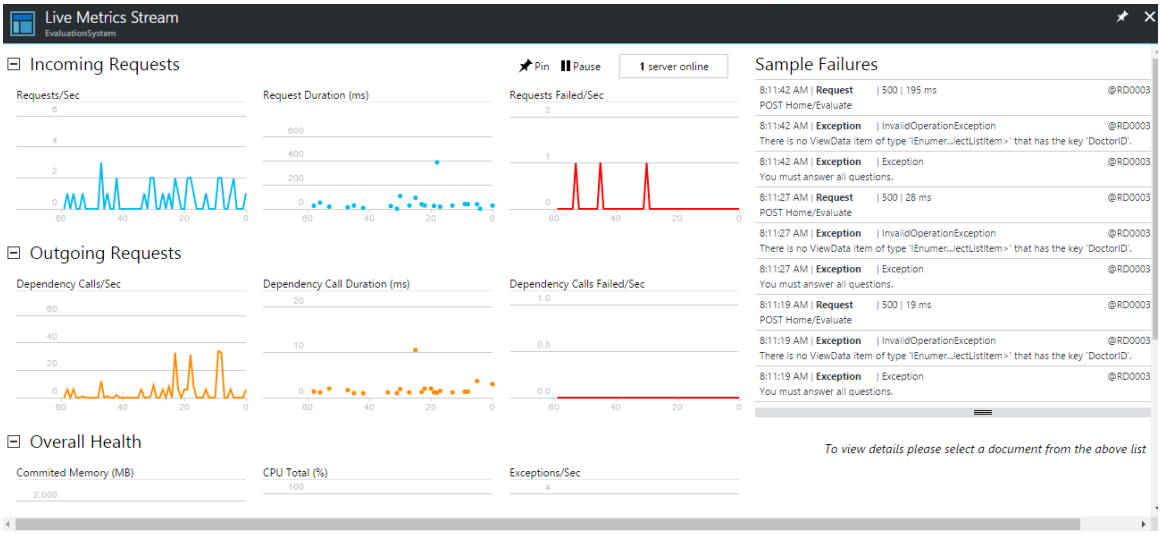

Fig. 7. Live detection.

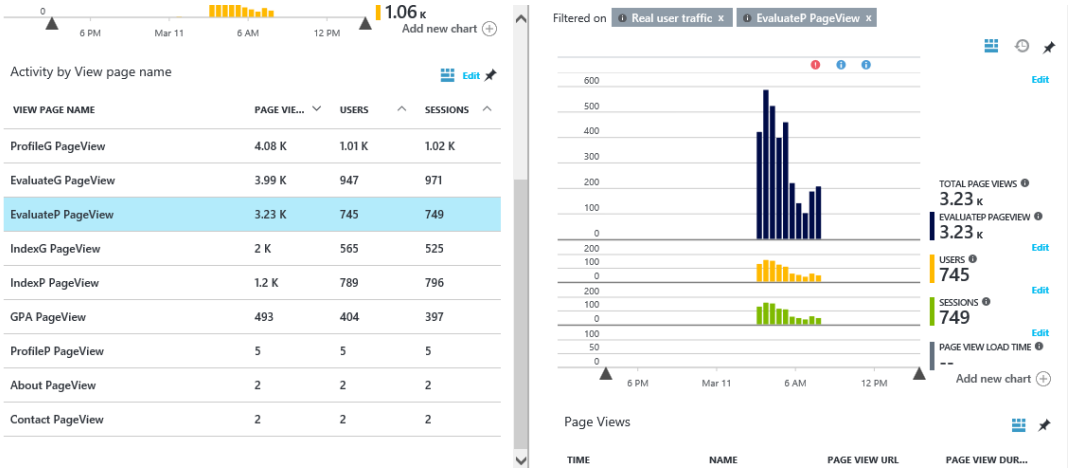

Fig. 8. Page views.

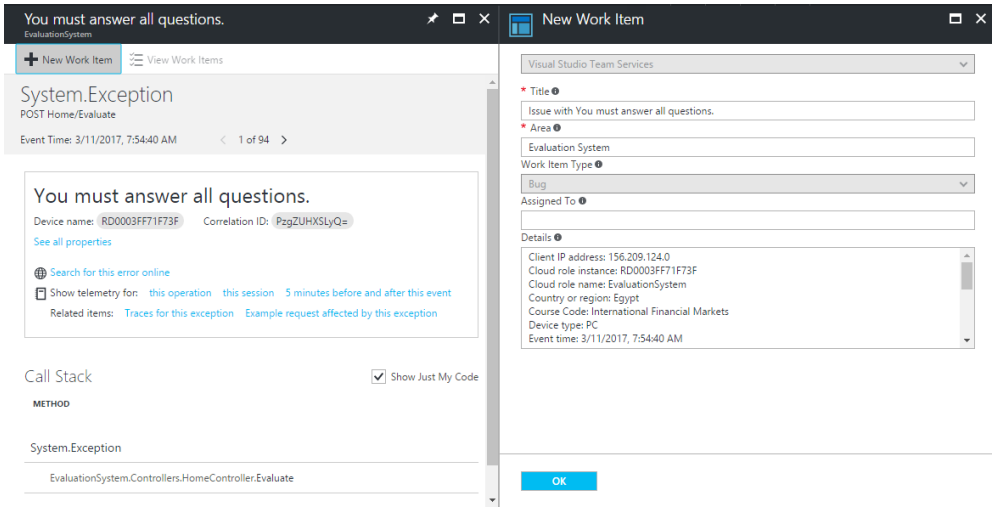

Fig. 9. New work item (bug).

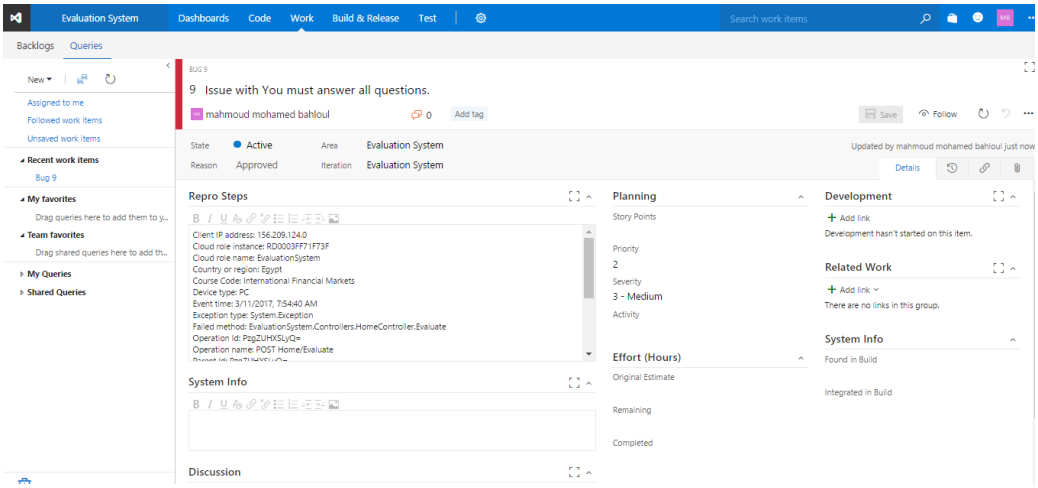

Fig. 10. Issue assigned to development team.

The success of any system not only depends on effective monitoring, but should be followed by a quick warning of any possible errors or problems in the system. Continuous monitoring leads to get immediate feedback on a deployment. When the operations team members discover the error, they send the data obtained to the developers to address and fix it as soon as possible as shown in Fig. 9.

The detected problem reaches to the developer in his queries as a bug to solve it and he gives it his priority as shown in Fig. 10.

VIII. Study Phase 3: Verifying Applicability of Research Work

The project was divided into 4 iterations. When using LSD the number of hours worked was expected to be 132 hour, but with the emergence of these errors and increased these hours became actual hours are 139 hour as shown in Fig. 11.

But when applying DevOps the number of hours worked was expected to be 132 hour, but with the emergence of these errors and increased these hours became actual hours are 135 hour as shown in Fig. 12.

Process	Start Date	Estimate End	Estimate Working Hours	Estimate Bugs	Actual End	Actual Hours
iteration 1	1/2/2017	4/2/2017	24		4/2/2017	24
iteration 2	6/2/2017	9/2/2017	24	2	9/2/2017	26
iteration 3	11/2/2017	19/2/2017	44	2	19/2/2017	46
iteration 4	21/2/2017	28/2/2017	40	3	28/2/2017	43

Fig. 11. Lean process.

Process	Start Date	Estimate End	Estimate Working Hours	Estimate Bugs	Actual End	Actual Hours
iteration 1	1/2/2017	4/2/2017	24		4/2/2017	24
iteration 2	5/2/2017	8/2/2017	24	1	8/2/2017	25
iteration 3	9/2/2017	17/2/2017	44	1	17/2/2017	45
iteration 4	18/2/2017	25/2/2017	40	1	25/2/2017	41

Fig. 12. FMI DevOps process.

Enhancement	Delivery Date	Resolved Date	# ofDays
I1: Writing characters in student id	5/2/2017	9/2/2017	4
I2: Repeatation doctors and students data	20/2/2017	28/2/2017	8
I3: Postback evaluate page	2/3/2017	3/3/2017	1

Fig. 13. Enhancement issues in lean process.

Enhancement	Delivery Date	Resolved Date	# ofDays
I1: Writing characters in student id	5/2/2017	6/2/2017	0
Repeatation doctors and students d	20/2/2017	21/2/2017	1
I3: Postback evaluate page	2/3/2017	2/3/2017	0

Fig. 14. Enhancement issues in DevOps process.

Issue	Number of affected	Total	Presentage
I3: Postback evaluate page	50	320	15.625

Fig. 15. Number of affected particular issue in lean process.

Issue	Number of affected	Total	Presentage
I3: Postback evaluate page	4	1200	0.333333333

Fig. 16. Number of affected particular issue in DevOps process.

In LSD the errors detected by the user is then reported to the operations team and the development team members is notified of this error by regular meetings or team reminder email. Consequently, it was a loss of a lot of time until the update of what is new and satisfy the needs of customers. As shown in Fig. 13 the number of days needed to solve some of the problems that appeared in the system.

Using DevOps errors are detected at the time they occur or even there are notifications of an error, this led to reduce the number of days to resolve the errors. As shown in Fig. 14 the number of days needed to solve some of the problems that appeared in the system

In LSD Delay in product improvement response lead to leaving the customer to the product as a result of the number of customers affected by the problem is large and consequently leads to the loss of a large number of customer loyalty to the company. As shown in Fig. 15 the number of students affected by Postback Evaluate Page is 50 students from total 320.

In DevOps finding errors quickly led to reduce the number of students to be affected by errors and it becomes very small. As shown in Fig.16 the number of students affected by Postback Evaluate Page is 4 students from total 1200.

IX. Discussion:

From the previous study, the results can be summed up, which displaying the role of DevOps to enhance and improve Lean Software Development as shown in Fig. 17 and 18.

Rapid feedback provides accurate expectations for customer needs that lead to lower levels of deployment pains and lower change fail rates. The percentage of students who were affected by specific issue through using lean software development was greater than the percentage of students who were affected by the DevOps.

	Lean	DevOps
Presentage of affected	15.625	0.333333333
Predictability	0.2	0.8
Delivery Time	10	90

Fig. 17. Summarizes up the results of using both lean software development and DevOps.

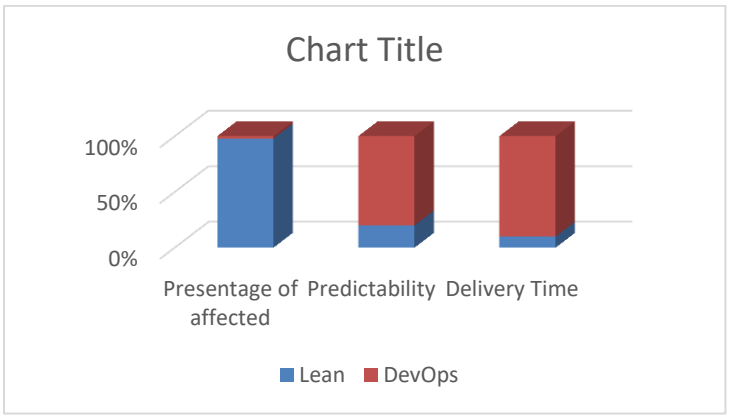

Fig. 18. Summarizes up the results of using both lean software development and DevOps.

Improve the company's time and market potentially from months and weeks to days and hours in addition to faster time to value through reduced cycle times. Consequently, the predictability percentage of problems that may occur or know and follow what the user is doing through DevOps is much higher than the lean software development.

Changing business needs always required to provide products for faster time to market due to Competition among software companies puts an increasing pressure to produce new features extremely fast. Consequently, DevOps increases the rate of software delivery more than lean software development.

X. Conclusion and Future Work

As explained above in this paper how using practices of DevOps to enhance Lean Software Development that allows to cover the entire life-cycle from development to operations environments. Enhancing of Lean Software Development process was done through determine the causes of the lean software development wastes and how using DevOps practices in improving and addressing this wastes. The reasons identified in the Lean Software Development and the role of DevOps in the addressing or improvement this wastes lead to create a new lean and DevOps framework that used to enhance process of Lean through reduce time to market and increases the rate of software delivery.

Changing business needs always required to provide products for faster time to market due to Competition among software companies puts an increasing pressure to produce new features extremely fast so that need to rapid feedback that provides accurate expectations for customer needs that lead to lower levels of deployment pains and lower change fail rates.

In the light of that, it is suggested as future works: The field of software engineering is changing very quickly so more studies will help the companies to apply this framework in other large environments and to use DevOps Practices to enhance other process methods as Scrum or XP.

References

[1] MULLAGURU, Surendra Naidu. Changing Scenario of Testing Paradigms using DevOps–A Comparative Study with Classical Models. *Global Journal of Computer Science and Technology*, 2015, 15.2.

[2] RODRÍGUEZ, Pilar, et al. Survey on agile and lean usage in finish software industry. In: *Empirical Software Engineering and Measurement (ESEM), 2012 ACM-IEEE International Symposium on*. IEEE, 2012. p. 139-148.

[3] POPPENDIECK, Mary; CUSUMANO, Michael A. Lean software development: A tutorial. *IEEE software*, 2012, 29.5: 26-32.

[4] WETTINGER, Johannes; ANDRIKOPOULOS, Vasilios; LEYMANN, Frank. Automated capturing and systematic usage of DevOps knowledge for cloud applications. In: *Cloud Engineering (IC2E), 2015 IEEE International Conference on*. IEEE, 2015. p. 60-65.

[5] M. Poppendieck and T. Poppendieck, Lean Software Development: An Agile Toolkit (The Agile Software Development Series). 2003, p. 240.

[6] AL-BAIK, Osama; MILLER, James. Waste identification and elimination in information technology organizations. *Empirical Software Engineering*, 2014, 19.6: 2019-2061.

[7] POPPENDIECK, Mary; POPPENDIECK, Tom. *Implementing lean software development: From concept to cash*. Pearson Education, 2007.

[8] JONSSON, Henrik; LARSSON, Stig; PUNNEKKAT, Sasikumar. Synthesizing a comprehensive framework for lean software development. In: *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*. IEEE, 2013. p. 1-8.

[9] HTTERMANN, Michael. *DevOps for developers*. Apress, 2012.

[10] DEBOIS, Patrick. Agile infrastructure and operations: how infra-gile are you?. In: *Agile, 2008. AGILE'08. Conference*. IEEE, 2008. p. 202-207.

[11] HUMBLE, Jez; MOLESKY, Joanne. Why enterprises must adopt DevOps to enable continuous delivery. *Cutter IT Journal*, 2011, 24.8: 6.

[12] HUMBLE, Jez; FARLEY, David. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader)*. Pearson Education, 2010.

[13] LEHTOLA, Laura, et al. Linking business and requirements engineering: is solution planning a missing activity in software product companies?. *Requirements engineering*, 2009, 14.2: 113-128.

[14] NEELY, Steve; STOLT, Steve. Continuous delivery? easy! just change everything (well, maybe it is not that easy). In: *Agile Conference (AGILE), 2013*. IEEE, 2013. p. 121-128.

[15] CUKIER, Daniel. DevOps patterns to scale web applications using cloud services. In: *Proceedings of the 2013 companion publication for conference on Systems, programming, & applications: software for humanity*. ACM, 2013. p. 143-152.

[16] WANG, Xiaofeng. The combination of agile and lean in software development: An experience report analysis. In: *Agile Conference (AGILE), 2011*. IEEE, 2011. p. 1-9.

[17] WANG, Xiaofeng; CONBOY, Kieran; CAWLEY, Oisin. "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. *Journal of Systems and Software*, 2012, 85.6: 1287-1299.

[18] MUJTABA, Shahid; FELDT, Robert; PETERSEN, Kai. Waste and lead time reduction in a software product customization process with value stream maps. In: *Software Engineering Conference (ASWEC), 2010 21st Australian*. IEEE, 2010. p. 139-148.

[19] RODRÍGUEZ, Pilar, et al. Building lean thinking in a telecom software development organization: strengths and challenges. In: *Proceedings of the 2013 international conference on software and system process*. ACM, 2013. p. 98-107.

[20] JONSSON, Henrik; LARSSON, Stig; PUNNEKKAT, Sasikumar. Synthesizing a comprehensive framework for lean software development. In: *Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference on*. IEEE, 2013. p. 1-8.

[21] RODRÍGUEZ, Pilar, et al. Combining lean thinking and agile methods for software development: A case study of a finnish provider of wireless embedded systems detailed. In: *System Sciences (HICSS), 2014 47th Hawaii International Conference on*. IEEE, 2014. p. 4770-4779.