IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 3, 2019, accepted December 18, 2019, date of publication December 25, 2019, date of current version January 8, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2962257

Agile Software Development Using Cloud Computing: A Case Study

MUHAMMAD YOUNAS[1,2], DAYANG NORHAYATI ABANG JAWAWI[2], AHMAD KAMIL MAHMOOD[3], MOHAMMAD NAZIR AHMAD[4], MUHAMMAD UMER SARWAR[1], AND MOHD YAZID IDRIS[2]

[1]Department of Computer Science, Government College University Faisalabad, Faisalabad 38000, Pakistan
[2]Faculty of Engineering, School of Computing, Universiti Teknologi Malaysia, Johor Bahru 81310, Malaysia
[3]High Performance Cloud Computing Center (HPC), Universiti Teknologi Petronas, Sri Iskandar 32610, Malaysia
[4]Institute of Visual Informatics, Universiti Kebangsaan Malaysia, Bangi 43600, Malaysia

Corresponding author: Muhammad Younas (younas.76@gmail.com)

This work was supported in part by the Universiti Teknologi Malaysia (UTM) for Transactions disciplinary Research under Grant 06G23, and in part by the Government College University Faisalabad (GCUF), Pakistan.

ABSTRACT Agile software development is successful due to self-organizing teams, adaptive planning, a cooperative environment with respect to communication with clients and team members, small development cycles, continuous design improvements, continuous delivery and feedback of clients. Cloud computing helps to reduce cost, enables scalability and enhances communication through its services. A generic framework with the conjunction of Agile Development and Cloud Computing (ADCC) proposed in an earlier study is evaluated in this study. The Malaysia Research and education network (MyRen) cloud is utilized to implement the framework. A case study is conducted to evaluate the framework. Before conducting the case study, the participants are educated on the ADCC framework. The results of the case study show that the performance of agile methods is improved with the usage of the ADCC framework. The improvement is measured in terms of local and distributed agile development environments.

INDEX TERMS Agile development, case study, cloud-based agile tools, cloud computing.

I. INTRODUCTION

Development method and environment plays a vital role in the success of software development. To meet the fast and ever-changing needs of users and industry, software development processes have evolved with the passage of time. The adoption of new technologies (such as the advent of big data, cloud computing and IOTs) is another reason for the evolution of software development processes. These stressing factors divert software development towards parallel processing, iterative development cycles, prioritizing tasks [1]–[3], accelerated productivity, increasing quality while lowering costs [4]–[6] and performing collaborative tasks to achieve a faster paced development [7], [8].

Agile software development teams work efficiently, think differently, learn from previous work and experienced fellows. The aim of agile software development is encapsulated into twelve agile principles [10]. The key features of agile principles are customer satisfaction, face to face communication, close interaction with user, accommodation of changes in requirements, iterative improvement in design, sustaining simple and good design, frequent delivery of software, self-organizing teams and maintaining pace of development.

Although agile principles elaborate the importance of agility in software development, however, the guidelines described in agile principles are not practiced widely, due to changing market demands and distributed development environment. The challenges faced in agile development are scalability, transparency [1], face-to-face communication [11], availability of experts [3], smooth control of development, ability to build applications from distributed locations [8], [9] and resource management [1], [7]. The changing demands require an environment to test innovative ideas. The provision of resources for testing innovative ideas increases the development cost.

Only agile models or processes alone are not sufficient to solve these challenges. New helpers are explored such as cloud computing. Cloud computing has enough potential to

The associate editor coordinating the review of this manuscript and approving it for publication was Xiaobing Sun.

reduce the cost of software development by providing software infrastructure (software and hardware) on a pay per use basis [8], [15], [16]. Cloud computing reduces the overhead of new installations, re-installations, and updating patches [14]. The time delays in software development are reduced due to provision of on demand test servers. The risks in agile software development are reduced due to visualization and transparency. On time provision of cloud services enable the testing of new ideas produced in the marketplace with zero delay [12], [13].

The unification of agile and cloud is beneficial in distributed agile development by using visualization (creating virtual machines), data sharing, prioritizing task, transparency and provision of infrastructure [1], [5], [8], [17]. Furthermore, cloud-based services help in project management, issue management and improve build cycle by automated testing, continuous integration / delivery. It turns agile development into true parallel activities. Cloud computing honors the ecosystem of agile software development with increasing prominence [3], [18], [19]. Ecosystem refers to a system or a group of interconnected elements such as a development environment, team inter connectivity and a complete system working smoothly and quickly. Cloud computing helps in agile requirement engineering [18], [34], [39], development [13], [19], [35], testing [34] and deployment of software [19], [34], [36], and reduces development time. A study [17] predicts the variables which can affect the adoption of cloud computing in agile software development. Another study [37] highlights the risk analysis and management for agile software development due to cloud computing and iot.

Agile development is famous due to short development cycles, accommodation of changes at any stage of development, frequent delivery of software and strong user interaction in the development cycle. In order to implement these features of agile development, cloud computing has an important role. Our previous study proposed a framework for Agile Development in Cloud Computing environment (ADCC) [20]. A systematic literature review in the domain of agile development and cloud computing [21] highlights the lack of evaluation studies in this area. Therefore, this study evaluates the ADCC framework through a case study.

This paper is divided into the following sections. Section II describes related studies. Section III explains the research methodology and experimental setup. Section IV presents the case study and the artifacts used in the case study. Section V describes the results of the case study. In Section 6, the conclusion of case study is presented. Finally, Section 7 describes the threats to validity.

II. RELATED WORK

In order to highlight the problem background, a Systematic Literature Review (SLR) study was conducted [21]. After a search on all known research resources, the primary studies were categorized according to Wieringa Maiden [22] classification as shown in Fig.1.

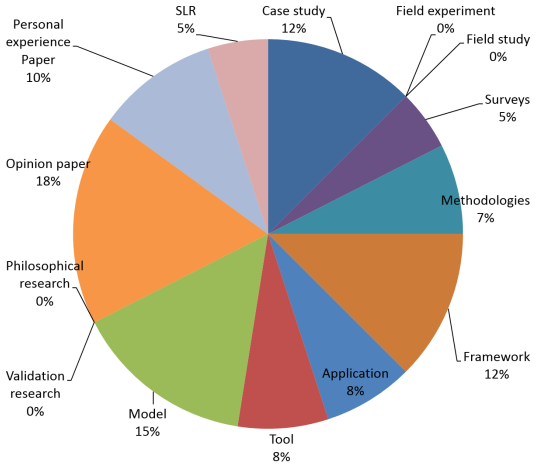

FIGURE 1. Distribution of primary studies research contributions adopted by a study [21].

Fig. 1 shows that the contributors in primary studies are methodologies (7%), frameworks (12%), an application developed (8%), tools (8%) and the model proposed (15%). The contribution of studies with respect to research type are case study (12%), and survey paper (5%). The findings of the SLR show that there is no contribution to the field study. Field experiment and applications are 8%. A few studies focus on validation research and philosophical research. There is a need for evaluation research through real-world case studies and experience reports in this area.

The pace of development becomes very fast due to rapid change in market behavior and business. In order to fulfill the market demand, there is a need for the helping tools for agile development [9]. Cloud computing services helps agile development by facilitating a large range of tools for managing software development activities such as sprint management, velocity metrics, budgeting, analytics, time tracking and reporting, resource management, Kanban board, and customer feedback. The examples for such tools are AgileFant, CloudForge, and Jira, which are used primarily for agile project management. The source code repositories include examples such as BitBucket [24], GitHub [11], [18], [19], [25], CodeSpace, GoogleCode, SourceForge, and Unfuddle. Code repositories help in managing source codes and their versions. For testing of software and infrastructure, TestInfra tool is used for network infrastructure testing. Furthermore, JUnit for unit testing, Jmeter and Cassandra tool for performance testing and lynis tool for security test [26].

Communication among team members has an important role in agile software development. The requirements are refined and firmed with customer collaboration. In distributed agile software development, the role of the communication medium is increased. There are a number of new collaboration methods used for social interaction such as discussion forums, wikis and real-time reports [27]. Furthermore, Skype is used for scrum meetings [5], [6]. For other agile development activities, AWS-EC2 instance provides databases for

TABLE 1. Primary studies using existing tools for their solution adopted by [18].

Study Reference	Team Collaboration	Agile / Cloud Support	Project Management	Code Repository / Development IDE	Agile Model	Evaluation
[15]	Confluences	Chef / puppet	Cloud Spoke on Top coder	GitHub, Cloud IDE	-	Pilot Project
[3]	-	mongoDB	-	Ruby on Rails	Agile	-
[13]	Skype	Cloud Forge	-	Eclipse	-	case study
[22]	Skype	Google App Engine	-	Eclipse	DSDM	case study warehouse
[25]	Cloud Telephony	EC2-AWS	agileFant	-	Agile	migration from agile to cloud
[24]	Discussion forum, wikis, instant reports	Team Forge, Cloud Forge	Team Forge	Eclipse, Visual Studio	Agile	-
[28]	-	Google App Engine	-	Eclipse IDE	-	Survey

sharing data [24], [28], Jenkins, XUnit, and Junit provide testing environments and Puppet, Chef, Jclouds, and Whirr provide platforms for software deployment [29]. The studies using existing tools for agile development activities are listed in Table 1.

The studies listed in Table 1 present the solutions based on a concept of re-usability. The re-usability of tools enhances agile software development; however, the development teams face problems in adjusting to new and unfamiliar environments and sometimes configuration management between different artifacts also becomes complicated.

III. RESEARCH METHODOLOGY AND EXPERIMENTAL SETUP

This Section explains the research methodology and experimental setup for evaluating the ADCC framework, the set of requirements used in the experiment, the overview of the teams participating in the experiment and the evaluation criteria. The research methodology explains the conceptual view of the ADCC framework and its execution steps. The execution of the ADCC framework generates a different set of possible ADCC environments. One of these possible environments is evaluated in this study. Furthermore, the methodology explains the pilot test project used for the evaluation of the ADCC environment. The details are as follows:

A. EXECUTION OF THE ADCC FRAMEWORK

The ADCC framework [20], [23] uses existing techniques and tools for establishing environments for agile development in cloud computing, which promote re-usability of existing solutions. Fig.2 elaborates the conceptual design of the ADCC framework. The cloud provider helps in cloud services for managing agile development activities. The workflow of different roles such as developers, testers and users is depicted in Fig.2. The red, green and blue connecting lines represent the role and scope of the developers, testers, project managers and users.

In Fig.2, the first part of ADCC framework is agile software project management as depicted by red lines.

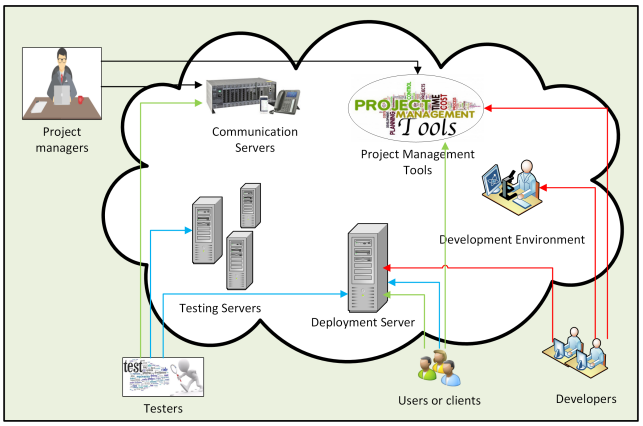

FIGURE 2. Conceptual design of framework.

The second component is the selection of cloud platforms such as development, testing and deployment servers represented by blue lines in Fig.2. The third part of the ADCC framework consists of communication and collaboration medium such as email, Skype and Google drive, as represented by green lines in Fig.2; and the fourth component refers to code management tools such as GitHub. The execution steps for the ADCC framework are explained below in Fig 3.

In order to build an ADCC environment, the study adopts the procedure defined in Fig.3, the details of which are as follows.

Fig 3 has four steps to implements the ADCC framework. First step is the selection of agile tools, which is based on agile methodology and features used by the development team. The second step is the selection of the cloud computing platform, which is based on the project budget, size, security and nature. For managing code and versioning in an agile distributed environment, the code repository is selected in step three. In step 4, the selection of communication tools is done. For selecting all these tools in four steps, there is a need to check compatibility among the tools, which is

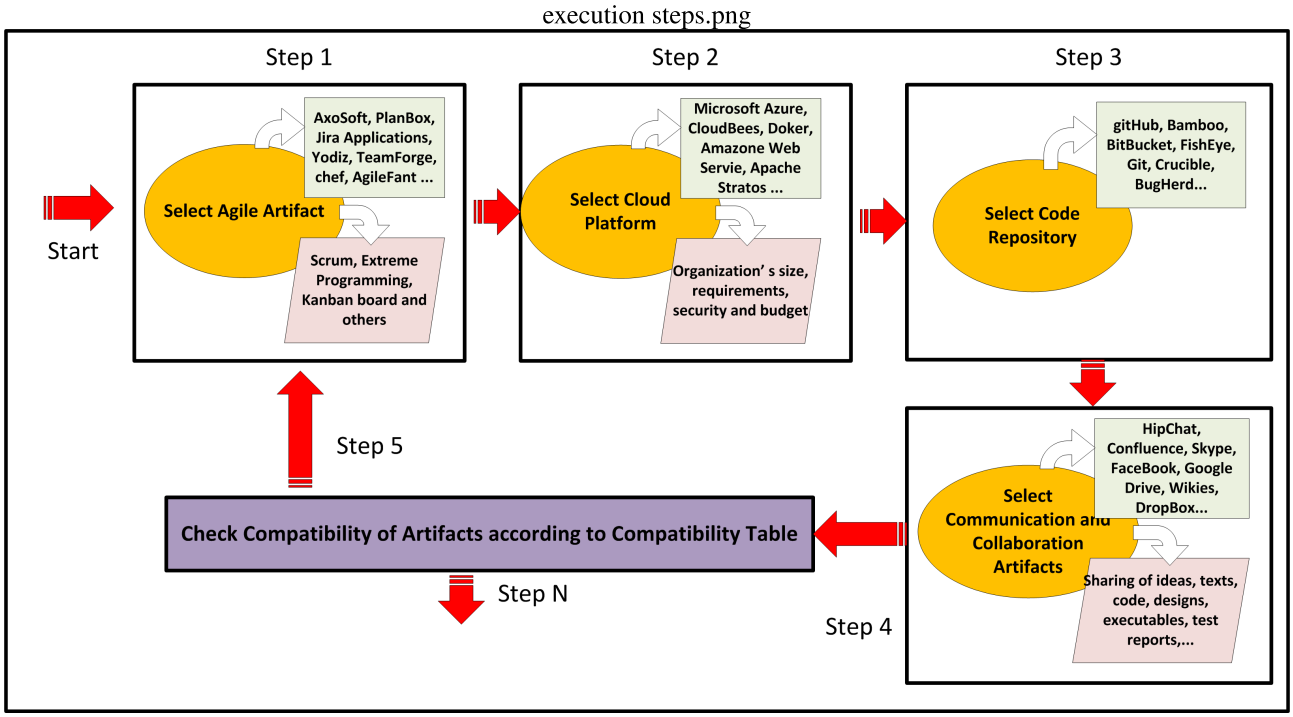

FIGURE 3. Implementation procedure for conceptual design of framework.

performed in step five. The comparability table in Study [20] helps in resolving the compatibility issues. In case of an unsuccessful selection, steps one to four are repeated until all the compatibility issues are resolved.

The details of finalized artifacts for the ADCC environment selected in the previous paragraph, and their configurations, are described in Fig 4. CloudAgility tool, an enhanced version of SAgile tool [31] is used for agile project management in this case study. This tool is configured in MyRen 1 virtual machine as shown in Fig.4. Selenium IDE, which is configured on Firefox browser, is used for testing the agile project. It is a simple playback tool for testing without the need to learn scripting language.

Malaysia Research and education network (MyRen) cloud [32] is used for establishing the environment for agile development. MyRen provides virtual machines to the software engineers through cloud admins. The clones of virtual machines can be created and when needed. The project management tool CloudAgility is configured on separate virtual machine. The test server and the development environment is also configured on a separate machine as shown in Fig.4.

In the case study, eclipse IDE is used as the development environment. It is configured with GitHub to manage code in a distributed environment. The same virtual machine is used as the deployment server. and shared with the customers and other members. For sharing of code, design concepts and views email, Skype and google drive are used. The ADCC environment is evaluated by developing a pilot project of a hospital management system. The description of the pilot project in terms of requirement sentences is as follows:

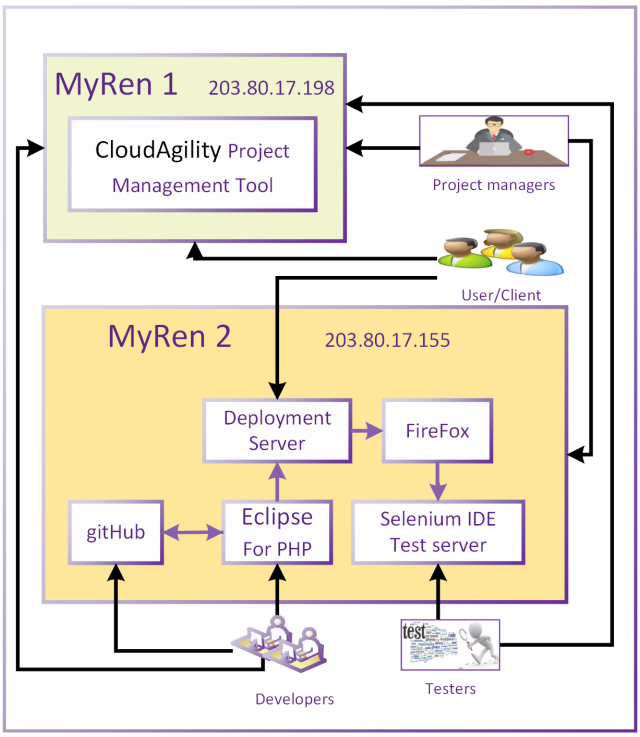

FIGURE 4. Experimental setup for implementation of framework.

B. OVERVIEW OF HOSPITAL MANAGEMENT SYSTEM

The Hospital Management System (HMS) is designed to replace an existing manual, paper-based system. The HMS is used to manage

- administrative affairs

- patient information
- patient administration
- room availability
- staff and operating room schedules
- patient invoices and billing

The HMS software must be flexible, reliable, and easy to adapt. A significant part of the operation of HMS is acquisition, management and timely retrieval of great volumes of information. This information typically involves

- patient personal information and medical history
- staff information
- room and ward scheduling
- staff scheduling
- operation theater scheduling and waiting lists for various facilities

The HMS system generates different reports to utilize resources effectively while keeping the integrity of data. All the information is managed in an efficient and cost effective manner. HMS will automate the management of the hospital, making it more efficient and error free. It aims at standardizing data, consolidating data, ensuring data integrity and reducing inconsistencies.

C. TEAM'S OVERVIEW

In this case study, four teams participate in four different scenarios. The scenarios are with respect to the selection of environment (such as simple agile or ADCC) and mode of environment (such as distributed or local) as shown in Figure 5. Two development teams consisting of four students of a class of Bachelor of Computer Science who are attending the course Software Engineering II in the fourth year of the program at the computer science department in Government College University Faisalabad (GCUF), Pakistan participate in the case study. Each team consists of one female and three male students having a CGPA greater than 3.0 out of 4.0. The competency of each team for both scenarios is same. All students can do the jobs of designing, coding and testing. The instructor of the class and his assistant performs the role of product owner and end user. In this case study, there is a role of product owner and end user. They help team members in understanding and clarification of the requirements and minimize the ambiguities. The secondary purpose of the product owner and end user is to complete the agile development cycle. Furthermore, two development teams comprising of four students of class Bachelor of Science in Computer Science (Software Engineering), attending the course Application Development in the third year of the program in Universiti Teknologi Malaysia (UTM), Johor, Malaysia, also participate in the case study. Each team consists of three female students and one male student, having a CGPA greater than 3.0. The competency of each team for both scenarios is the same. All students can do the jobs of designing, coding and testing. The Two Ph.D. students perform the role of product owner and end user. The purpose of two case studies is to balance the impact of participants. The average result of both case studies is compiled.

In scenario 1, simple agile methodology is adopted by teams from GCUF and UTM working locally and independently. In scenario 2, both teams from GCUF and UTM use the ADCC environment to work locally for the case study. In scenario 3 and scenario 4, the mode is distributed environment, and each team comprises of two members from GCUF and two members from UTM. Scenario 3 uses simple agile development and scenario 4 uses ADCC environment.

TABLE 2. HMS release plan and story points.

Modules / releases	User stories	Story Points
Reception	Doctor visit schedule	3
	Doctor Appointment Scheduling	2
	Inquiry of Patient	1
	Find History of Patient Inquired	1
Administration	Employee Detail Recording.	2
	Doctor Type	1
	Doctor Master	2
	Referral Doctor	1
Pharmacy	Receipt and bills	2
	Stock keeping	3
	Stock return	1
	Stock maintenance	3
Laboratory	Patient test	1
	Wards billing	3
	Test billing	2
	Sample collection	2
	Available services	1
	In/out patient registration	2
Registration	Patient registration	1
	Patient history	2
	IPD/OPD management	1
Discharge Summary	Discharge report	3
	Medical history	2
	Diagnose history	4
	Drug prescription	2
	Present illness	2

D. DESCRIPTION OF PILOT TEST PROJECT USED IN CASE STUDY

In this case study, a software "Hospital Management System" (HMS), is developed by the teams. Scrum methodology is adopted for the completion of the HMS application. The application is completed in 6 releases. The details of the user stories and work breakdown structure of the HMS project is given in Table 2. The project is distributed in user stories by performing the agile activities such as defining vision, ranking the backlog, having the release planning meeting and final release. The HMS application is developed two times with different teams and work environments. In Table 2, the story points show the amount of effort required to complete the new story. This is a relevant point value which also has been compared with the points of other stories which the team has completed.

IV. CASE STUDY FOR EVALUATION OF ADCC FRAMEWORK

For evaluation, the case study comprises of four scenarios with respect to agile software development (such as using

TABLE 3. Performance of teams in local / on-premise environment on using agile development and ADCC.

Development Phases	No. of Days					
	Team 1-UTM		Team 2-GCUF		Average Days	
	Using Agile Development	Using ADCC	Using Agile Development	Using ADCC	Using Agile Development	Using ADCC
Requirement elicitation	1.5	1	2.5	2	2	1.5
Planning, design and coding	33	30	39	36	36	33
Testing and deployment	7.5	6	8.5	7	8	6.5

TABLE 4. Performance of team on distributed environment with using agile development and ADCC.

Development Phases	No. of Days					
	Team 1-UTM-GCUF		Team 2-UTM-GCUF		Average Days	
	Using Agile Development	Using ADCC	Using Agile Development	Using ADCC	Using Agile Development	Using ADCC
Requirement elicitation	3	1.5	3	1.5	3	1.5
Planning, design and coding	37	31	43	35	40	33
Testing and deployment	10	7	8	6	9	6.5

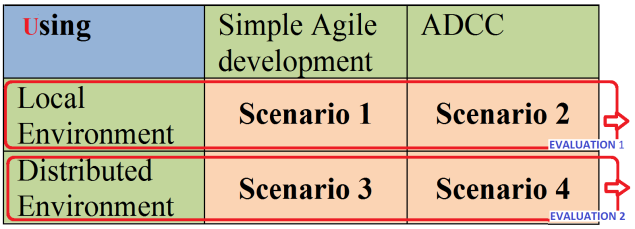

FIGURE 5. Case study scenarios.

ADCC and without using ADCC) and the development environment (such as local and distributed work environment) as shown in Figure 5. The description of each team participating in four scenarios is described in Section III-C. In all the scenarios, an application of Hospital Management System (HMS) is developed. The description of HMS is given in Section III-B. The application is completed in 6 releases. In the first evaluation, the performance of software development by usage of simple agile and ADCC environment in local / on-premise is compared. The scenario 1 and 2 is used in the first evaluation. In the second evaluation, the performance of software development by usage of simple agile and ADCC environment in distributed locations is compared. The scenario 3 and 4 is used in second evaluation.

A. APPLICATION DEVELOPMENT SETUP USING SIMPLE AGILE METHODS

The team developed a web-based application using PHP and MySQL. They installed NetBeans IDE and configured it for Apache HTTP server, PHP engine, MySQL database server, and then sets up the environment. Each team member installed all the software on their work consoles. The team worked together to design phases for solving the problems in all iterations. After completing each iteration, they shared it with the instructor. After completing iterations, they integrated all the iterations to complete the deliverable. They completed six releases in the same way. The results in the form of completion time for each phase are collectively counted for all deliverables. The evaluation is measured in terms of number of days taken to complete three software development phases i.e. i) requirement elicitation, ii) planning, design and coding and iii) testing and deployment.

B. APPLICATION DEVELOPMENT SETUP USING ADCC FRAMEWORK

In the second and fourth scenario, the same application is developed using the ADCC framework which is organized by the Malaysia Research and education network (MyRen) cloud. The practical setup of ADCC framework uses CloudAgility tool for managing project activities. Eclipse for PHP is used as a development IDE in this experiment. GitHub is used for code management and versioning. For team communication, Skype and email are used. The provision of cloud virtual machine is done through the cloud admin. For testing of the software, the Selenium test server is used. The experimental environment is described earlier in Section III-A. The performance is measured in terms of number of days taken to complete three software development phases.There are two evaluations: first in local development environment, where the comparison is of scenario 1 to scenario 2, in terms of number of days taken to complete three phases. The comparison of both scenarios evaluates the difference of only using an agile development process and ADCC framework, which is an agile development process using a cloud computing environment. The second evaluation is in a distributed environment, where the comparison of

scenario 3 with scenario 4 in terms of number of days is taken to complete the three phases.

During the application development, the developers design the prototype, test and revise it. All the activities are performed in parallel through cloud virtual machines. The selenium test server is configured with a development environment as described in previous Section III-A. On completion of the first iteration, it is deployed on the deployment server, and the instructor (product owner) and the end user can access it, validate and can add new requirements if needed. All iterations are completed in the same way and on completion of all releases, the module is integrated. The complete solution is submitted to the instructor. The results in the form of completion time for each phase collectively for all deliverable are counted. The same task is performed by the different teams, who perform the application development on the different development environment.

V. RESULTS OF CASE STUDY

The case study has two sub parts evaluation 1 and evaluation 2. In evaluation 1, the results in scenario 1 are compared with results in scenario 2. Three phases of software development are compared in scenario 1 and 2, such as using simple agile environment and using ADCC environment work locally / on-premise. The phases used for measuring performance are i) requirement elicitation, ii) planning design and coding and iii) testing and deployment. The scenario is in terms of using only agile software development and using ADCC framework. The work is measured in terms of the total number of days used in the completion of the artifact. In evaluation 2, the case study measures the performance of scenario 3 and scenario 4 in a distributed development environment.

First, the performance of teams working in local (on-premise) is measured. A team from UTM and GCUF worked separately and independently in the local environment. The results of a team from GCUF and a team from UTM in terms of number of days taken to complete the development phases are given in Table 3. The competency of each team for both scenarios is nearly equal. Furthermore, to moderate the competency effect, the average of both teams is calculated. All students can do the jobs of designing, coding and testing. The instructor of the class and his assistant performs the role of product owner and end user (if needed). The average of all teams with respect to three development phases is calculated. The difference in terms of a number of days spent in completing agile development activities in both scenarios is shown in Fig.6. In a local environment, Figure 6 shows that development time is efficient when using ADCC as compared to a simple agile environment.

Evaluation 2 is using a distributed environment, the mix of students of UTM and GCUF work as one team and develop the HMS software on the simple agile development environment (scenario 3) and then the same software is developed in an ADCC environment (scenario 4). The performance of scenario 3 and scenario 4 in terms of number of days taken to complete three development phases is given in Table 4.

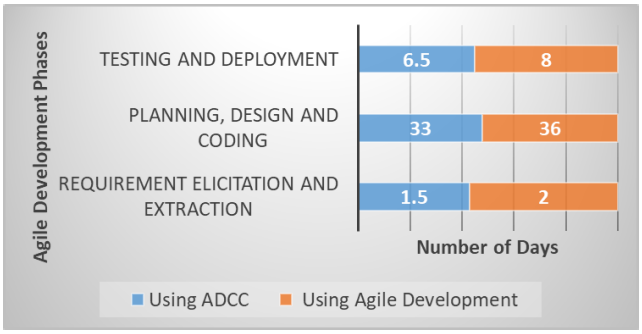

FIGURE 6. Comparison of agile phases with and without using ADCC in a local development environment.

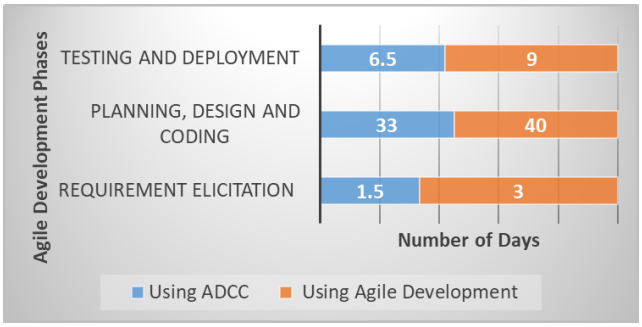

FIGURE 7. Comparison of agile phases with and without using ADCC in a distributed development environment.

In order to minimize the competency effects of teams the average of two composite teams is calculated in Table 4. Figure 7 shows that, using the ADCC environment results in better performance as compared to a simple agile environment. The reason for the improvement seen in the ADCC environment is due to zero delay in communication of teams and users, the provision of hardware and computing resource without worrying about configurations of patches, updates, and installations. The ADCC environment provides the testing and deployment environment with zero delays.

In a distributed environment, teams are located at distant places logistically. There is communication delay due to cultures, languages and time zones difference [38]. The case study forms the teams with a mix of UTM and GCUF students and has them develop the HMS software in both scenarios. Table 4 shows the number of days spent by the teams using both scenarios in different development phases.

In a distributed environment, a simple agile software development process can be delayed due to communication delay between development teams and other stakeholders, ultimately affecting the requirement gathering process. On the other hand, The ADCC framework provides communication and collaboration facility, which helps in requirement gathering and planning phases. As such, the elicitation process using only agile software development is completed in three days and in one and half days by using ADCC framework as shown in Fig.7.

In an agile development, after completing one iteration, it is sent to the product owner to get feedback and it is noted that

all this process is manual or not inter-connected. On the other hand, the ADCC provides an infrastructure for a development environment in a distributed environment. Due to parallelism, the performance of planning design and coding increases by using ADCC. The code versioning facility helps the distant team members. Plans and design artifacts are shared easily among team members. Furthermore, the planning, design and coding activities are completed in 40 days and by using ADCC, completed in 33 days as shown in Fig.7.

In an agile development, users test the code and share with another member manually through email or flash drive. And to show the output they make arrangement to get feedback of user and product owner. On the contrary, the testing and deployment servers are facilitated by ADCC, and become a cause to reduce the completion time. There is no waste of time in getting to a test environment. The scalability issue is resolved due to the use of ADCC. Each team member can view the progress of software development by accessing and viewing the deployment server facilitated by the ADCC framework. By using the agile method, the testing and deployment activities are completed in 9 days and by using ADCC, they are completed in six and half days as shown in Fig.7.

In agile methods the requirements are confirmed with the user interaction until the end of the project; however, due to communication and collaboration difficulties, the requirement elicitation process gets delayed and job completion time has increased as shown in Fig.8. In a distributed environment, there are difficulties in sharing design, concepts, code and prototypes among team members and clients. Due to communication and collaboration delays, the completion time of design and coding is increased as shown in Fig.8. Furthermore, the sharing of prototypes and progress of development by means of deliverable to the client and test reports face difficulties in a distributed environment and ultimately increase the job completion time. In a distributed environment the latency increases in software development phases. Fig.8 shows the increase in time in different phases of agile development due to the distributed environment.

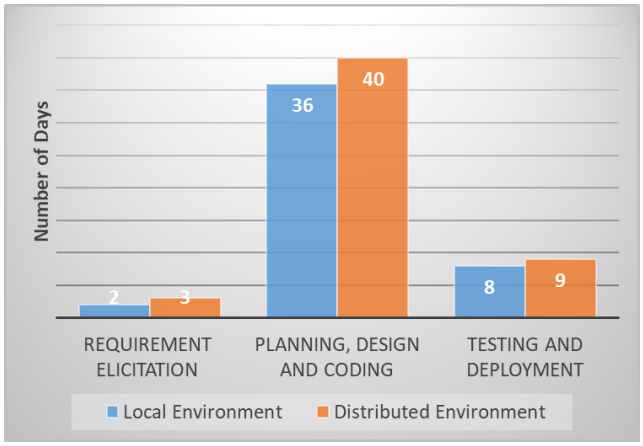

FIGURE 8. Impact of local and distributed environment on agile development.

VI. CONCLUSION

The ADCC framework is evaluated with the help of a case study. The purpose of this case study is to frame the problems faced during agile software development and how these problems are solved using the ADCC framework. During an agile software development process, the team has resources, but these are not interconnected like in real software development. If dedicated resources are arranged, then it becomes expansive, an extra network staff is required to maintain the system. On the other hand, the ADCC framework (cloud computing) provides an environment which is interconnected. All the developers and other team members feel that they are working in the on-premise environment. All systems behave like a single system. The difference in the number of days consumed to perform software development activities show the benefits of ADCC framework over a simple agile development environment. In general, the ADCC framework has the following findings based on the results of the case study.

- The ADCC framework reduces communication latency in agile development.
- The ADCC framework describes the range of tools to support agile development activities in context of cloud computing and their compatibility.
- The use of the ADCC framework produces a more structured and managed environment for agile development by providing development infrastructure, on-demand testing servers and continuous deployment.
- The ADCC framework is provided on demand to any number of test servers which have enabled software development activities.

The ADCC framework is a generic solution for agile development in cloud computing environment, however as a future aspect, it is applicable on other software engineering methods by selecting collection of tools and environments.

VII. THREATS TO VALIDITY

A case study is conducted to evaluate the ADCC framework. The framework possesses different choices for selection of platforms and tools. The case study has the following threats to validity. A performance bias may occur when same project is developed by different teams with and without the ADCC framework. There is a chance of different competency level of teams. In the case study, the teams are educated about the experimental setup of the ADCC, so that the attitude during educating the environment may influence the results. In the case of a distributed environment, there is no control on the communication of the team members from the same class/university. They can communicate directly, which is a threat. Another limitation is that this study has not been compared with existing studies. A study [6] conducted a case study of a warehouse management application using DSDM model. They compared the performance of agile software development using agile and using agile-cloud. Furthermore,

they did not share the requirements of the developed application. Another study [2] built a ShaMoCloud application and compared it with the DSDM application, by assuming that their ShaMoCloud application is nearly equal to the number of requirements in the DSDM application. Furthermore, both applications did not share their requirements used in the case study, which is a biasness. Under these limitations, the current study does not compares with the existing studies due to unavailability of requirements used in case studies for application development.

ACKNOWLEDGMENT

The authors would like to thank Universiti Teknologi Malaysia (UTM) for Trans disciplinary Research Grant 06G23 and the Government College University Faisalabad (GCUF), Pakistan, for the students participating in case studies, and the Malaysia Research and education network (MyRen) cloud, in to provide resources for this research.

REFERENCES

[1] A. Tuli, N. Hasteer, M. Sharma, and A. Bansal, "Empirical investigation of agile software development: Cloud perspective," *SIGSOFT Softw. Eng. Notes*, vol. 39, no. 4, pp. 1–6, 2014.

[2] F. Almudarra and B. Qureshi, "Issues in adopting agile development principles for mobile cloud computing applications," *Procedia Comput. Sci.*, vol. 52, pp. 1133–1140, 2015.

[3] A. Nazir, A. Raana, and M. F. Khan, "Cloud computing ensembles agile development methodologies for successful project development," *Int. J. Mod. Educ. Comput. Sci.*, vol. 11, pp. 28–35, Nov. 2013.

[4] N. Jain and S. Dubey, "Agile development methodology with cloud computing," *Int. J. Eng. Comput. Sci.*, vol. 3, no. 4, pp. 5373–5378, 2014.

[5] I. Inayat, S. S. Salim, and Z. M. Kasirun, "Agile-based software product development using cloud computing services: Findings from a case study," *Sci. Int. (Lahore)*, pp. 1065–1069, 2013.

[6] S. Kalem, D. Donko, and D. Boskovic, "Agile methods for cloud computing," in *Proc. 36th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2013, pp. 1079–1083.

[7] A. Dumbre, S. P. Senthil, and S. S. Ghag, "Practising agile software development on the windows azure platform," Infosys, Bengaluru, India, White Paper, 2011.

[8] W. Wang. (2011). *Reinforcing Agile Software Development in the Cloud*. Accessed: Jan. 30, 2016. [Online]. Available: https://www.opencollab.net/media/pdfs/CollabNet%20Whitepaper_Reinforcing%20Agile%20Dev%20in%20the%20Cloud.pdf?_=d

[9] A. Sever, "Modeling distributed agile software development utilizing cloud computing: A holistic framework," *Current J. Appl. Sci. Technol.*, vol. 35, no. 6, pp. 1–12, 2019.

[10] A. Alliance. (Feb. 25, 2001). *Agile Manifesto*. [Online]. Available: http://www.agilemanifesto.org

[11] M. R. J. Qureshi and I. Sayid, "Scheme of global scrum management software," *Int. J. Inf. Eng. Electron. Bus.*, pp. 1–7, Mar. 2015.

[12] R. Shriver. (2012). *Agile Cloud Development: The Future of Software*. Accessed: Feb. 20, 2016. [Online]. Available: http://www.virtualizationpractice.com/agile-cloud-development-the-future-of-software-16226/

[13] N. Rathod and A. Surve, "Test orchestration a framework for continuous integration and continuous deployment," in *Proc. Int. Conf. Pervas. Comput. (ICPC)*, Jan. 2015, pp. 1–5.

[14] B. Portelli. (2010). *The Beauty of Agile in The Cloud*. Accessed: Sep. 2017. Available: [Online]. Available: https://www.agileconnection.com/article/beauty-agile-cloud

[15] H. Benfenatki, C. F. D. Silva, A.-N. Benharkat, and P. Ghodous, "Cloud–based business applications development methodology," in *Proc. IEEE 23rd Int. WETICE Conf.*, Jun. 2014, pp. 275–280.

[16] T. G. P. Mell. (2011). *The NIST Definition of Cloud Computing*. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

[17] S. A. Butt, M. I. Tariq, T. Jamal, A. Ali, J. L. D. Martinez, and E. De-La-Hoz-Franco, "Predictive variables for agile development merging cloud computing services," *IEEE Access*, vol. 7, pp. 99273–99282, 2019.

[18] T. Haig-Smith and M. Tanner, "Cloud computing as an enabler of agile global software development," *Issues Informing Sci. Inf. Technol.*, vol. 13, pp. 121–144, Mar. 2016.

[19] W.-T. Tsai, W. Wu, and M. N. Huhns, "Cloud–based software crowdsourcing," *IEEE Internet Comput.*, vol. 18, no. 3, pp. 78–83, May 2014.

[20] M. Younas, I. Ghani, D. N. Jawawi, and M. M. Khan, "A framework for agile development in cloud computing environment," *J. Internet Comput. Services*, vol. 17, no. 5, pp. 67–74, 2016.

[21] M. Younas, D. N. Jawawi, I. Ghani, T. Fries, and R. Kazmi, "Agile development in the cloud computing environment: A systematic review," *Inf. Softw. Technol.*, vol. 103, pp. 142–158, Nov. 2018.

[22] R. Wieringa, N. Maiden, N. Mead, and C. Rolland, "Requirements engineering paper classification and evaluation criteria: A proposal and a discussion," *Requirements Eng.*, vol. 11, no. 1, pp. 102–107, Mar. 2006.

[23] M. Younas, D. N. A. Jawawi, I. Ghani, M. A. Shah, M. M. Khurshid, and S. H. H. Madni, "Framework for agile development using cloud computing: A survey," *Arabian J. Sci. Eng.*, vol. 44, no. 11, pp. 8989–9005, 2019.

[24] F. Almudarra and B. Qureshi, "Issues in adopting agile development principles for mobile cloud computing applications," in *Proc. 6th Int. Conf. Ambient Syst., Netw. Technol.*, E. Shakshuki, Ed., vol. 52, 2015, pp. 1133–1140.

[25] S. Franken, S. Kolvenbach, W. Prinz, I. Alvertis, and S. Koussouris, "CloudTeams: Bridging the gap between developers and customers during software development processes," *Procedia Comput. Sci.*, vol. 68, pp. 188–195, 2015.

[26] J. de Castro Martins, A. F. M. Pinto, G. S. Goncalves, R. A. L. Shigemura, W. C. Neto, A. M. da Cunha, and L. A. V. Dias, "Agile testing quadrants on problem-based learning involving agile development, big data, and cloud computing," *Information Technology-New Generations*. Cham, Switzerland: Springer, 2018, pp. 429–441.

[27] S. Singh and I. Chana, "Introducing agility in cloud based software development through ASD," *Int. J. u- e-Service, Sci. Technol.*, vol. 6, no. 5, pp. 191–202, Oct. 2013.

[28] M. Manuja and Manisha, "Moving agile based projects on cloud," in *Proc. IEEE Int. Advance Comput. Conf. (IACC)*, Feb. 2014, pp. 1392–1397.

[29] M. Miglierina, "Application deployment and management in the cloud," in *Proc. 16th Int. Symp. Symbolic Numeric Algorithms Sci. Comput.*, Sep. 2014, pp. 422–428.

[30] M. M. A. Ghosh and W. F. Al Sarraj, "A case study on academic services application using agile methodology for mobile cloud computing," *Int. J. Res. Eng. Sci.*, vol. 4, no. 2, pp. 22–30, 2016.

[31] A. F. Binti Arbain, I. Ghani, and W. M. N. Wan Kadir, "Agile non functional requiremnents (NFR) traceability metamodel," in *Proc. 8th. Malaysian Softw. Eng. Conf. (MySEC)*, Sep. 2014, pp. 228–233.

[32] Myren. (2017). *Malaysian Research & Education Network*. [Online]. Available: https://myren.net.my/

[33] U. Durrani, J. Richardson, J. Lenarcic, and Z. Pita, "Lean traceability solution through SLAM model: A case study of a hybrid delivery team in a hybrid cloud computing environment," in *Proc. 22nd Australas. Softw. Eng. Conf. (ASWEC)*, 2013, pp. 16–19.

[34] S. Karunakaran, "Impact of cloud adoption on agile software development," in *Software Engineering Frameworks for The Cloud Computing Paradigm*, Springer, pp. 213–234.

[35] B. P. Gopularam, C. B. Yogeesha, and P. Periasamy, Highly scalable model for tests execution in cloud environments, in *Proc. 18th Int. Conf. Adv. Comput. Commun. (ADCOM)*, Dec. 2012, pp. 54–58.

[36] V. E. Jyothi and K. N. Rao, "Effective implementation of agile practices," *Int. J. Adv. Comput. Sci. Appl.*, vol. 2, no. 4, pp. 41–48, 2014.

[37] P. Gouthaman and S. Sankaranarayanan, "Agile software risk management architecture for IoT–Fog based systems," in *Proc. Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Dec. 2018, pp. 48–51, doi: 10.1109/icssit.2018.8748457.

[38] V. Lalsing, "People factors in agile software development and project management," *Int. J. Softw. Eng. Appl.*, vol. 3, no. 1, pp. 117–137, Jan. 2012.

[39] M. Younas, D. N. Jawawi, I. Ghani, R. Kazmi, "Non-functional requirements elicitation guideline for agile methods," *J. Telecommun., Electron. Comput. Eng.*, vol. 9, no. 3–4, pp. 137–142, Oct. 2017.

MUHAMMAD YOUNAS received the Ph.D. degree from the Faculty of Engineering, School of Computing, Universiti Teknologi Malaysia (UTM). He is currently working as an Assistant Professor with the Computer Science Department, Government College University Faisalabad, Pakistan. His research interests include software engineering, agile software development, cloud computing, and code clone detection.

DAYANG NORHAYATI ABANG JAWAWI is currently working as an Associate Professor and the Associate Chair (Academic and Student Development) with the Faculty of Engineering, School of Computing, Universiti Teknologi Malaysia (UTM). She is also a member of Software Engineering Research Group (SERG).

AHMAD KAMIL MAHMOOD received the Ph.D. degree. He is currently working as an Associate Professor with Universiti Teknologi Petronas, Sri Iskandar, Malaysia.

MOHAMMAD NAZIR AHMAD is currently working as an Associate Professor with the Institute of Visual Informatics, Universiti Kebangsaan Malaysia, Bangi, Malaysia.

MUHAMMAD UMER SARWAR is currently working as an Assistant Professor with the Computer Science Department, Government College University Faisalabad, Pakistan.

MOHD YAZID IDRIS is currently working as a Senior Lecturer with the Department of Software Engineering, Faculty of Computing, Universiti Teknologi Malaysia.