I.J. Modern Education and Computer Science, **2019, 5, 24-33**
Published Online May 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2019.05.03

Lean Agile Integration for the Development of Large Size Projects

Rizwan J. Qureshi, Madini O. Alassafi
Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
Email: {rmuhammd,malasafi}@kau.edu.sa

Hafiz M. Shahzad
Faculty of Computer Science & Information Technology Department, Superior University, Lahore, Pakistan
Email: anriz123my@yahoo.com

Received: 03 February 2019; Accepted: 21 March 2019; Published: 08 May 2019

Abstract—Scrum is a well-known agile model due to its strong management practices. It can be mingled with many software development models such as extreme programming (XP), Agile Unified Process, and Feature Driven Development (FDD). Lean is a very popular known process in the automobile industry due to its effective practices such as Kanban bard and smooth workflow. Lean development is gaining popularity in the software industry from the last few years. Lean development is rational, convenient, responsive, and team-based and it adds value to the enterprise. Scrum is useable and practical for small and medium projects but it does not render positive support for the large size projects. In order to adopt Scrum for large size projects, there is a need to integrate Lean and Scrum. It is required to inherit some properties of the Lean into Scrum, without compromising the speed, quality, efficiency, and standards, to accomplish large size projects successfully such as enterprise resource planning (ERP) systems. It is anticipated that the proposed Lean Scrum integration will make it suitable to develop large size projects. The same is accomplished by purposing an integrated LScrum model in this research. The proposed model is validated using a survey to conclude the results. The results of the survey support the proposed integration of Lean and Scrum for the development of large size projects.

Index Terms—Enterprise Resource Planning (ERP) system, Scrum, Lean, Kanban board, Work in Progress (WIP), Quality.

I. Introduction

There are several traditional software development methodologies e.g. Waterfall, Prototyping, Rapid Application Development, Spiral, and Incremental. There are two types of software development methodologies i.e., descriptive and prescriptive. Descriptive is used as the basis for understanding and improving software development processes. Prescriptive, on the other hand, provides guidelines to develop software and it requires extensive documentation. Descriptive and prescriptive methodologies are used to organize, plan, estimate cost, estimate schedule and managing the software projects [1,2].

The most popular addition to software development methodologies is agile. The agile methodologies are lightweight in nature allowing a team to develop software in the rapidly changing requirements. The agile models are based on such principles that force and allow the development team to build software in a short span of time. The agile models are only suitable for small and medium scale projects [3]. Agile methodologies focus mainly to handle the changes during software development such as extreme programming (XP) and Scrum. XP defines change as the norm and stability in an abnormal situation [4]. Scrum is a lightweight methodology that is composed of interrelated practices and set of laws to optimize the development environment and reduce organizational overheads. Scrum is just like the eight members of the Rugby team, they all are packed together and work as a team. A team has to achieve only one goal at a single interval of time [5].

Scrum delivers quick releases through sprints. The development focused on sprints and brief meetings. The meetings are held on a daily basis to make sure that the teams are working on the right track to accomplish the project successfully. At each sprint, the functionality of the components should be vividly clear to meet the deadlines and deliver each component on time [6]. Lean thinking is based on the concepts of the Toyota Production System that is introduced by Taiichi Ohno in the mid-1940s. Eliminate waste is an important property of lean. If the goals are achieved by eliminating any activity, that activity is wasted. The extra processes and modules are totally wasted in the project [7].

In order to successfully accomplish new large scale projects in Scrum, we have to pick some properties (Eliminate Waste, Build Quality, Create Knowledge, deliver fast, optimize the whole) of the lean development so that the period required for the completion of the project should not differ with the main properties of the Scrum. The techniques of both the processes Scrum and lean development should be judiciously interlinked with each other in order to achieve the middle way approach to cope with the problems faced by the new large scale projects Lean Software Development is a translation of lean manufacturing principles and practices to the software development domain. Adapted from the Toyota Production System, a pro-lean subculture is emerging within the agile community [8]. This research is conducted to integrate Lean and Scrum approaches to enhance the strengths of both approaches and eliminate their limitations with the aim of implementing it for the development of large size software projects.

Further paper is arranged as: Section 2 focuses on the related work. Section 3 defines the problem in hand. Section 4 proposes an integrated LScrum model. Section 5 describes outcomes of this research.

II. Related Work

The adoption of Scrum process model in Yahoo Company is discussed [9]. In 2002, the Yahoo Company used Waterfall as a process model and it was called as "Product development Process (PDP)" but their team members ignored that process model because the Waterfall processes slowed down their project. The Yahoo Company started working on agile in 2005. Twenty-five teams worked on Scrum comprising about 84 percent of team members. They supported Scrum as compared to the PDP process model. Benefield [9] also discusses the problems in Scrum while adoption. It is mentioned that where the teams are wrong and how to cope with the problems in Scrum. Yahoo Company has more than 150 teams working on Scrum now. There are few teams still using PDP and the management solved problems of two different types of teams using PDP and Scrum. Yahoo Company is still facing it difficult to adopt Scrum overall teams [9].

In agile software development, people have diversified views about agile i.e., Scrum is not suitable for software development and it is the most suitable model to integrate with other methodologies [10]. Like all other process models, the Scrum is not exempted from the problems and challenges. In fact, it is suitable for small projects but not for the large scale projects. For large scale projects, Scrum is not supportive of the unique nature of ERP systems. Agile development methods are suitable for small teams, but for larger projects, other processes are more appropriate e.g., Waterfall and RUP [11].

Perfection does not mean that you cannot add more features in a product but it means that you cannot cut even a single feature from it [12]. It is a common observation in software engineering that majority of features do not increase the value of software but they only increase complexity and overhead cost of development such as failure of Netscape browser, Nokia's Symbian 60 and Microsoft Vista. According to Ebert et al. [12], thirty to fifty percent of the functionalities in software are excessive. These extra functionalities are responsible to increase the complexity and cost of software. The time, size and cost of software development are increasing enormously from last several years such as ERP systems. There is a firm need to decrease the cost of development to increase the affordability of customers. This factor has become the driving force to apply Lean principles in software development as well. Many studies show that lean principles are quite effective and efficient to cut down the cost of executing a system development life cycle (SDLC) phases by removing waste and increasing value. According to a survey in 2010, 35% of software development companies describe that they are integrating the agile methodologies and lean principles [12].

There are several ASD methodologies such as XP, Scrum, Feature Driven Development (FDD), Crystal, Dynamic System Development Method (DSDM) and Agile Unified Process (AUP). ASD methodologies require different tactics and strategies to integrate with lean. It is required to identify the core features of each ASD methodology to effectively integrate with the Lean. Rodriguez et al. [13] investigate the features to assist the software industry in integrating agile and lean. The area of research is selected because the software industry has shown great interest in the last few years to accept and apply lean and ASD together due to their several potential benefits such as time-saving, flexibility and cost-effectiveness. There is limited literature available about the effective integration of lean and ASD. Therefore, there is a firm need to investigate the features that will help software development companies to effectively integrate lean and ASD.

A survey is distributed in two hundred software development companies and the results are concluded based on the participation of four hundred and eight people [13]. The results show that 58% of professionals are using agile and/or lean. 33% of the practitioners are only using agile methods, 21.6% of the respondents report the adoption of agile and lean methods whereas 2.7% of the developers show the usage of only lean methods. The results show that ASD brings features such as quality products, time-saving and cost reduction whereas lean also targets factors such as faster delivery, quality products and customer satisfaction (through add value). It is also inferred from the results of a survey that both ASD and lean can be effectively applied in parallel on a sprint/release without replacing each other. The respondents are of the opinion that the successful integration of lean and ASD lies in the fact of the appropriate selection of methods and practices by the adopted team. The research lacks in providing a mechanism to integrate lean and ASD effectively.

A systematic review is reported about the effects of lean methods on the field of software engineering [14]. The main objective is to explore the main practices and approaches to integrate lean and software development methods. Four parameters are used as selection criteria to include a study in the research to investigate. Five hundred and forty-nine studies are approached that are published from 2012 to 2016. Eighty studies are included in the research to infer the results. The research categorizes seventeen tools and thirty-five practices to apply lean methods. Agile methodologies are getting popularity from last several years such as XP and Scrum. It is reported that more studies are required to integrate lean and agile methods to accumulate the benefits of both approaches.

The study is conducted to check the effective integration of lean and agile software development (ASD) in Elektrobit (EB) Company of Finland that is well known in the market of wireless embedded systems [15]. EB initializes the Scrum in 2007 and the integration of lean and Scrum is started in 2010. Rodriguez et al. [15] conduct workshops, discussion sessions and survey in EB to measure the effectiveness of lean and Scrum integration. The main goals of evaluation are established on the basis of lean principles and core agile practices. Three research questions are narrated to identify features to integrate, challenges to overcome while integrating, and required outcome after integrating lean and Scrum.

The results of the study are evident to previous studies such as include certified Scrum Masters; keep control of work in progress (WIP), effective communication and coordination, early delivery, the culture of pull model instead of a push model, consistent integration, automated testing and transparency [15]. The notable difference in the study of EB and previous studies is the extensive usage of JIRA during the integration of lean and Scrum in EB. JIRA is a strong project management tool to plan, monitor and control the agile projects. JIRA enables the entire team of EB to view information throughout the SDLC about each activity that is related to the development or business tactics and objectives. The main limitation of the study is that the results are concluded based on a single case study. There is a need to conduct more studies in other research settings to integrate lean and Scrum to generalize the results.

A survey of thirty applications, from twenty-eight companies, is illustrated about the experiences of teams deploying lean and ASD approaches [16]. The research is conducted to achieve two core objectives i.e., comprehensive understanding of lean and ASD strategies to integrate both approaches and highlight the methods to employ these strategies for the effective integration of lean and ASD. The study is an attempt to find the commonalities between lean and ASD principles, practices and methods. It is concluded that eliminate waste and Kanban practices of lean are widely practiced in software development companies due to their compatibilities with ASD. Four out of twenty-eight ASD companies are getting the benefit of work in progress (WIP) practice of lean. A notable point is concluded that ASD software companies are gradually switching from time box processes to flow based processes in the recent years while adopting lean approaches such as 30 days' release cycle of sprints using Scrum is replaced by controlling WIP and Kanban board practices of lean. It is also concluded that there is no single solution that can fit all ASD companies to adopt lean. The ASD companies have to tailor the lean principles according to their requirements, resources, and objectives. The software companies are included in the survey vary in business domains, team size, experiences of team members and size of projects and these factors are against the sampling rules.

III. Problem Statement

The software industry has shown a great interest to integrate lean and Scrum in recent years [13]. The freeze sprint backlog is an issue of the Scrum model [16]. Scrum also lacks in engineering practices and there are several potential benefits to integrating lean and Scrum such as keep control of work in progress (WIP), effective communication and coordination, early delivery, the culture of pull model instead of a push model, consistent integration, automated testing and transparency [15]. The existing studies support the integration of lean and Scrum but there is no evidence provided with respect to a modified Scrum model after integrating lean principles and practices [15]. Therefore, there is a pressing need to propose a modified Scrum model to show the effective integration of lean and Scrum. This paper attempts to propose the LScrum model merging Scrum and lean activities to accumulate the benefits of both approaches and remove their shortcomings.

- How would an integrated Lean and Scrum model impact positively on the large size software development projects?
- Would a novel LScrum model be appreciated and adopted by the software industry for the development of large size projects as well?

IV. The Proposed LScrum Model

Sketch, Build & Assemble, Test and Inspect are the main phases of proposed LScrum as shown in fig. 1. The flow diagram of the proposed model is mentioned in fig. 2. Table 1 shows the main activities of the proposed model. Table 2 compares lean and agile approaches.

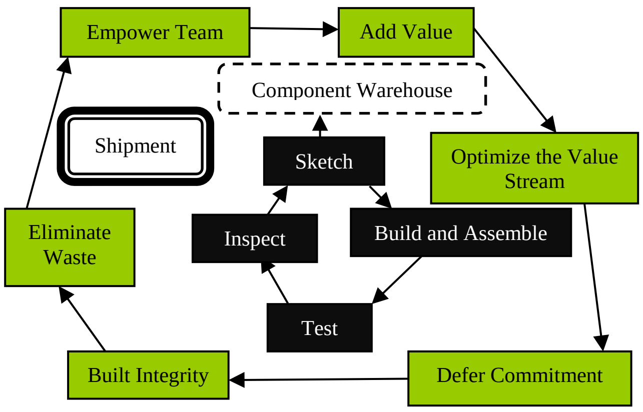

Fig.1. The proposed LScrum Model.

The core objective of the proposed model is to adapt it for large size projects to address the main limitation of existing Scrum. Therefore, large size projects are divided into several small size projects as per the compartmentalization principle of scheduling to embed agility [17]. It is suggested to have multiple teams and multiple Scrum masters to deal with large size projects. Each team is responsible for an independent part of a project and a Scrum master will manage one independent cross-functional team. The role of the scrum master is to protect his team from overburdening and distraction during a sprint in order to complete a project successfully.

A. 'Sketch' Phase

The goal of sketch phase is to accomplish planning, analysis and design activities of system development life cycle such as establish product backlog, initialize lean process, sprint planning meeting, sprint backlog and impact analysis meeting. The user stories are gathered by the product owner and the prototyping technique is used to verify the user stories. Using the prototype technique, a feedback cycle is introduced in the proposed model to deal with large size projects. Early feedback from the customer will facilitate the team to reduce the risk of changes in the user stories. The feedback loop will not terminate till the user stories are clearer, feasible and final to set the product backlog. The prototyping is not used in the existing Scrum model and this is the major reason to make it unsuitable for large size projects. The introduction of the prototype will enable a customer to change the user stories during the development of a sprint. The induction of feedback cycle will also help to implement eliminate waste principle of lean by cutting down cost, time and resources. The sketch phase will deal with the 5S of lean thinking. 5S stands for sort, set in order, shine, standardize, and sustain. The product owner and Scrum master will use 5S to manage the workplace, requirements, processes, and teams (WRPT).

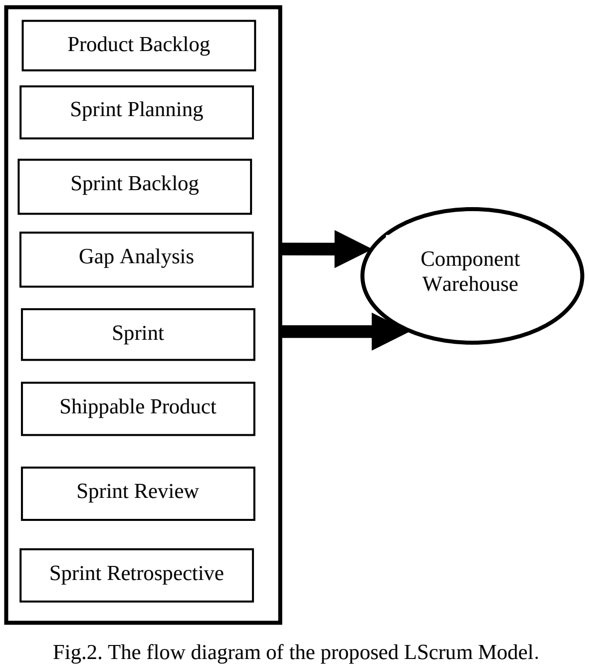

Fig.2. The flow diagram of the proposed LScrum Model.

The product owner prioritizes the product backlog. The product backlog contains the list of prioritized user stories. The product owner describes the functions and goals to be achieved. The Scrum Master and team define sprint backlog using the sprint planning meeting. The sprint backlog is the number of stories to deliver in a sprint. The scrum master, product owner, and all team members attend the sprint planning meeting. The second principle of lean is to add value to the customer. It is implemented during the sprint planning phase to find user stories with a high return on investment (ROI). The stories with high ROI are gathered in the sprint backlog. The add value principle of lean will also help the team to include only necessary features of the user stories for each sprint backlog to avoid the high cost and complexity that may lead to failing software.

A high-level architecture of the system is designed. It is kept simple to robust scalability and ease of understanding. According to Cao et al. [18], agile methodologies are not suitable for the development of medium and large projects because of wimpy architectural planning, extra-focusing on initial results, poor documentation and incomplete test coverage. The agile methodologies are more code-centric as compared to detailed architecture design and documentation that is a fundamental requirement for the medium and large projects. The agile approach is in the favor to waste all those architectural elements that do not support to current [19]. This approach works well for small projects but eliminates essential architectural features for medium and large projects and a change in architectural design consumes a significant amount of time, cost, effort, and resources [20].

The scrum master conducts a gap analysis meeting with all team members. Empower the team principle of lean is implemented during the gap analysis meeting. The team decides the number of components requires to develop from scratch and adapt using a component warehouse. The team also discusses the efforts required to develop and adapt the components in a sprint. The risks regarding the use of existing components are evaluated and managed. The relationships among components are identified. A high-level architecture is discussed to select suitable design patterns to save time and cost. The design pattern also increases efficiency, reliability, and reusability of the developed software. Scrum of Scrum meeting is organized between scrum masters of multiple sites after the sprint planning meeting to keep track of the project in the right direction. The velocity of the project is also measured to keep track of the delivery of each sprint.

B. 'Build and Assemble' Phase

New components are built and reusable components are adapted. Constraints are always there even after the components qualify for reuse into the new application, such as technical and integration constraints. Components are wrapped to manage constraints during adaptation. Assembling involves the integration of components into the architecture of the new application. The key practices of agile and lean are exercised to improve quality.

- Pair programming is used to build and assemble a component.

- Refactoring technique is performed throughout the development to improve the quality of code.
- Kanban board technique is used to monitor the work in progress (WIP).
- Plan, do, check and act (PDCA) cycles are exercised by the team.
- Defer commitment principle of lean is exercised during the build and assemble phase to ensure flexibility in requirements change management, architecture, design and code.

Table 1. The Main Activities of Proposed LScrum Model

Phases	Sketch	Build & Assemble	Test	Inspect
Activities	• Communicate to customer • Prototype • Initialize 5 S • Product backlog • Define sprint goals • Sprint Planning • Sprint backlog • Daily Scrum meeting • Scrum of Scrum meeting • Self-organized Team • accept requirements at any stage of projects (only for small projects) • High level Architecture • Gap Analysis meting • Use of design pattern	• Pair programming • Test Driven Development (TDD) • Refactoring • Kanban board • Cumulative flow diagram • Keep work in progress (WIP) under control • Plan, do, check and act (PDCA) • Defer Commitment • Code Ownership • Integration	• Unit • Integration • System • Maintain • Shippable product	• Sprint Review • Retrospective

Table 2. Commonalities between Lean and Agile Approaches

Lean Approach [12]	Agile Approach [17]
Empower the team	• Cross-functional and self-organized team (Let the team decides) • Prefer people over processes and tools • Motivated individuals to implement (Y) pull model.
Eliminate waste	• Sprint review • Inspect daily • Sprint retrospective • Amplify learning to avoid mistakes in the upcoming releases • Working software over comprehensive documentation • Trust and respect among team members to achieve a jell team to improve productivity and efficiency
Add value to the customer	• Deliver early working version to satisfy the customer • Deliver consistent releases throughout the software development • Deliver every demo and sprint
Optimize the value stream	• Sprint retrospective • Burndown Chart • Collaborate, deliver, reflect and improve cycle to facilitate communication and coordination
Built Integrity	• Test first development to minimize bugs • Pair programming to improve the quality of code • Sprint Review • Refactoring technique to improve the quality of design and code • Small releases will improve traceability, testing, maintenance • Code ownership will improve the quality of code • Coding standards to use the code as a technical document • Face to face meetings will achieve vivid and stable requirements • Keep it simple (KIS) principle • Customer collaboration over contract negotiation • Strong communication and coordination to deliver high quality software • Continues integration will help to identify integration errors
Deliver as fast as possible	• Deliver 1st release in couple of weeks and complete software in couple of months • Metaphor practice will help to deliver quick releases • Use automated tools to speed up the process of testing and maintenance
Defer commitment	• Welcome to change requirements at any stage of development • Adjust and tune according to the situation • Velocity of the project must be measured throughout the software development • Respond to change over following a strict plan

C. 'Test' Phase

The Unit tests are designed before coding to implement test-first development. Test-driven development (TDD) environment will be used to speed up the development and testing cycles. The use of automated testing tools will ease a team to improve the quality of a build and it is, in contrast, to lean development approach i.e., built integrity. The acceptance test is performed before a sprint is released. The product owner makes sure that software is delivered in small releases to achieve scalability, manageability, productivity, quality assurance and customer satisfaction. As the unit tests are completed, the developer will integrate the code into a configuration management system to accomplish the integration and system testing. The integration and system tests are performed to identify the technical, integration and architectural bugs. The configuration manager will perform integration and system testing.

D. 'Inspect' Phase

The sprint review and retrospective meetings are conducted during the inspect phase. The sprint review meeting is performed at the end of a sprint. The post mortem analysis activity is added into the sprint review meeting in the proposed LScrum model to increase its scalability to develop the large size projects as supported by other researchers [21,24]. The sprint review meeting is conducted with an objective to add value to the customer and it is coherent to lean principles as well. The sprint is labeled to deliver after approval of the product owner.

The sprint retrospective meeting is conducted to improve learning of team members and avoid mistakes in the upcoming sprints. The objective of retrospective meeting is to establish a mechanism of feedback loop throughout the software development. One of the main objectives is to reorganize extra actions required to improve upcoming sprints. The lessons learned are considered at the start of upcoming sprint. The sprint retrospective meeting will facilitate a team to take improvement actions, monitor the level of adoption, get feedback about the experiences of team and recommend necessary actions for the upcoming sprints. This proposed model would be helpful for software development to manage the large size projects and succeed the following goals in software industries [15].

E. Goal 1-Determine the benefits/suitability of the proposed L-Scrum model over existing agile and traditional models

The lean and agile integration as described in [16], our first goal is to see whether the proposed model is better than traditional models adopted by software development organizations to accomplish large size projects successfully. Software development organizations are professional centers and follow some traditionally specified model for the development process. These models are adapted and exercised by them for years. Therefore, our basic goal is to check the suitability of the proposed LScrum model in these organizations for large size projects.

F. Goal 2-Reduce development time and cost due to the proposed LScrum in Software Development Organizations

It was found by Ebert et al. [12] that the lean approach can decrease the time to develop the large size projects efficiently if:

- strategy of market is understood well;
- changes in requirements are analyzed and better managed;
- learn from previous bugs;
- software should be developed from reusable components;
- stress on repeatable processes;
- add value to the customer by early delivery of working software;
- optimize the value stream to reduce the cost of software development life cycle;
- remove extra features;
- estimate the cost and time impact before considering any new requirement to implement;
- increase collaboration to avoid any ambiguity and achieve common scope and goal;
- use automated tools to decrease number of defects and detect early bugs.

By affirming these guidelines for the integration of lean and agile, it would be provoking to propose a LScrum model to cut down the development time and cost with better quality.

G. Goal 3-Reduce waste and unnecessary features using Kanban to keep WIP under control

The kanban approach is a notable contribution to the agile and lean software development [16]. Kanban term is also taken lean manufacturing. The major function of using a Kanban approach to reduce the amount of WIP. Excessive WIP is also considered as a waste in lean approach. The kanban is integrated in the proposed LScrum model to use the pull strategy instead of a push strategy. Goal 3 is narrated to measure the impact of Kanban on the proposed model to keep the WIP under control.

H. Goal 4-Implementation of the Proposed LScrum model to achieve high quality software

The traditional methodologies are not accomplishing the requirement of software organizations to attain fast development without sacrificing quality while agile methodologies are not suitable for medium and large development projects because of weak documentation, poor architecture and lack of risk management practices. Our goal is to:

- implement the proposed LScrum model in organizations where traditional methodologies are

already implemented to develop large-scale software;
- measure its impact on the quality of software development.

V. Validation

In quantitative research, more than 30 sample sizes or any number of interviews can be conducted to know their views, thinking, and ideas regarding a specific topic. Structured questionnaires are usually used mainly based on close-ended questions with set responses [25]. Quantitative studies consist of many methods (e.g. interviews, questionnaires, analysis etc.) to perform exploratory research. It is a studying process to deal with different variables and outcomes. Survey research is one of the important areas of measurement and it is applied in social research. The wide area of survey research encompasses any measurement procedures that involve asking questions from respondents and experts. This research uses questionnaire to conclude the results. The questionnaire is distributed in twenty-four software development companies of Pakistan. The random sampling is used in this research that is suitable for our research settings. The total number of sample size is thirty-four in our research. The minimum sample size is thirty and it is considered sufficient to calculate mean, variance and cumulative frequency of the questionnaire [26]. Our research is descriptive in nature and the results are mainly concluded based on frequency tables and bar charts.

A. Cumulative Analysis of Goal 1.

Table 3. Cumulative frequency analysis of goal 1

Q. No.	Very Low	Low	Nominal	High	Very High
Q1	0	2.9	0	50	47.1
Q2	0	2.9	8.8	73.5	14.7
Q3	0	0	26.5	44.1	29.4
Q4	0	2.9	20.6	38.2	38.2
Total	0	8.7	55.9	205.8	129.4
Avg.	0%	2.1%	13.9%	51.4%	32.3%

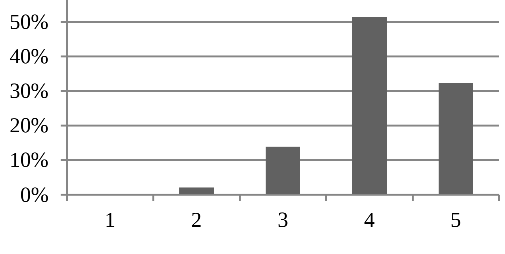

Fig.3. The Cumulative Analysis of Goal 1.

Cumulative responses of Goal 1 (Determine the benefits/suitability of the proposed L-Scrum model over the existing agile and traditional models) were shown in Table 3. Table 3 showed that eighty-two percent of the responses were in favor of Goal 1, of which fifty-one percent agreed while thirty-two percent strongly agreed to the effects of the proposed model over the existing and traditional models. Thirteen percent of the cumulative responses were neutral while two percent of the participants were not in favor of Goal 1. Fig. 3 shows the results graphically.

B. Cumulative Analysis of Goal 2.

Cumulative responses of Goal 2 (Reduce complexity and cost) were shown in Table 4. Table 4 showed that sixty-eight percent of the responses were in favor of Goal 2, of which forty-two percent agreed while twenty-six percent strongly agreed to the effects of the proposed model on reducing the cost and time. Twenty-three percent of the cumulative responses were neutral while seven percent of the participants were not in favor of Goal 2. Among them, six percent of the respondents were not agreed and one percent of the professionals were strongly disagreed. Fig. 4 shows the results graphically.

Table 4. Cumulative analysis of goal 2

Q. No	Very Low	Low	Nominal	High	Very High
Q6	2.9	17.6	23.5	44.1	11.8
Q7	0	5.9	41.2	41.2	11.8
Q8	0	11.8	14.7	32.4	41.2
Q9	5.9	0	26.5	50	17.6
Q10	0	2.9	17.6	41.2	38.2
Q11	0	5.9	17.6	38.2	38.2
Q12	0	0	26.5	50	23.5
Total	8.8	44.1	167.6	297.1	182.3
Avg.	1.2%	6.3%	23.9%	42.4%	26.0%

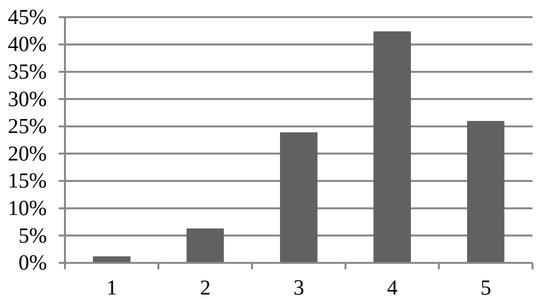

Fig.4. The Cumulative Analysis of Goal 2.

C. Cumulative Analysis of Goal 3.

Table 5. Cumulative frequency analysis of goal 3

Q. No.	Very Low	Low	Nominal	High	Very High
Q13	2.9	26.5	47.1	14.7	8.8
Q14	5.9	26.5	35.3	23.5	8.8
Total	8.8	53	82.4	38.2	17.6
Avg.	4.4%	26.5%	41.2%	19.1%	8.8%

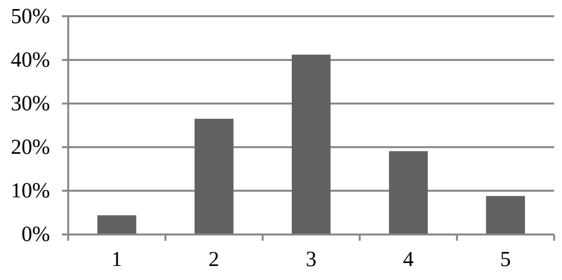

Fig.5. The Cumulative Analysis of Goal 3.

Cumulative responses of Goal 3 (Reduce waste and unnecessary features using Kanban to keep the WIP under control) were shown in Table 5. Table 5 showed that twenty-seven percent of the responses were in favor of Goal 3, of which nineteen percent agreed while eight percent strongly agreed to the effects of the Kanban on eliminating waste and unnecessary features. Forty-one percent of the cumulative responses were neutral while thirty percent of the participants were not in favor of Goal 3. Among them, twenty-six percent of the respondents were not agreed and four percent of the professionals were strongly disagreed. Fig. 5 shows the results graphically.

D. *Cumulative Analysis of Goal 4.*

Cumulative responses of Goal 4 (Increase the quality of software due to the proposed LScrum model) were shown in Table 6. Table 6 showed that sixty-seven percent of the responses were in favor of Goal 4, of which twenty-one percent agreed while forty-six percent strongly agreed to the effects of the proposed model on increasing the quality of the developed software. Twenty-five percent of the cumulative responses were neutral while six percent of the participants were not in favor of Goal 4. Fig. 6 shows the results graphically.

Table 6. Cumulative analysis of goal 4

Q. No	Very Low	Low	Nominal	High	Very High
Q15	0	8.8	26.5	44.1	20.6
Q16	0	0	17.6	58.8	23.5
Q17	0	11.8	32.4	41.2	14.7
Q18	0	8.8	41.2	38.2	11.8
Q19	0	8.8	14.7	50	26.5
Q20	0	2.9	20.6	44.1	32.4
Total	0	41.1	153	276.4	129.5
Avg.	0%	6.8%	25.5%	46.0%	21.5%

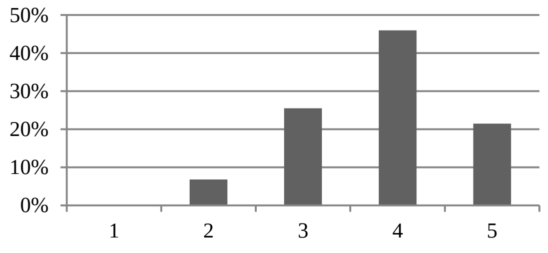

Fig.6. The Cumulative Analysis of Goal 4.

E. *Final Cumulative Analysis of Four Goals.*

Final cumulative responses of the four Goals were shown in Table 7. Table 7 showed that sixty-six percent of the responses were in favor of four goals, of which forty-two percent agreed while twenty-three percent strongly agreed to the proposal of LScrum model. Twenty-four percent of the cumulative responses were remained neutral while nine percent of the participants were not in favor of four Goals. Fig. 7 shows the results graphically.

Table 7. Cumulative frequency analysis of 4 Goals

Goal No.	Very Low	Low	Nominal	High	Very High
Goal 1	0.2	2.1	13.9	51.4	32.3
Goal 2	1.2	6.3	23.9	42.4	26.0
Goal 3	4.4	26.5	41.2	19.1	8.8
Goal 4	0	6.8	25.5	46.0	21.5
Total	8	51	158	277	154
Avg.	1.2%	7.8%	24.3%	42.6%	23.7%

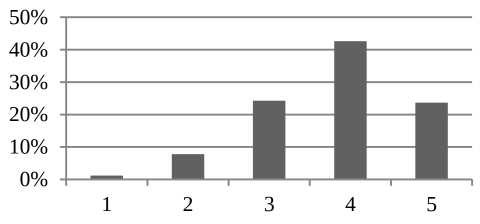

Fig.7. The Cumulative Analysis of 4 Goals.

VI. Conclusion

The agile models are proposed to deal small size project with small teams. The agile models are getting popularity from the last several years in the software industry due to their enormous benefits such as time-saving, economical, high quality and customer satisfaction. There are several adoptions of agile models reported in recent years to scale them for medium and large size projects. Scrum is one of the widely practiced agile models in the software industry due to its strong management practices and flexibility to integrate with other models. It is a lightweight process model and it supports only small and medium size projects. Lean principles are introduced to the manufacturing industry. The lean development main principles are implemented by Toyota Company in the 1950s and several benefits are achieved such as manufacturing cost is minimized, quality of the product is increased and significant effect on return on investment. There is a common spirit in both the lean and agile approaches. The scrum and lean are selected to conduct this research because both are very popular in their domains. The LScrum model is proposed to accumulate the benefits of lean and Scrum and making it suitable for the development of large size projects. A survey is used to evaluate the proposed LScrum model. The results are found encouraging. Future work is to check the effectiveness of the proposed LScrum model using industrial case studies to generalize the results.

REFERENCES

[1] W. Sacchi, *Process Models in Software Engineering*. John Wiley and Sons, New York, 2001.

[2] B. Curtis, H. Krasner, and N. Iscoe, "A Field Study of the Software Design Process for Large Systems," *ACM Communications*, vol. 31, no. 11, pp. 1268-1287, 1988.

[3] L. Jiang, and A. Eberlein, "Towards a framework for understanding the relationships between classical software engineering and agile methodologies," *Proc. Workshop. Scrutinizing Agile Practices or shoot-out at the agile corral*, Leipzig, Germany, 2008.

[4] J. Armitage, "Are agile methods good for design?," *ACM Interactions J.*, vol. 11, no. 1, pp. 14-23, 2004.

[5] M. Beedle, M. Devos, Y. Sharon, K. Schwaber, and J. Sutherland, "SCRUM: An extension pattern language for hyper productive software development," Available at: http://jeffsutherland.org/scrum/scrum_plop.pdf, [Accessed: January 30, 2019].

[6] B. Barton, "All-Out Organizational Scrum as an Innovation Value Chain," *Proc. 24th Int. Conf. On System Sciences, Hawaii*, 2009.

[7] J. Widman, S. Y. Hua, and S. C. Ross, Applying Lean Principles In Software Development Process–A Case Study. *The International Journal of Issues and System*, vol. 11, no. 1, pp. 635-639, 2010.

[8] Y. Sugimori, K. Kusunoki, F. Cho, and S. Uchikawa, Toyota production system and Kanban system: materialisation of just-in-time and respect-for-human system. *International Journal of Production Research*, vol. 15, no. 6, pp. 553-564, 1977.

[9] G. Benefield, "Rolling Out Agile in a Large Enterprise," *Proceedings of the 41st Conference on System Sciences*, Waikoloa, HI, USA, 2008.

[10] L. Williams, and A. Cockburn, Agile software development: it's about feedback and change. *Computer J.*, vol. 36, no. 6, pp. 39-43, 2003.

[11] D. Cohen, M. Lindvall, and P. Costa, An introduction to agile methods. *Advances in Computers J.*, vol. 62, no. 66, pp. 1-66, 2004.

[12] C. Ebert, P. Abrahamsson, and N. Oza, Lean Software Development. *IEEE Software*, vol. 25, no. 5, pp. 22-25, 2012.

[13] P. Rodrǵuez, J. Markkula, M. Oivo, and J. Garbajosa, "Analyzing the Drivers of the Combination of Lean and Agile in Software Development Companies," *Proceedings of International Conference on Product Focused Software Process Improvement*, Spain, 2012.

[14] F. Sambinelli, and M. A. F. Borges, Lean Thinking in software Engineering: a Systematic Review. *International Journal of Software Engineering & Applications (IJSEA)*, vol. 8, no. 3, pp. 15-32, 2017.

[15] P. Rodrǵuez, J. Partanen, P. Kuvaja, and M. Oivo, "Combining Lean Thinking and Agile Methods for Software Development A Case Study of a Finnish Provider of Wireless Embedded Systems," *Proceedings of the 47th International Conference on System Science*, Hawaii, 2014.

[16] X. Wang, K. Conboy, and O. Cawley, Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. *The Journal of Systems and Software*, vol. 85, no. 6, pp. 1287-1299, 2012.

[17] R. S. Pressman, Software Engineering A Practitioner's Approach. McGraw Hill, USA, 2015.

[18] L. Cao, K. Mohan, X. Peng, and B. Ramesh, "How extreme does extreme programming have to be adapting xp practices to large scale projects," *Proceedings of the 37th Annual International Conference On system sciences*, Hawaii, 2004.

[19] B. Boehm, Get ready for agile methods with care. *Computer*, vol. 35, no. 1, pp. 64-69, 2002.

[20] D. E. Turk, R.B. France, and B. Rumpe, Assumptions underlying agile software-development processes. *Journal of Database management*, vol. 16, no. 4, pp. 62-87, 2005.

[21] P. Abrahamsson, and J. Koskela, "Extreme programming: a survey of empirical data from a controlled case study," *Proceedings of the International Symposium Empirical Software Engineering*, USA, 2004.

[22] O. Salo, and P. Abrahamsson, "Empirical Evaluation of Agile Software Development: The Controlled Case Study Approach," *Lecture Notes in Computer Science*, Springer Verlag, Berl ń, Heidelberg, 2004.

[23] M. Visconti, and C. Cook, "An Ideal Process Model for Agile Methods," Lecture Notes in Computer Science, Springer Verlag, Berl ń, Heidelberg, 2004.

[24] R. Miller, "Demystifying Extreme Programming, "XP distilled," Available at: https://www.ibm.com/developerworks/java/library/j-xp1008/index.html, [Accessed: January 31, 2019].

[25] M. D. Medley, "Using qualitative research software for CS education research," Proceedings of the 6th annual Conf. on Innovation and technology in computer science education, UK, 2001.

[26] iSixSigma, "How to Determine Sample Size, Determining Sample Size," Available at: http://www.isixsigma.com/library/content/c000709a.asp, [Accessed: January 31, 2019].

Authors' Profiles

Dr. M. Rizwan Jameel Qureshi received his Ph.D. degree in Computer Sciences from National College of Business Administration & Economics, Pakistan 2009. He is currently working as a Professor in the Department of IT, King Abdulaziz University, Jeddah, Saudi Arabia. This author is the best researcher awardees from the Department of Information Technology, King Abdulaziz University in 2013 and 2016. He is also honoured as a best researcher in Computer Science discipline from seven campuses of COMSATS Institute of Information Technology, Pakistan in 2008.

Dr. Madini O. Alassafi received his Ph.D. degree in Cloud Computing, University of Southampton, UK, 2018. He is currently working as a Chairman in the Department of IT, King Abdulaziz University, Jeddah, Saudi Arabia.

Hafiz M. Shahzad received his MSCS degree in Computer Sciences from COMSATS Institute of Information Technology, Lahore, Pakistan in 2009. He is currently working as an Assistant Professor in the Department of Computer Science, Superior University, Lahore, Pakistan. His research interests include integration of agile and lean software development.

How to cite this paper: Rizwan J. Qureshi, Madini O. Alassafi, Hafiz M. Shahzad, "Lean Agile Integration for the Development of Large Size Projects", International Journal of Modern Education and Computer Science(IJMECS), Vol.11, No.5, pp. 24-33, 2019.DOI: 10.5815/ijmecs.2019.05.03