Software Development by Merging Prototyping and Agile Approaches

Mohammed Al-Refa, Basem Zughoul, Yousef Al-Raba‘nah, Mohammad Alauthman, Ahmad Muraish

1, 3, 4 Department of Computer Science, Faculty of IT, Zarqa University, Zarqa, Jordan
2 Department of Software Engineering Faculty of IT, Aqaba University of Technology, Aqaba, Jordan
5 Department of Software Engineering Faculty of IT, Zarqa University, Zarqa, Jordan

Abstract—during the past four decades, new software development approaches were presented to going with new trends of the software development companies and developers. Nowadays most software companies search to customise short period and minimal costs to produce valuable software products. These productions are within unstable, changing environments. Agile Methodologies were thus introduced to achieve the new requirements of the software development companies. Agile development invented for handling change. The key benefit of agile methodology is used to simplify the change-oriented software engineering process, but agile methodology needs to take more consideration to user point of view. Although the weaknesses of prototyping approach have, it can be used effectively by end users to describe requirements that developers may not have considered. This paper presents a proposed framework based on merge prototyping and agile approaches not only to cure the weaknesses of agile development and prototype methodology but to enrol the user in every single step, which will lead to faster and much accepted overall results.

Keywords—Agile; Prototyping; SDLC, Merging Approaches

I. INTRODUCTION

There are several ways to develop software in different systems. Waterfall, prototype (evolutionary), Incremental and agile are Examples of Software development approaches. Choosing Software development method depends on the nature of the system. Scientists and developers always try to expand and change in the System Development Life Cycle (SDLC) approaches to increase productivity and flexibility. The software has merged into many diverse fields and is becoming more sophisticated to select one approach. Also changing requirements from customers increase the difficulty of choosing some approaches. Traditional SDLC methods are not able to satisfy the new demands of end users in the best way, anymore. As a result, of these problems and improvements, new software development methods are evolved, as agile methodologies, in all of its types. The new approaches include modifications to SDLC, to make them more productive and flexible [9, 3,10,17].

Typical SDLC approaches assume that all requirements are complete and can be implemented directly to develop the application. However, in today projects, changes are frequently in project requirements which are in the development process. These systems need a particular development that keeps in mind the customer requirements changes. Requirement engineering in today systems are doing in parallel to software development. Requirements Changes occur much time during the development process to obtain customer satisfaction[2,11,12].

This paper presents a framework based on combining prototyping and agile approaches not only to cure the weaknesses of agile development and prototype approaches but to enrol the user in every single step, which will lead to faster and much accepted overall results.

This paper organised as follows. After this introduction in section one, section two discusses related work. Section three presents prototype and agile approaches properties, strengths and weaknesses. Section four explains the proposed framework. In section five conclusion and future work are presented.

II. RELATED WORK

There are a lot of studies on SDLC in different approaches; researchers focused on improving methodologies to obtain productive and flexible approach. This section presents some of these studies to give the reader some feedback.

In [7, 9] a review of new software development approach was introduced to fit the new technology of the software development companies. This research presented an analysis of three agile approaches the Extreme Programming, SCRUM, and Agile Modeling. It described the differences between them and recommended when to use
them. The paper defined that the main aim of agile methodologies is to deliver what is needed when it is required. The paper defined the various type of Agile methodologies. Although they share the same basic concepts, They have some variations. The main agile approaches that used include XP, SCRUM, and Agile Modeling. XP is the coding of what the customer specifies and the testing of that code. SCRUM, on the other hand, supports management role in software development. Agile Modeling defines a collection of values, principles, and practices which describe how to streamline modeling and documentation efforts. Agile methodologies are not always the best suited for all projects. When development team includes mainly beginners, or when communication between developer and the customer is difficult. On the other hand, when there is a strong communication, the development team compromises skilled team members, or when the deadlines and budgets are tight, then Agile methodologies are among the best software development approaches to implement [7].

In [8] researchers presented an empirical study for the software industry to make integration of both agile and traditional v model. A survey conducted to compare the result of and systematic review of literature carried out to test the results of this integrated methodology. The results provided positive responses for better enhancement of quality, efficiency, maintainability and suitability. Also, the result reduced the time and cost which is a great benefit for SDLC methodology. The limitation of this research suggested that for further research to enhance this research it should use a big data collection, as in this work the sample size is very low and data collected is small. If the size of data is small, it will help in understanding more clearly [8].

In [1] a dissertation presented a comparison among SDLC methodologies, it described the characteristics of some traditional and agile that development approaches. It has also discussed the strengths and weakness of different methods and provided the challenges associated with implementing agile processes in the software development. According to their findings, agile approaches can better for small and medium scaled projects, but for large projects, traditional approaches seem better [1].

III. AGILE AND PROTOTYPING APPROACHES

There are several ways to develop software projects, the most popular methods are: waterfall, prototyping or evolutionary and Agile methodologies. In this section agile and prototyping approaches will be discussed and compared with each other.

1. Agile: at the starting of software development, most of the user’s requirements were stable, and SDLC followed the plans without major changes. Agile came about as a solution to some difficulties of the waterfall approach [13][14]. These difficulties include evolving requirements, customer involvement, deadlines and budgets, and miscommunications. The agile methodology uses an incremental approach instead of the traditional waterfall approach. Developers start the project with a simple project design and then begin working on small modules. The work on these modules stays for weekly or monthly sprints, and at the end of each sprint, project priorities evaluated and tested. These sprints allow discovering bugs, and customer feedback to be considered in the design before the next sprint starts.

![Fig. 1 ordinary agile life cycle, where the user's opinion is only taken after the work is partially done](image-url)
The agile methodology should use when rapid production is more important than the quality; clients can change the scope of the project, the picture of the final product is not clear and when the developers have the skills for adaptation and independent thinking [6].

Agile Methodology has many advantages; it allows making changes after the initial planning, so it is easier to add new features that will keep the product up to date with the latest developments. At the end of each sprint, project priorities re-evaluated, which allows the customer to add their feedback so that they get their desired product. The testing at the end of each sprint ensures that the bugs are caught and will not found at the end of the project. On the other hand, the agile methodology has disadvantages; as the initial project does not have a defined plan, the final product can be different than what was initially intended, and the software project is likely to come in late and over budget [15]. Figure one shows the agile model and how that the user opinion taken after the project is partially done.

2. Prototyping approach: An iterative process of quickly and inexpensively creating a live and working model to test out requirements and assumptions [5]. Prototyping used as an alternative or in addition to formal functional specifications, the prototyping method capture works well if the software product is not yet entirely defined. Prototyping can also be used effectively by end users to describe and prove the need for requirements that developers may not have considered [1].

Prototyping methodology has many advantages; improved communication, improved user involvement, a feedback loop established, learning by doing, Leads to Stakeholder acceptance, reduces the need for documentation, lowers maintenance costs. On the other hand, the prototyping methodology has disadvantages; the user may misunderstand the nature of the prototype, loss of project control and standards are possible, an additional cost of building a prototype and focus on the user-friendly interface could be at the expense of machine efficiency [5]. Figure two shows prototyping model that explains how customer makes his decision and evaluation up to the effect prototype design.

![Fig. 2 Ordinary prototyping model, where the user's opinion depends on the prototype sample.](image)

Both the Agile and Prototyping methodologies have their strengths and weaknesses. Considered properties of both approaches are taking and combining them to make the best possible software development process for software projects.

IV. PROPOSED FRAMEWORK

After the presentation of prototyping and agile methodologies in the previous section, this paper suggests a new frame work that customises the strengths of both models to enrol user in each step in agile by present a simple and easy short prototype. Our proposed solution based on merging prototyping and agile approaches together not only to cure the weakness of agile development but to also enrol the user in every single step, which will lead to faster and much accepted overall results.

The above figure adds another step to every segment in the life cycle in agile approaches, which means the user's opinion will be taken at the end of each segment, and after all, segments are complete the combination will be tested and the user will also give his feedback for it.
Fig. 2 Proposed model that enrol the user in each step of all agile process

As shown in figure three the proposed framework enrol the user in developing each phase using prototype before integration and testing. This process repeated for each step in the agile methodology.

V. CONCLUSIONS

In this research, a framework of software development process proposed and explained. This paper starts with describing the prosperities of agile and prototyping development methodologies. It explained the strengths and weaknesses of these approaches. The idea beyond this research is to combine agile and prototyping methods to enrol the user in all steps during the SDLC. Researchers and industry should adopt and apply this idea in the future.

REFERENCES


