Research Article

Estimation of software quality parameters for hybrid agile process model

Lalband Neelu[1] · D. Kavitha[2]

Received: 17 August 2020 / Accepted: 29 January 2021 / Published online: 9 February 2021

Abstract
This study aims to motivate a new hybrid agile methodology is a combination of agile models from Scrum, Extreme Programming, and Lean Software Development. The main aim of the hybrid agile model is the timely delivery of projects to clients with high quality at a reduced rate. But the main difficulty in hybrid agile model to effectively reflect the software quality attributes. Basically, the failure of a software project is mainly not because of inefficiency of functional features but due to inefficiency of quality attributes, like performance, reliability and effective usability. The work presents the introduction of Hybrid Agile Quality Parameter Analysis (HAQPE) that is a quality attribute driven agile development method. The outcome of developed quality attribute HAQPE was evaluated through hybrid agile process assessment by applying it to a commercial project of software industry. The results demonstrated that the developed quality attribute model is more efficient than the legacy agile development process.

Keywords Agile software development · Software quality · Hybrid agile

1 Introduction

There is a more challenging demand nowadays for software organizations to deliver high quality product on time with low budget. Software processes help them achieve this goal, gain their customer satisfaction, accomplish competitive advantage and satisfy their shareholders financial demands.

"Software engineering can be explained as a phenomenon that includes some steps to produce high-quality software depending on the customers [1]. It is a task that is intricate to come up with high-quality software. It is because of software engineering that people can design software and make improvements to the quality throughout the SDLC (Software Development Life Cycle). A framework that offers ways of structure plans that are different, and also controls the process of developing software is known as software development methodology [2]. Methods, including principals, policies, and procedures are utilized for the implementation of the software. Most methodologies of software development are found in the software industry [3]. Every methodology has its software development cycle that makes it different from the others. Some of the names that are essential developments include prototyping, Waterfall, incremental, and spiral. Some ways were used in the past to develop software and are recognized as methodologies that are heavyweight [4]. The methods include lesser involvements of customers, gigantic documentations, and low flexibility for creeping needs. A survey was done by Cockburn showing that in the case of changes in a business, a higher percentage of the functionalities that are delivered are rarely utilized and, in some cases, never used at all [5]. In the last two decades, methodologies of Agile software development such as

✉ Lalband Neelu, neelulalband@gprec.ac.in; D. Kavitha, dwaramkavithareddy@gmail.com | [1]Department of Computer Science and Engineering, Jawaharlal Nehru Technological University, Anantapur, India. [2]Department of Computer Science and Engineering, G. Pulla Reddy Engineering College, Kurnool, India.

extreme programming (XP), Scrum, Kanban, dynamic systems development method (DSDM), Lean Software Development, Feature Driven Development (FDD) among others are the best known for the development of life cycles that is fast, the need for lesser documentation, and costs that are low for projects [6, 7]. In Agile, there is a strong collaboration between the testers and developers; increasing chances of errors can be reduced [8]. Agile methodologies are not rigid and include more customers' involvement as a practice of quality assurance [9]. However, every methodology of developing software has its advantages and disadvantages. Research confirms that the success rate of agile software development is around 71.5% [10]. Agile methodologies are being embraced by more and more organizations recently because of the rigid behavior of traditional frameworks and the complexity that is rising of software projects [11].Agile methods are being utilized as a new process in the development of software. They are mostly used in fastening software development cycle while focusing on quality. The use of agile methods in any project with the perspective of quality assurance is an essential task. In modern-days, most software developing companies use agile methods in their projects to reduce the production cost. Most agile methods are utilized by company that develop software to produce high quality services and products.

To be able to deal with the advantages and disadvantages of the various methodologies, the organizations of software development are also utilizing hybrid models in the development of software [12]. The hybrid models are a combination of either two or more methodologies and are utilized with the quest to merge their benefits in one place so that companies can have the best to get maximum output [13]. Livermore states that Scrum is most of the time combined with practices of Extreme Programming (XP) [14], and they both have different flavors. According to Pressman, for building a successful software increment, XP is combined with DSDM [15]. It will result in the generation of a combo methodology that defines a powerful process model (the lifecycle of DSDM) by the use of practices of bolts and nuts (XP) [16]. Some studies combine traditional methodologies with agile. CMMI and Scrum were combined by Lina and Dan [17] to get a better framework for companies that are both small and medium-sized. An introduction that is brief of the three agile methodologies is, as shown in Fig. 1 below.

In this paper by considering all the points, a new hybrid agile software development model was proposed by

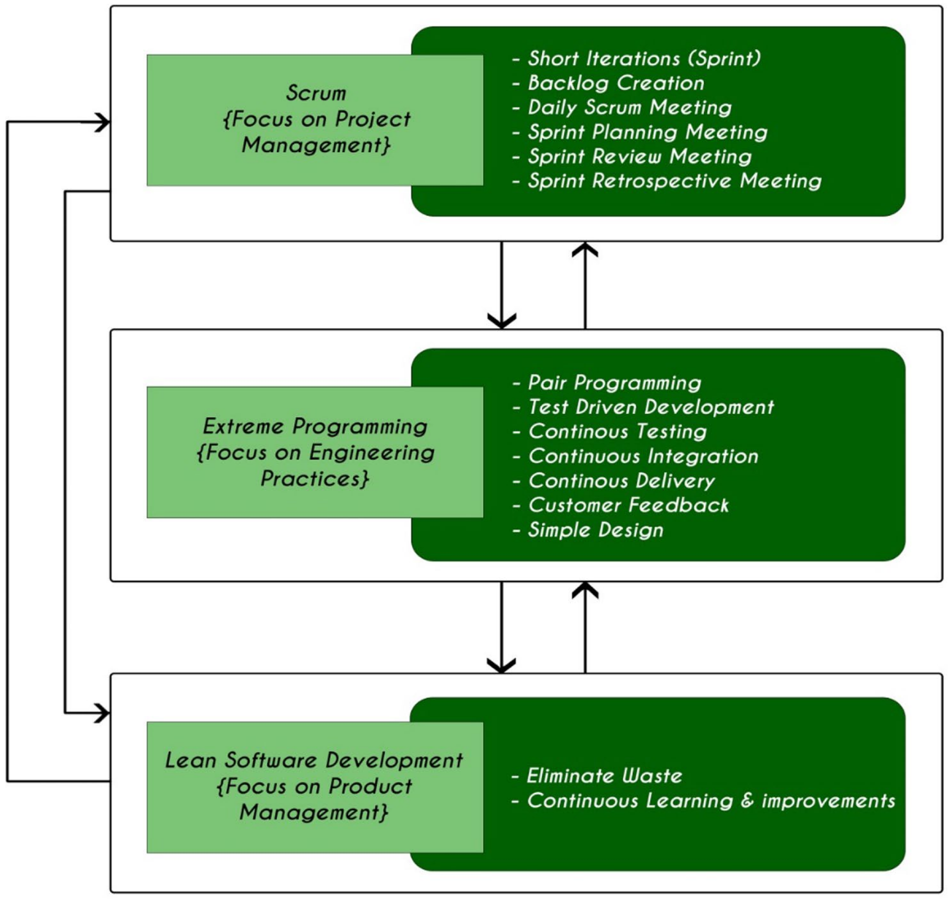
Scrum {Focus on Project Management}
- Short Iterations (Sprint)
- Backlog Creation
- Daily Scrum Meeting
- Sprint Planning Meeting
- Sprint Review Meeting
- Sprint Retrospective Meeting
Extreme Programming {Focus on Engineering Practices}
- Pair Programming
- Test Driven Development
- Continous Testing
- Continuous Integration
- Continous Delivery
- Customer Feedback
- Simple Design
Lean Software Development {Focus on Product Management}
- Eliminate Waste
- Continuous Learning & improvements

Fig. 1 Agile principles and practices of 3 different techniques

bringing together Scrum, extreme programming, and lean software development. The hybrid agile model includes all the strengths of the models that are already in existence i.e., Scrum, extreme programming, and lean software development. Additionally, the model offers better results when it comes to quality assurance when applied to IoT system projects.

The remaining of the paper is structured as follows, Sect. 2 provides the most important literatures on the domain, Sect. 3 proposed and explains about a hybrid agile model. Section 4discusses on the results obtained. Section 5 describes about the Conclusion and future scopes are provided at the end of the manuscript.

2 Background

2.1 Scrum

Scrum is being utilized for a time that is much longer when compared to other methodologies. Scrum, together with XP, is one of the agile methodologies that are widely used. Scrum focuses on the idea that processes that are defined and repeatable work in a way that they only deal with issues that are defined and repeatable with individuals that are defined and repeatable in surroundings that are also defined and repeatable [18], which is a thing that is not possible. Solving the issue of defined and repeatable processes needs the iterations of a project are divided by Scrum into 30 days sprints. Before the beginning of Sprint, there is the definition of the functionality required for the team, and Sprint left to deliver it. The aim is to stabilize the requirement during the Sprint. It is one of the reasons that Scrum is taken into consideration as the best agile methodology when it comes to application development. A sprint is a development cycle that takes about 2 to 4 weeks to implement, test, and release features agreed in the sprint backlog. The main focus of Scrum is to adopt in a way that is continuous to any changes of requirements at run time. It is a development approach that is exceptionally flexible and is guided by the principle of development that is iterative and incremental. Scrum promotes a collaborative surrounding where many cross-functional teams come together to get product targets [19, 20].

2.2 Extreme programing

Extreme programming, also known as XP, is an agile methodology that depends on fast communication, quick feedback, and simplicity to develop a software product [21]. The main focus of XP is development. XP works in a way that is basically where the work is done while assuming that time is not enough to fulfill the requirements and documentation that is exhaustive. There is also a team of not more than 12 individuals who work in pairs with matters referring to programming [22]. Apart from each of the groups, there is a client as the center of all expectations. The customer is a vital part because, during the development process, the customer can intervene to make possible modifications. The modifications are made in periods that are short such as two days which can be talking to a customer about what they need and begin programming, showing the customer what was done, in the case that the customer is happy, what follows is the next activity, and in the fact that the customer is not satisfied, corrections are made. XP is different from traditional methodologies, mostly because it focuses more on adaptability than permissibility. Defenders of XP take into consideration that the requirements of changes are an aspect that is natural and inevitable and even desirable in project development [23].

2.3 Lean software development

Lean software development is an agile framework on the basis of the optimization of time and resources for development, elimination of waste, and delivery of what is needed by a product. Lean software development like Scrum is more a set of practices of project management than an exact process. Bob Charette developed lean software development, and the success is drawn on that lean manufacturing gained in the automotive industry in the 1980s. The target of lean software development is on how CEOs consider change when it comes to the management of projects. Lean development is on the basis of the idea of lean thinking, and the origin is from the production of lean that started with Toyota Automotive manufacturing company [24]. Most of the time, lean software development is known as the Minimum Viable Product strategy, where a team releases aversion that is bare-minimum of a product to the market, learns from customers what they do not like, and wants to be added, and later iterates based on the offered feedback. Some of the strengths of lean software development are eliminating unnecessary activities, reducing costs, allowing more functionality to be delivered in less time, and empowering the development team to make decisions that can boost morale [25].

Dynamic systems development method (DSDM) [26] is a method developed by a dedicated consortium in the UK. The first release of the method was in 1994. The fundamental idea behind DSDM is that instead of fixing the amount of functionality in a product, and then adjusting time and resources to reach that functionality, it is preferred to fix time and resources, and then adjust the amount of functionality accordingly. The origins of DSDM are in rapid application development. DSDM can be seen as the first truly agile software development method.

Crystal Clear [27] is targeted at a D6 project and could be applied to a C6 or a E6 project and possibly to a D10 project.

Feature Driven Development (FDD) [28] authors Peter Coad and Jeff de Luca characterizes the methodology as having "just enough process to ensure scalability and repeatability and encourage creativity and innovation all along the way." Throughout, FDD emphasizes the importance of having good people and strong domain experts. FDD is build around eight best practices: domain object modeling; developing by feature; individual class ownership; feature teams; inspections; regular builds; configuration management; reporting/visibility of results. UML models [29] are used extensively in FDD. Table 1 shows the comparative analysis of the existing approaches.

In this section, presented our analysis of different agile software methods. We also describe what are the problems faced during implementation of agile software development. The objective is to help software engineers to understand the key characteristics of these methods and therefore select the most suitable method with respect to the type of software projects they develop. So in this work, a hybrid agile model is proposed for obtaining better results.

Table 1 Comparative analysis of the existing approaches

Concept	XP	SCRUM	DSDM	CRYSTAL	FDD
Team size	3–16	5–9	2–6	4–8	6–15
Number of teams	1	1–4	1–6	1–10	1–3
Volatility	High	High	low	High	low
Team distribution	No	No	Yes	Yes	yes

3 A proposed hybrid agile model

The hybrid agile model that is proposed, which is a combination of agile models that are widely used, namely Scrum, extreme programming, and lean software development. The hybrid agile model is a combination of the strengths of all the three models while removing their weaknesses. It is a combination of the practices of Scrum's project management with an approach that is business focused on lean software development and covering whole SDLC under the umbrella of XP engineering practices. Therefore, it makes the hybrid model that is more powerful for development projects. The best features of the hybrid model are the management of software projects in a fixed time constraint and working in a way that is well with other requirements of changing. The defect rate of the hybrid model is low when a comparison is made with the three agile models, resulting in a high-quality software product.

It has phases of both pre-project and post-project elements of SDLC. The system backlog, sprint backlog, and increment features of the hybrid model are the ones presented in Scrum. During the whole SDLC, the various phases are covered under the practices of engineering of XP. The methods include pair programming, continuous feedback, customer feedback, continuous integration, test-driven development, simple design, and continuous delivery. The combination of all the features makes the hybrid model a model that is interesting for software development organizations. The hybrid agile model is shown in Fig. 2 below. It begins with the customer's involvement phase. It is the phase where a feasibility study takes the place of the software to be built. The feasibility study ensures that the software product will be completed within a given time constraint and budget. The phase of

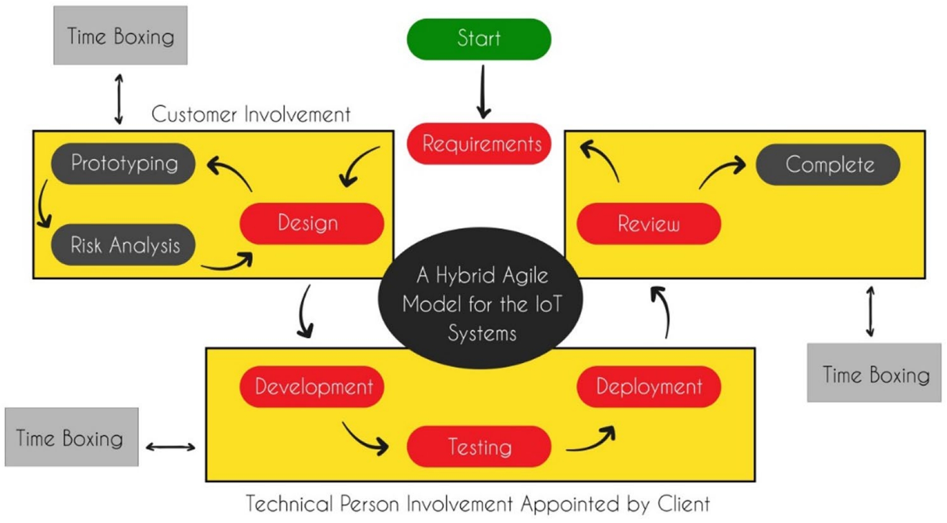

Fig. 2 Proposed software development model

customer involvement is conducted before the project officially begins.

The first essential phase of the software development life cycle starts, and it is the customers' involvement phase. It is a phase where the initial design of a product is constructed. The input to the stage is the SRS document. The phase includes prototyping, risk analysis, design along with time boxing. After modeling the product, the phase that follows is the development phase. The development phase includes coding, testing, deployment, and time boxing. Here a technical person is appointed by client is involved for better understand of the project. In this phase, the requirement classification is done depending on the priorities of the customers. The development phase is completed with consultations held with stakeholders. The stakeholders involve the hybrid model experts, owner of the product, and the hybrid model production team. The requirements that are the first priority need to be completed first. Then the main phase of SDLC will begin, and this involves detailed designs of the product, deployment, and deployment test. The phases are conducted under the umbrella of practices of XP, as shown in Fig. 3 below. It includes pair programming, coding standards, customer feedback, and continuous delivery.

After the completion of a sprint, there will be a sprint meeting held for evaluation. The purpose of the meeting is to make sure that all the requirements of the Sprint have been met and implemented. If there are any remaining or new requirements explored in the meeting, these become part of the scrum review meetings, which is the input of the system backlog for the implementation of the next Sprint. If the whole requirements are implemented, and the product owner is satisfied with the sprint outcome, then the sprint release is considered the hybrid model increment. The last stage of the hybrid agile model is the product release phase and involves working product increment. The maintenance of the project is done in the future. If a customer finds any bugs in the product, it will be resolved in the future under the product release phase. The complete process of the hybrid agile model intends to cover all the aspects of SDLC that are needed for the production of software with low costs, premium quality, and little defects. Every spring cycle of the hybrid agile model, as shown in Fig. 3 below, is 2–4 weeks long. A project will have 4–8 sprints. Therefore, the duration of the project will be different. The daily scrum meeting will be for 30–40 min, while the Sprint planning meeting will be 2–4 h. The spring kick-off meeting will take about 3–4 h because of the issues of the new products that will have to be discussed. People that will play different roles in the hybrid agile model are the hybrid agile model experts, product owner, and the production team of the hybrid agile model. The events, tasks, artifacts, and phases of the hybrid agile model are explained in detail in the following sections.

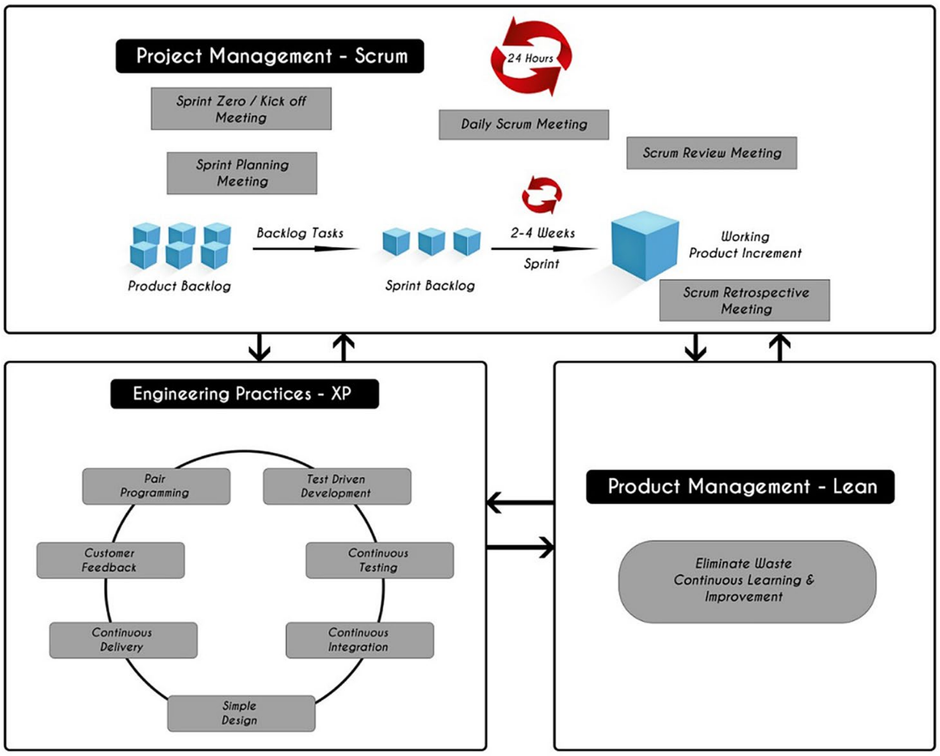

Fig. 3 A hybrid agile model

3.1 Roles of the hybrid agile model

The various roles in the hybrid agile model will be played by the hybrid agile model experts, product owner, and the production team for the hybrid agile model. The hybrid agile model expert is the person that is responsible for the whole progress of the development process in the hybrid agile model. The hybrid agile model experts act like managers and are responsible for completion that is successful in the software that is free of errors in time and within the budget that is proposed. The owner of the product is a stakeholder in the hybrid agile model and is responsible for making decisions on the work that is to be done. The product owner writes and prioritizes software requirements in the type of user stories. The user stories will then be part of the system backlog. The production team, a group of professionals, will be responsible for delivering error-free software according to a set of specific functional requirements. In the hybrid agile model, the production team consists of 5–8 fulltime members.

3.2 The hybrid agile model events

The various events of the hybrid agile model involve sprint kick-off meetings, meetings of sprint planning, daily scrum meetings, scrum review meetings, and scrum retrospective meetings. A sprint kick-off meeting is conducted as each sprint starts in the hybrid agile model to plan about the sprint that is to be started. The various participants of the sprint kick-off meeting are the hybrid agile model experts, the owner of the product, and the production team of the hybrid agile model. The daily scrum meetings are held daily for measuring the progress of the production team. The participants of the sessions include experts of the hybrid agile model and the hybrid agile model production team. The sprint planning meeting and scrum review meeting are conducted at the end of every sprint for measuring the progress of all activities of the previous sprint. The participants of the scrum review meeting and scrum review meeting are the hybrid agile model expert, the hybrid agile model production team, product owner, clients, and other partners of business in the project.

3.3 The hybrid agile model artifacts

The various hybrid agile model artifacts involve product backlog, sprint backlog, and working product increment. The system backlog in the hybrid agile model contains all the prioritized requirements that are functional and non-functional of the systems to be implemented. The hybrid agile model expert gathers the requirements from product owners in user stories on a user story card. The product backlog is shown in Fig. 3. The output of every spring is a working set of the product known as the working product increment. The scrum retrospective meeting will include a document that will be created during the daily scrum meetings and consists of the remaining or incomplete requirements. The document's requirements act as input to system backlog to be implemented in the next sprint cycle.

3.4 Phases of the hybrid agile quality parameter analysis (HAQPA) process

The phase of the hybrid agile model is the backbone of the model. They need to be conducted in a smooth way to get high-quality products that are free of defects. The phases involve the customer's involvement phase, development phase, and product release phase where HAQPA is introduced as shown in the Fig. 4. During the customer's involvement phase, there is a feasibility study where the report of a project is generated and involves answers to various essential questions about the project. The time of completion, resources, risks, and prices suggested, and

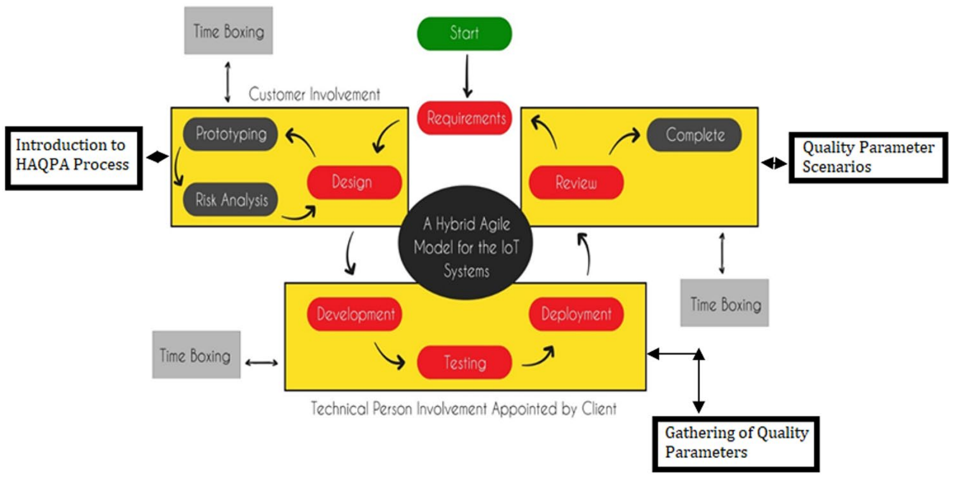
Time Boxing
Start
Customer Involvement
Requirements
Introduction to HAQPA Process
Prototyping
Design
Risk Analysis
Complete
Review
Quality Parameter Scenarios
A Hybrid Agile Model for the IoT Systems
Time Boxing
Development
Deployment
Testing
Time Boxing
Technical Person Involvement Appointed by Client
Gathering of Quality Parameters

Fig. 4 Hybrid agile quality parameter analysis (HAQPA) process

the project's usefulness is discussed in detail in the feasibility report along with traceability matrix and quality parameters. The development phase in the hybrid agile model is the first design of the project with suitable quality parameters. The first design is made on paper and then reviewed. It also offers ways that are best of solving the issues that take place during the development. The development phase in the hybrid agile model will take place once the sprints are decided in sprint kick-off meetings. The phase offers a detailed design of the modules that are implemented in a given sprint. The designing and development of the modules are in iterations. During the development, the modules of design are coded for implementation and quality parameter scenarios are developed. The hybrid agile model production team takes part in the process of development. The hybrid agile model is also there to assist the production team if they need any assistance. The whole process of development is conducted under the umbrella of the practices of XP. The exercises involve pair programming, customer feedback, continuous delivery, continuous integration, sample design, and 30 h per week. Testing is an activity that is essential in the hybrid agile model.

During this stage, the software increment is tested against all requirements that are functional and non-functional. Various types of testing involve unit testing, integration testing, and system testing. The product release phase is the last phase of the hybrid agile model, which is also a post-project phase. The product release phase starts once the products are handed to the clients. During use, if customers find any bug in the product, the customers can report it to the team of production for maintenance. The team of production removes that bug by releasing or issuing some patch. The support is done by using a similar sprint cycle as done earlier to build the software.

4 Results and discussions

There was a case study that was controlled for the validation of the proposed hybrid agile model. The results of the case study were recorded from day one. There is also a table showing the total meetings held during the implementation of the hybrid agile model. Four hybrid agile increments were produced during the case study. The initial two increments were of a duration of two weeks, while the last two increments were of a duration of 1 week. There are records of data from all four sprints. The data that is numerically collected from the project is shown in Table 2. The table shows the data from all four sprints individually in columns. The last column shows the total of all four sprints. The collected data is explained in detail. The first row indicates that the first two sprints are finished in two weeks, while the last ones took one week each, and makes the total number of sprints to be six for the project to be completed. The next row represents the total number of modules implemented during each of the modules of the sprints cycle. Thirteen modules are implemented during the first Sprint, and seven are implemented during the last sprint. A total of 40 modules are implemented during the project. The third row has 40 user stories implemented in the project in all the four sprints, which are the actual user

Table 2 Case study results of the hybrid agile model

S. No	Parameter	Sprint 1	Sprint 2	Sprint 3	Sprint 4	Total
1	Sprint duration in weeks	2	2	1	1	6
2	No. of sprint modules	13	11	9	7	40
3	No. of user stories	16	12	7	5	40
4	Estimated work effort in hours	336	336	190	190	1052
5	Actual work effort	318	307	282	104	1011
6	No. of implemented classes	56	40	32	21	149
7	Total lines of code	2010	1702	1017	8031	55,331
8	Kilo lines of code	20.10	17.02	10.17	8.03	55.33
9	Tested lines of code	9048	4033	3021	8754	24,856
10	Pre-release defects	9	7	5	2	23
11	Post-release defects	5	4	2	2	13
12	Performance/quality	0.25	0.24	0.20	0.25	0.24 (Average)
13	Suggestions of number of sprint evaluation feedback	5	4	3	2	14
14	No. of unit test	248	207	192	142	789
15	Customer satisfaction	94%	86%	97%	95%	93%(Average)
16	Team productivity	55.31	44.12	52.65	67.60	54.92 (Average)
17	Time to implement a user story	21.00	28.00	27.14	38	28.54 (Average)

requirements. The fourth row indicates a total of the estimated hours of work needed for the completed project. The total numbers of hours that are estimated for the project to be completed are 1052. The use of the formula calculates the numbers of working hours that are estimated:

$$\begin{aligned}\textit{Estimated Work Effort} = {} & \textit{Duration of a sprint in a week} \\ & \times \textit{total number of working days in a week} \\ & \times \textit{total hours of working in a day} \\ & \times \textit{the size of the production team}\end{aligned}$$

For instance, for sprint one, the estimated work effort can be calculated:

$$EWE = 2 \times 6 \times 7 \times 4 = 336$$

Row 5 indicates the actual effort in hours that are recorded for the completion of a sprint. The table shows that the end of the first Sprint took 318 h, and the total of the four sprints is 1011 h. Row 6 shows the number of implemented classes implemented in every Sprint, and a total of 149 classes are implemented in the project. Row 7 is the total lines of codes for the project, and the team of production coded a total of 55,331 for the project. Row 8 shows the number of kilo lines code, which is 47.19. The use of the following formula calculates it:

$$\textit{Kilo Lines of Code} = \textit{Lines of Code} / 1000$$

Row 9 shows the tested lines of code, and they included test cases. A total of 24,856 lines are included in test cases out of 55,331 lines. Row 10 shows the defects that are pre-released of every Sprint. The flaws that are pre-released are pointed out by the team of production during the development process and can be fixed using an iterative process. The last column shows the total defects that are pre-released being 23. Row 11 shows a total of post-release errors and is pointed out by the customers or product owner during the sprint evaluation meeting, which is 13 in number. The recording of the defects is done in the scrum review meeting, which becomes part of the product backlog, and implementation takes place in the next Sprint. Row 12 shows the quality of the performance of the project. The calculations are done by dividing defects that are post-released by kilo lines of code.

In row 13, sprint 1 has 5, sprint 2 has 4, sprint 3 has three, and sprint 4 has two evaluation suggestion feedbacks. The total number of sprint evaluation feedback suggestions that occurred during the project is 14. The feedback includes post-release defects and also recommendations and new requirements that the customers indicated. Row 14 shows the number of unit tests that are conducted during the testing phase. A total of 789 unit tests were conducted during the project. Row 15 shows the percentage of customer satisfaction level in each of the Sprint, and the last column shows the average of the sprints that shows that an average of 93% of customer satisfaction was achieved. The percentages were obtained by conducting surveys during the implementation of the project. Row 16 shows the productivity team, which is explained as the number of lines coded by the development team in an hour. The average of the product is shown in the last column which is.

Row 16 shows team productivity and is explained as the number of lines coded by the development team in an hour. The average of the project is shown in the last column, which is 60.19 lines per hour. Row 17 shows the time for implementing a user story in each of the sprints. The final column shows the average of all the sprints. The use of the following formula can calculate the time for implementing a user story in any sprint:

$$\begin{aligned}&\textit{Time to implement a user story} \\ &\quad = \textit{Actual work effort} / \textit{No of user stories}\end{aligned}$$

4.1 Availability of quality attributes model

There are many well-known and effectively like McCall Quality Model, FURPS Quality Model, Boehm Quality Model, IEEE Quality Model, ISO 9126–1 Quality Model etc. have been using in the software industries and can be found in the literature [30]. All of previously developed quality model is restricted in evaluating the quality of software applications which have been developed by conventional software models. Whereas agile software development and hybrid agile software development processes are different in real meaning when compared to the conventional software development. Therefore, the quality attributes developed previously may not be suitable in evaluating quality parameters for agile and hybrid agile application development. Quality attributes impact agile and hybrid agile application development process twice. Initially, as the developed hybrid agile application development is an iterative process as discussed in the earlier sections; therefore to develop future iterations product should have certain quality attributes. Finally in the last step, quality of the product is checked after performing all the integrated increments and developing the final product. So in this research, a quality model for hybrid agile model has been developed that can be used to evaluate quality of a product. Following are the quality attributes that are critical for an agile application.

4.2 Case study with HAQPA implementation

As discussed earlier in Fig. 4, HAQPA is introduced to hybrid agile model in three steps which covers availability, scalability, flexibility, performance, usability and

Table 3 Overview of the case study implementing HAQPA process

Project (Man*Month)	18 M/M
Defect density	0.34
Average fixing time	3.4 days (8 h a day work)

understandability. In the first step, during the development of hybrid agile model, HAQPA process is implemented in which the availability of the model is tested as one of the quality parameter. Later, in the second step, where the technical personnel involvement is made mandate by the project head and in that step; scalability flexibility and performance were tested in front of the whole technical team. The errors obtained in this step are corrected simultaneously. Finally, in the last step, customer along with one project team mate checked for the usability and understandability of the product.Table3 shows the project overview with HAQPA implementation for hybrid agile process.

The details of the quality evaluation scale are shown in Table 4.

The evaluation matrix along with overall impact is shown in Table 5. The evaluation parameters and their design, specification values etc. are provided in the table.

Fromm the Fig. 5, it can be said that based on the obtained evaluation matrix, it is evident that understandability, with overall impact of 24 is the attribute that has greatest impact on the quality of developed hybrid agile application followed by performance, usability and availability with an overall impact of 21. Quality attributes evaluation graph has been drawn in Fig. 5, which shows the overall impact of each quality attribute on the developed hybrid agile model.

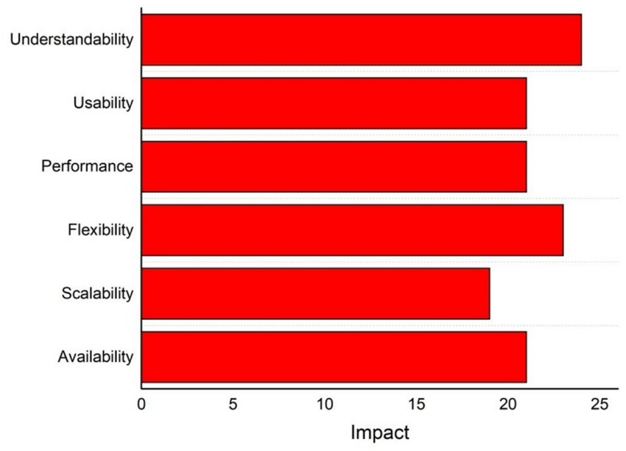

Fig. 5 Overall quality evaluation graph of hybrid agile process

5 Conclusion

The hybrid agile model offers a complete life cycle of software development, but it only works best when utilized in business projects and suited for all forms of project. Scrum and XP overcome the weakness. Additionally, XP does not have planning, considering that it has poor performances for projects that are both medium and large scale. Both Scrum and XP overcomes the weaknesses of XP. Lean software development and XP merged assists in overcoming the weaknesses of Scrum. This paper provides an expansion of a new hybrid agile model that offers a complete life cycle of software development for software development companies. There is also validation with the assistance of a case study and various factors discussed that affect the quality of projects. The results confirm that the hybrid agile model is a model that is more elegant, compact, and powerful when compared to agile methodologies. It shows

Table 4 Quality parameters evaluation scale matrix

Point	1	2	3	4	5
Value	No impact	Little impact	Average impact	High impact	Very high impact

Table 5 Quality Parameters Evaluation Matrix with Overall Impact

	Specification Requirement	Design	Implementation	Testing	Integration	Release	Overall impact
Availability	3	4	2	5	4	3	21
Scalability	2	4	3	4	4	2	19
Flexibility	4	4	4	3	4	4	23
Performance	3	3	4	5	3	3	21
Usability	4	4	3	5	3	3	21
Understandability	4	4	5	4	3	4	24

that the main strengths of the hybrid agile model are the delivery of the project to clients promptly with reduced prices. The developed HAQPA quality attribution model is successfully implemented for the hybrid agile model as obtained better results. The developed model can be used for any software development project as it contains all the attributes. For instance, it is evident that understandability, with overall impact of 24 is the attribute that has greatest impact on the quality of developed hybrid agile application followed by performance, usability and availability with an overall impact of 21. Hence the hybrid agile model is a model that is presented to be used for IoT System projects. The model can be modified to be used in any kind of projects by implementing additional features of XP, Scrum, and lean software development. The requirement specifications of the hybrid agile model can be automated in the future by the use of techniques of artificial intelligence, Machine learning and other projects.

Acknowledgements Our sincere thanks to the experts from software organizations like Wells Fargo, Dell Technologies, Harman Connected Services, L & T Technology Services, North Carolina State Department of Information Technology, Siemens Healthcare Pvt. Ltd., for lending their support and suggestions for successful completion of the work .The authors would also like to thank the management of GPREC for providing us the research facilities and also extending their support for successful completion of the work.

Compliance with ethical standards

Conflict of interest The author declares that there are no conflicts of interests regarding the publication of this paper.

References

1. Pahl GWB, Feldhusen J, Grote K-H (2007) Engineering design. Springer, New York
2. Boehm BW (1988) A spiral model of software development and enhancement. Computer 21(5):61–72
3. Krishnan P, Duttagupta S, Achuthan K (2019) VARMAN: multi-plane security framework for software defined networks. Comput Commun 148:215–239
4. Greenfield J, and Short K (2003) Software factories: assembling applications with patterns, models, frameworks and tools. Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications pp. 16–27
5. Abrahamsson P, Warsta J, Siponen MT, and Ronkainen J (2003) New directions on agile methods: a comparative analysis. International Conference on Software Engineering, 2003 Proceedings pp. 244–254
6. Cockburn A (2004) Crystal clear: a human-powered methodology for small teams: a human-powered methodology for small teams. Pearson Education, London, pp 1–313
7. Lalband N, Kavitha D (2019) Software engineering for smart healthcare applications. Int J Innov Technol Explor Eng 8:325–331
8. Awad MA (2005) A comparison between agile and traditional software development methodologies. University of Western Australia, Crawley, pp 1–84
9. Cohn M, Ford D (2003) Introducing an agile process to an organization [software development]. Computer 36(6):74–78
10. Farhan S, Tauseef H, Fahiem MA (2009) Adding agility to architecture tradeoff analysis method for mapping on crystal. IEEE WRI World Congress on Softw Eng 4:121–125
11. Abrahamsson P, Babar MA, Kruchten P (2010) Agility and architecture: can they coexist? IEEE Softw 27(2):16–22
12. Lattanze AJ (2005) The architecture centric development method school of computer science [Institute for software research international. Carnegie Mellon University, Pittsburgh, pp 1–57
13. Ambler S (2002) Agile modeling: effective practices for extreme programming and the unified process. Wiley, Amsterdam
14. Livermore JA (2008) Factors that significantly Impact the Implementation of an agile software development methodology. JSW 3(4):31–36
15. Pressman RS (2005) Software engineering: a practitioner's approach. Palgrave macmillan, UK, pp 1–402
16. Mushtaq Z, Qureshi MRJ (2012) Novel hybrid model: integrating scrum and XP. Int J Inf Technol Computer Sci (IJITCS) 4(6):39–44
17. Boehm B (2002) Get ready for agile methods, with care. Computer 35(1):64–69
18. Fowler M (2000) Put your process on a diet-as a reaction to cumbersome approaches to development, new methodologies have appeared. these methods attempt a compromise between no process and too much process. Softw Dev 8(12):32–39
19. Abrahamsson P, Salo O, Ronkainen J, Warsta J (2017) Agile software development methods: review and analysis. arXiv preprint arXiv:1709.08439
20. Schwaber K, Beedle M (2002) Agile software development with scrum vol 1. Prentice Hall, Upper Saddle River, pp 1–53
21. Beck K (2000) Extreme programming explained: embrace change. Addison-wesley professional, Boston, pp 1–24
22. Syed-Abdullah S, Holcombe M, Karn J, Thompson C. The study of an agile methodology in action: a qualitative approach, vol 10, pp 1–9
23. Sfetsos P, Angelis L, Stamelos I, and Bleris GL (2004) Evaluating the Extreme Programming System–An Empirical Study. In International Conference on Extreme Programming and Agile Processes in Software Engineering, pp. 227–230
24. Poppendieck M, Poppendieck T (2003) Lean software development: an agile toolkit. Addison-Wesley, Boston
25. Grieves M (2005) Product lifecycle management: driving the next generation of lean thinking: driving the next generation of lean thinking: driving the next generation of lean thinking. McGraw Hill Professional, NY
26. DSDM Consortium, Dynamic Systems Development Method, version 3. Ashford, Eng.: DSDM Consortium, 1997

27. Cockburn A (2000) Writing effective use cases the crystal collection for software professionals. Addison-Wesley Professional, Boston
28. Coad P, Lefebvre E, Luca JD (1999) Java modeling in color with uml: enterprise components and process. Cap 6:182–203
29. Baskerville R, Levine L, Pries-Heje J, Slaughter S (2001) How internet software companies negotiate quality. Computer 34(5):51–57
30. Malik MU, Nasir H, Javed A (2014) An efficient objective quality model for agile application development. Int J Computer Appl 85(8):19–24