Agile Methodologies in Software Maintenance: A Systematic Review

Sandhya Tarwani
USICT, Guru Gobind Singh Indraprastha University
Sector-16C, Delhi-110078
E-mail: sandhya.tarwani@gmail.com

Anuradha Chug
USICT, Guru Gobind Singh Indraprastha University
Sector-16C, Delhi-110078
E-mail: anuradha@ipu.ac.in

Keywords: Agile methodology, scrum, extreme programming, software maintenance, quality

Received: November 8, 2016

Agile Methodologies has been gaining popularity since 2000. The Software Maintenance phase of software lifecycle is the most expensive and tedious in nature and use of Agile methodologies helps in maintaining software over time in flexible and iterative manner. This study reviews several papers with different case studies to evaluate the performance and quality of software using agile methodologies. In this study, more than 30 research studies are investigated which are conducted between 2001 and 2015 and have been categorized according to the publication year, datasets, tools, type of techniques etc. This will be the first review paper on the use of the Agile in software maintenance which will help the researchers and encourages companies and beginners to adopt these methodologies to gain software quality. It can be concluded that by adopting agile methodologies it is guaranteed that there will be continuous improvement, greater productivity and enhanced quality of the software. It will also help software development team to finish their work within real time constraints. This study would be helpful to professional academicians also so that they can identify the current trends and future gaps in the field of agile methodologies.

Povzetek: Podan je pregled agilnih metodologij za vzdrževanje programske opreme.

1 Introduction

Software maintenance is the most expensive phase of software development lifecycle. The maximum share of total project costs is being used to maintain software. Achieving the quality of software desired becomes difficult for developers as they often overstep budgetary constraints. Therefore, there is need to find appropriate solution that has the ability to minimize costs. Initially waterfall was used which was sequential in nature and this methodology dominated the world for longest period of time. This model was first cited by Winston W. Royce in 1970[1] when he divided the software development lifecycle in seven sequential and linear stages: Conception, Initiation, Analysis, Design, Construction, Testing and Maintenance. In the early days, waterfall was adopted by various large and small organizations. However, it is important to note that the model inherently has some drawbacks which includes, up-front requirements that increase the cost of maintenance as it become difficult to further change the software. Using this traditional model [2], 70% of the software could not achieve their objective. In a nutshell, the cost of maintenance phase has been tremendously increased as waterfall is sequential in nature making it difficult to move back to the previous stage in the course of maintenance.

Due to all these drawbacks, many organizations are moving towards agile methods for software process improvement. Agile Methodologies were first introduced in the Agile Manifesto [3] which was a summary written and signed in 2001 by Kent Beck also known as Agile Visionary.

Agile methods have gained tremendous success in the commercial industry since late 90's because of following advantages:

- They have up-front requirements.
- It focuses on the developers and customers relationship.
- It includes iterations so that product quality and performance enhances.
- It is iterative in nature which helps the organization to maintain their software in a more flexible and concise manner.
- Agile releases short prototypes after release planning so that users can review it and this continuous monitoring by users help in the maintenance phase.

As various companies had been using waterfall for software maintenance, it was difficult in the early phase to persuade their teams to adopt agile techniques. Wipro technologies were one of those to adopt agile methods. Initially it was difficult to change the mindset of team members but later as the advantages of agile methods came into focus, it becomes their prime focus.[4]. With the advancements in the technology, almost every organization, large and small, is adopting agile at their pace. Customers are more satisfied as the maintenance work consumes lesser time and cost as well as quality of the product has been enhanced. With the rapidness in the product delivery, the transition of maintenance from waterfall to agile methodology environment is increasingly faster. The use of extreme programming in the maintenance environment by Iona technologies [5], showed that by adopting its practices fully or partially in their Orbix project, there was an improvement of 67% although the team size reduced. Visibility was also a factor which plays an important motivational factor in this turnaround.

The aim of this systematic literature review is to summarize, analyze, plan and learn the following things:

(1) Various Agile methodologies for better performance in software maintenance
(2) Comparison of waterfall model and agile methodology lifecycle
(3) The switch from waterfall model to agile methodologies
(4) Various tools available for Agile methodologies
(5) Summarize the strength and weaknesses of Agile Methodologies.

Furthermore, there is a provision of future directions for practitioners.

The rest of the paper is organized as follows: Section 2 presents the research questions and the research criteria for the selection of the studies. It also provides an analysis of the number of research papers available per research questions. Section 3 provides the answer to the research questions identified in this literature survey. Section 4 provides the conclusion and future directions obtained from the systematic survey.

2 Review process

The planning, monitoring and reporting of the systematic literature review paper has been done as per the guidelines given by Kitchenham [6]. As shown in Fig. 1, planning has been done initially to identify the need of this literature review. A review was carried out in order to analyze the work in software maintainability with the help of Agile methodologies. Research studies with respect to their years, case studies, practices, tools used etc, have been investigated so that a trend can be established to find out the pace in this research area. During our survey, it was noticed that papers can be categorized on the basis of year, datasets, tools, techniques, etc. For example, many researchers use private datasets in their studies which make it difficult for the comparison of their performance. After figuring out the needs, keywords were searched for the formation of literature review of Agile methodologies in software maintenance. This was an important step as this is the first review paper in this field.

In the next step, the process of including and excluding case studies was identified. The fourth step involved the formation of the research questions that are being involved and answered in this literature review. In the fifth step, we have analyzed the data.

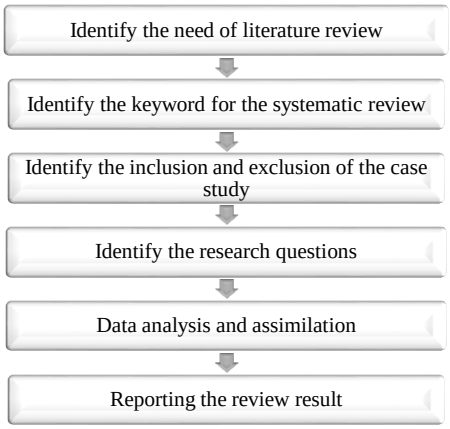

Figure 1: Systematic review process.

2.1 Search keywords strategy

We formed the search terms by using the Boolean expression ‘OR’ and combining main search terms using ‘AND’. The following general search terms were used to collect the data and to form the basis of this literature review:

Agile AND (software OR development OR tool OR testing) AND (XP OR scrum OR lean) AND software (Maintenance OR Maintainability OR Quality OR complexity) AND (quality factors OR reliability OR effects OR refactoring OR metrics). In the table 1, subject along with the search terms are available which can be used to search the various papers including this research review.

Subject	Search terms
Agile methodologies	Agile software, agile development, agile tools, agile testing, XP agile case, agile in small medium companies, agile scrum, agile in software maintainability, extreme programming effects
Software maintenance	Software maintenance, software quality, software complexity, software reliability, software maintenance maturity level, quality factors, refactoring, metrics

Table 1: Search term along with the subjects.

The terms on agile methodologies in software maintenance were derived from textbooks and various research papers. After the identification of the search terms, digital portals were selected and were not restricted only at the home university. The primary source of the literature survey is Google scholars which extracted data from various databases including IEEE Xplore, Wiley online library, ICSR, Science digest, SpringerLink, World Scientific and Digital library.

2.2 Inclusion and exclusion of the study

After identifying the search terms, primary study needed to be selected. There are various studies available in these fields but we needed to apply the inclusion and exclusion so that only important and primary studies would be looked at for the purpose of this review. 30 primary case studies we considered for this review process. The studies were selected after following the inclusion and exclusion criteria given below:

Inclusion criteria:

- Empirical studies using the agile methodologies.
- Empirical study comparing the waterfall and agile methodologies.
- Empirical study combining agile methodologies and Data mining.
- Empirical study using extreme programming, scrum and test driven development.

Exclusion criteria:

- Studies without empirical results of agile methodologies.
- Review studies.
- Web links
- Studies without validation of data.

These inclusion and exclusion criterion helped in the identifying our 30 primary case studies.

2.3 Research Questions

The main focus on the systematical literature review is to answer some of questions which were raised. Table 2 presents 10 research questions addressed during the course of review survey. Firstly, the basic definition of the agile methodologies was identified (RQ1). The second question explains the strengths and weaknesses of agile methodologies (RQ2). RQ3, RQ4 and RQ5 explain the most dominant journal, kind of dataset used and percentage of publications during these years respectively. The issue of transition from waterfall to agile methodologies has been raised in RQ6. The different type of agile methodologies has been analyzed in RQ7. The improvements in software maintenance using extreme programming has being identified in RQ8 and the suitable project size along with the various tools available in market for agile methodologies have been discussed in RQ9 and RQ10.

Research Question	**Main Motivation**
RQ1: What is Agile software development?	Identify the definition of agile in software development
RQ2: What are the strengths and weaknesses of agile methodologies?	Identify the importance of agile methodologies along with their limitations
RQ3: Which journal is dominant in software maintainability using agile methodology?	Identify the most important agile methodologies and its effect on the software maintenance journal
RQ4: What kind of datasets are the most used in various journals?	Identify whether private or public datasets are being used by the researches
RQ5: What is the percentage of publications published during these years?	Identify whether paper published during these years represent large portion of papers in literature or not
RQ6: How does Software Maintenance Team switched from Waterfall to Agile?	Identify team progress from traditional sequential model to more iterative model
RQ7: What are the various sub parts of agile methodologies?	Identify different types of agile methodologies in use today
RQ8: How Extreme Programming practices help to improve performance of software maintenance?	Identify the extent of using extreme programming helps in improving performance
RQ9: Which size project is suitable for the Agile methodologies?	Identify the complexity of introducing agile methods in large, small and medium size
RQ10: What are the various tools available for agile methodologies?	Identify whether the tools available are open or commercial

Table 2: Research questions.

3 Review results

3.1 Agile software development (RQ1)

When Kent Beck investigated the cost of change curve of Barry Boehm [7], Agile Visionary, he observed that curve was more flattened in case of Agile[8,9]. Agile methods are iterative in nature. They uses design-code-test loop which is implemented once a day. Agile mainly balances four variables: Cost, Schedule, Requirements

and Quality. Velocity is also being introduced which is the amount of effort calculated.

Software development is a very tedious job because of evolving technology and there is always a need to develop high quality product [10]. Therefore, Agile methodologies were introduced which minimizes development life cycle. Agile methodologies have various advantages and are easy to learn and implement. Its most important advantage is its light weight characteristic which mainly concentrates on the delivery of high quality product. Extreme programming, one of the most acceptable and widely used agile methodologies, helps the small team organization to change requirements, tight schedules and meet high quality demands [11].

Agile methodology when mapped with the complex adaptive systems and its three dimensions results in the best possible practices [12]. The three dimensions are people, process and product which are not completely independent from each other and hence require identification of all metrics incorporated in all three dimensions. Agile methodology mainly focuses on the rapid iterations and small releases [13] so that users can bring change requirement to notice more rapidly. Serena [13] describes these two methodologies: extreme programming which focuses on the development aspect of software lifecycle rather than managerial aspect and scrum which has its focus on both managerial and development aspects.

Agile architecture can be divided into Product owners and sprintable form [14]. The product owners consist of Up front planning in which architecture is being designed and Story boarding structures the business need. Sprintable form has sprints which build the working software and its functionality by conducting the meetings which reviews and delivers software on time, very frequently. The most important factor influencing agile adoption was personal initiative [15]. As agile requires up front gathering of the requirements therefore turnaround time, software complexity and stability of requirements help in the decision of using agile approaches in business environment. With its challenges and limitations, agile software development has great future scope.

There is always conflict between the formal methods and agile software developments methods [16] because of lack of communication and understanding. Therefore interaction is needed to extract the best practices from both methods.

3.2 Strengths and weaknesses of agile methodologies (RQ2)

The main strength of agile due to which it had gained popularity over traditional and sequential waterfall model is that it is based on the concept of iterations [17]. The user will be able to get the working version of their respective project after each iteration. Based on this it becomes easy for the user to add requirements during the development phase and hence it enhances the flexibility. Even after the design phase has started and user wants to add or change the requirement, they can do so. This is what differentiates it from the waterfall model. In waterfall model, all the requirements have to be submitted at the start of project only. Testing is very important phase of the software life cycle and it needs to be done on a regular basis. With agile methodologies, it becomes easy to continuously integrate and test after every iterations as this method has the provision of continuous integration and constant testing. Developers are in direct contact with the customer which helps them understand the project placing communication at the centre of agile methodologies. With the involvement of the customer, teams inevitably gain motivation and this goes on to enhance the quality [18]. As the number of people in agile team is small, therefore coordination among the team members is easy. Main reason for the introduction of the agile methodologies is that the project needs to be completed on time and stay within the allocated budget something which is granted to the use of this solution.

Figure 2: Strengths of agile methodologies.

Although it has been established that agile methodologies comes with lots of advantages, but it has various weaknesses which must be considered before going head first into software development, so that the quality is not compromised. The major advantage of agile methodologies is the active participation of the customer or user throughout the development lifecycle but this can also leads to the major weaknesses [17]. Sometimes Customer do not have the time to interact. Agile methodologies generally use small teams to develop their projects which can sometimes make it challenging to complete large projects. Team members need to be at the same location throughout their work, but this can get difficult as it is not possible for those teams that work on the different projects and are far away from each other to come together and work at the same physical location. This makes coordination difficult. Also as requirements can be added any time, this will lead to the never ending project.

Miscommunication is the major factor that leads to the problem during the implementation of the agile methodologies in the software development lifecycle. Testing is conducted throughout the software development lifecycle therefore it requires the testers to be at the same place during the lifespan of the project development. This will unnecessarily increase the

resources of the project and increases the overall cost [19]. It becomes difficult to find the pace for the software development. The overall weaknesses of agile are summarized in fig. 3.

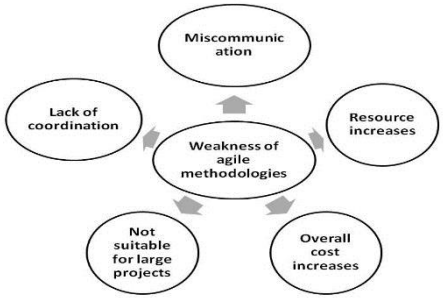

Figure 3: Weaknesses of agile methodologies.

3.3 Dominant journals in this research field (RQ3)

We used more than 35 research studies on software maintainability using agile methodologies. The most dominant research studies along with their ranks are categorised in Table 3.

Rank	Journal	Author
1	IEEE Transactions on Software Engineering	Poole [1]
2	International Journal of computer application	Agarwal [20]
3	International Journal of database theory and application	Upadhyay [21]
4	International Journal of computer Applications	Kumar [22]

Table 3: Most important software maintainability journals using agile methodologies.

3.4 Analysis of datasets used (RQ4)

The biggest difficulty faced during this analysis was the use of unknown and private data sets. Many of the research studies have been written by private firms that used their proprietary data from the analysis work. Papers have been divided into four subsections according to the type of data used for the analysis work: public, private, virtual and unknown.

Public datasets were mostly extracted from the interviews and questionnaire. These includes various case studies like student scientist, maintenance managers etc. They are located at CVS (Concurrent version system) repositories. Various companies volunteered for the analysis providing their projects and case studies and associated private datasets belong to these private companies and not available publicly which includes CSoft developed by Norwegian Software Company[33], projects from Samsung electronics, Orbix projects developed at Iona technologies[5] etc. Virtual datasets have been created by the researchers on their own so as to provide an analysis and proper understanding of the topic. These have been created by SPEM tool and EPF composer editor [23]. These are not included in the public repositories. If no information is available about the datasets, then they have been classified as unknown datasets. As shown in Fig. 4, 34% of papers utilise private datasets. This is what makes them not repeatable and verifiable. On the other hand, 43% of papers have used public datasets. And Only 13% have used virtual datasets.

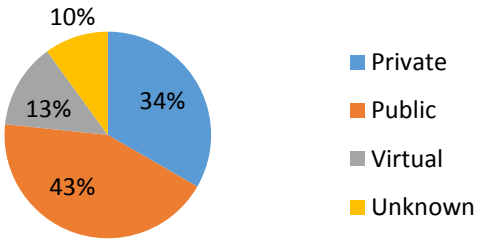

Figure 4: Distribution of datasets.

In table 4, all the information about the case studies and papers considered is provided that have used either private or public or virtual or unknown datasets.

Type of Datasets	Number of paper	Author name
Private	11	Poole [5], Hayes [24], Llieva [11], Szalvay [2], Zanker [25], Serena [13], Sureshchandra [4], Succi [26], Jeanette [37], Jeon [28], Dagnino [10], Hanssen [33],
Public	13	Reyes [29], Abrahamsson [30], Vijayasarathy [15], Saiedian [31], Christensen [32], Mattson [34], Hinchey [16], Thong [35], Mirza [36], Knipper [37], Chakka [38], Qureshi [39]
Virtual	4	Svensson [40], Singh [41], Beeson [42], Piattini [23]
Unknown	3	Huo [43], Jakobsen [44], Choudhari [45]

Table 4: Papers per datasets used.

3.5 Distribution of papers (RQ5)

We examined papers according to their publication year. Papers have been classified into two groups: Paper published before 2008 and paper published after 2008. Fig. 5 shows the distribution of papers regarding publication date.

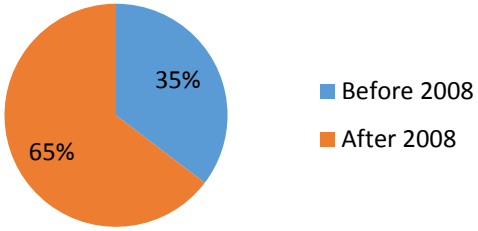

Figure 5: Distribution of papers.

Sixteen papers have been published before year 2008 and twenty nine papers have been published after that. Maximum papers are being published in 2008, IEEE Agile Conference. Fig. 6 shows the type of papers which were published till now. It can be categorized into three groups: journals, book chapters and conference.

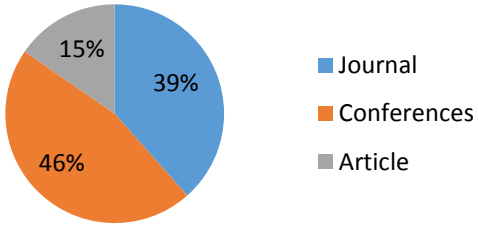

Figure 6: Distribution of type of paper.

3.6 Switching from waterfall to agile (RQ6)

Before 2000, software companies found it convenient to use the traditional waterfall method for the development of software. Due to its shortcomings, the development of software can often not completed on time. Even the maintenance cost associated with the use of this method was increasing. To address these issues, Agile methods in which all the phases overlap and the requirements are gathered in an iterative manner, were introduced. Under this methodology, all the requirements are reexamined at the beginning after each iteration. This minimizes the shortcomings of waterfall model and hence improves the software development process and is more cost efficient. With agile, maintaining software becomes quite easy which enhances the quality as well as reduces the cost. Agile is based on its four factors which include: Cost, Schedule, Requirements and Quality.

The biggest challenge in the world of software development was the conversion of waterfall team into an agile one. The mindset of teams had been set and it was becomes difficult to steer them away from traditional to more modern methods [4]. Basically, the entire project is divided into iterations and product is released iteration by iteration. These projects have phases which are implemented weekly. In the first week, team focuses mainly on the analysis and design. Second week consist of designing and unit testing. The subsequent third and fourth week consists of coding, unit and integration testing of the system. Fifth and sixth phase also involve integration and unit testing of the system. Prior to the start of every phase and iteration, testing is intrinsically considered important. However, what was clear that making the team's bend more towards agile methodologies could not be done without prior planning. Team members who were not able to feel comfortable in this new version of developing software could not continue to be included in the team.

Agile methods provide a faster delivery of product in short span of time and ensure high level of software quality at the same time. This is what plays a crucial role in preferably [43]. With agile practices, the quality of the software also enhances. Agile methods mainly relay on the feedback from the onsite customer who is involved throughout the development process. Pair programming refactoring is used continuously to enhance the productivity, an upgrade from the waterfall method. To check on the quality process, acceptance testing is being used regularly to achieve results successfully. Fig. 7 shows the comparison between waterfall and agile lifecycle methods.

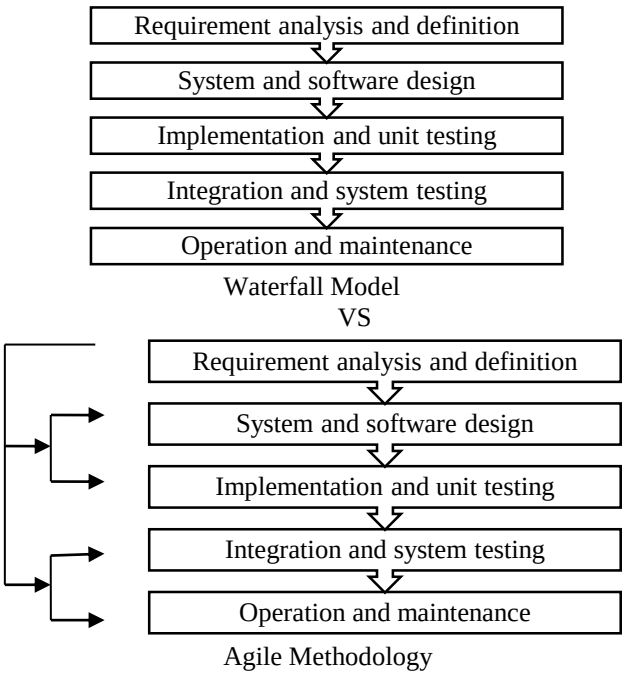

Figure 7: Comparison of waterfall and agile methodologies.

3.7 Types of agile methodologies (RQ7)

Many agile methodologies have been developed so far which helps in developing and maintaining the software at a lower cost. Fig. 8 shows the type of agile methodologies along with their founder name.

Extreme programming

Extreme programming is one of the most widely adopted agile methodologies which were created by Kent Beck. It primarily focuses on the development phase rather than the managerial aspect of software projects [13]. XP was mainly designed so that companies and firms could

comfortably accept some of the agile methodologies. A release plan is developed initially. User writes user stories to describe what they want and is part of the developer team. This ensures that all the requirements are being added in accordance and in presence of user. Team breaks the task into iterations and at the end of it; acceptance testing is being performed to satisfy the user.

Scrum

Scrum was applied in 1990's by Ken Schwaber and Mike Beedle. It is agile method which is incremental and iterative and focuses not only on development but at managerial aspects also[13, 20]. In scrum, work is divided into cycles of work called sprints. During each sprint, requirements are prioritised and are also called as user stories. This is done to develop the highest value requirement first for the user[46].

Test driven development

Test driven development relies on the repetition of very short development cycle. Test cases are being generated to provide improvement and limited code is generated to pass the test successfully so that refactoring can be done easily and code can be sent for the acceptable standard[31]. So therefore it is a quality-first approach where developers test cases are written before the functional code itself [2].

Lean

Lean is a production practice that primarily focuses on the expenditure of resourses. It is mainly used to preserve value for the end users who consumes a product or service with less work. Dynamic system development methodology gained popularity to provide a standard for agile framework that was called as Rapid application development. It revolves around the nine principles that includes active user involvement, frequent delivery, integrated testing etc.

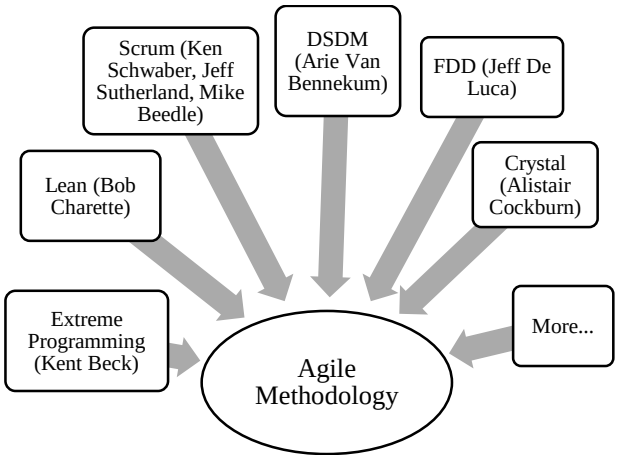

Figure 8: Types of agile methodologies.

Crystal

Crystal is the most light weight agile methodology that consist of agile family such as crystal clear, crystal orange which can be characterised according to the team size and priority.

3.8 Extreme programming practices (RQ8)

Extreme programming is the most successful agile methodology that focuses on the development aspect. XP consists of various practices that help in improving the software in a maintenance phase. Companies adopt full or partial practices of extreme programming to improve the quality of their software. Iona technologies partially use extreme programming practices in their environment [5]. Planning game and simple design is adopted partially. There were small releases of product which provides good productivity and enables the completion of the project within time and budget which was followed religiously. Pair programming in which two developers come together to write code was sparingly adopted. All other practices are fully adopted from starting. Because of the use of extreme programming practices, 67% of improvement is witnessed along with the improvement in the visibility.

Analysis says that when extreme programming practices along with the personal software experts i.e., eXPERT approach is used then there has being significant improvement in productivity, defect rate and effort spent [11]. The introduction of extreme programming in maintenance environment has a positive impact on the project [40]. All the twelve practices could not be able to introduce successfully and those which are assimilated into the project are adapted according to development team environment. Pair programming is one of the most important principles of extreme programming [10]. Pair programming contrast some of the results of empirical evidence as pair programming style has no higher productivity and in many cases it is low in coding standards. Therefore, it is not always necessary to have positive results from the extreme programming principles.

3.9 Suitable project size for agile methodologies (RQ9)

Small companies have a prime focus on the maintenance process; hence Agile_MANTEMA [23] was introduced to help small organizations in the maintenance of their product and to provide services to the customer. Extreme programming was traditionally used in the small organizations but was later extended to be used in medium and large organization [39]. Postmortem analysis and fault rate per KLOC is what is used as a basis for the comparison of traditional and extended extreme programming. The quality of the extended extreme programming is much better because of less fault rate per KLOC.

Agile practices are more easily adapted in less complex organizations [40]. As complexity increases, it becomes mandatory to redesign various methodologies and practices in order to fit in the existing environment.

3.10 Tools available (RQ10)

There are various tools available for agile software development in the industry. Open sources as well as commercial tools are available that helps in the proper implementation of all the advantages of agile methodologies. Free or open sources are agilefant, agile manager, fire scrum, ICEscrum, LeanKit KAnban, Xplanner, etc. These helps the small and medium companies to use these tools and figure out the burndown chart, to make user stories and to estimate the effort left.

Some of the description of open source tools is provided in Table 5.

Tools	Description
Agilefant	With agilefant, management work improves by formation of project burnup and iteration burndown chart. These tools provide three levels of backlogs i.e., product, project and iterations.
Agile manager	It provides designing management also along with the management work.
Fire Scrum	It is based on rich internet application which helps in project management.
ICEscrum	It's a J2EE based tool that helps in the scrum management.
LeanKit	It helps in the customer value and satisfaction.
Xplanner	It is a web based planning and tracking tool and implements with the help of java, jsp etc.

Table 5: Distribution of open tools.

Many companies have built their own agile software which are available to others as well but are paid. Hence, vendors have to buy them according to their needs and demands. These help the organization to make improvements and help others to avail the benefits of these products. These tools are version One, Agile Log, Agilo, ExtremePlanner, etc. Some of the description of these tools is provided in Table 6.

Tools	Description
Version One	It is a simple project management tool which focuses on the centralised version of the management.
Agile Log	It is loaded with tool to effectively manage project and efficiently manages cost.
Agilo	It is a robust platform for managing project.
ExtremePlanner	It has east to use interface that helps the teams to coordinate among themselves easily.

Table 6: Distribution of proprietary tools.

In order to explore further the features of open and proprietary tools, one example of each type is taken and compare in Table 7.

Feature	Agilefant	Agilo
Platform	It works on the Java and MYSQL	It works on Python and RDBMS
Ranking vs prioritization	Priority of 1-5 scale is being given to user stories.	Ranks as well as drag and drop function is available here
Story points	This functionality is not available	It is available here
Iteration burn down chart	Very poor functionality of the chart which decreases the motivation of team	Chart is developed in a good manner, hence enhanced motivation
Epics	Many user stories cannot be encompassed due to lack of this feature	User stories can be encompassed
Portfolio management	Real time overview of budget as well as progress of the project is possible	This feature is not available here
Story themes	It is available in the form of story labels	This feature is not available here
User role	No direct interaction as well as involvement of user	Users are part of teams
Reports	They are available in timesheets only	It saves the reports in customized query
Pros	Story hierarchy is available along with good iteration planning.	It delivers robust platform for the team and helps in coordination with the help of Scrum-teams
Cons	Customization is very poor and external systems integration is not possible	Management of people along with the user interface is not possible

Table 7: Comparison of open source(agilefant) vs proprietary(agilo) tools for implementing the agile methodologies.

4 Analysis

Sixteen journals and twenty four conference proceedings have been evaluated in this review. These were published during the years 2001 to 2015. Fig. 9 shows the curve for a particular year and number of publications developed during its course. It plots year on y-axis and number of publications on x-axis. After reviewing the papers, it was found that maximum papers are published in 2008. There were gradual increases in the beginning of the time period selected till 2008, after which the slope declined for years 2009 and 2010. To date, the work is being continually pursued in this research field to bring improvements in the performance.

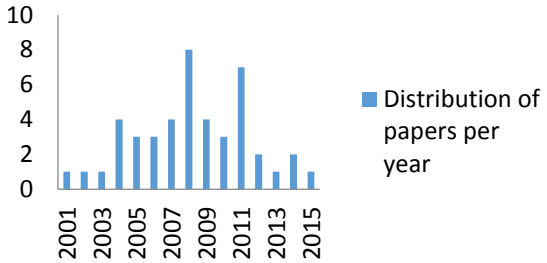

Figure 9: Distribution of papers per year in review.

Fig. 10 and Table 8, describes the number of papers which were used in explaining the answers for the research questions. It is clear that maximum papers have been published to describe what actually agile software development is. Initially, it was stated that design-code-test loop is implemented daily [2]. Agile support high quality delivery of product with upfront gathering of requirements [10]. Small team uses extreme programming in time constraints project [11]. Complex adaptive systems along with the agile software development help in identification of metrics that provides benefits [12]. Serena industries integrated with the agile software development to describe extreme programming and crystals [13]. Agile architecture interactions help in the delivery of working software on time [14]. The factor influencing agile usage and adoption is explained in detail [15]. The basic integration problem between the formal methods and agile methods are explained and extraction of best is done for best practices [16].

[5] Describes to the most dominant paper in this field. Public and private datasets are the closest in number in these research papers. 41% of data used belongs to the private datasets therefore it becomes difficult to compare various case studies. There are various different papers published during these years. 65% of the papers are being published after 2008. Waterfall was used initially but because of these shortcomings, agile was introduced [2]. With agile, quality as well as productivity enhances. The mindset of people was set that's why it was difficult to switch from traditional to iterative methods [4]. Agile provides better productivity and quality to the software as compare to waterfall model [43].

Extreme programming was one of the most acceptable methodologies of agile software development [13]. User stories and acceptance testing is done to provide the better productivity and quality. In scrum methodologies, sprints are being introduced in which daily meetings are held to discuss various requirements [46]. With test driven development, refactoring becomes an easy job [2, 31]. Different extreme programming practices like pair programming, continuous integration etc. are being used fully or partially to improve the productivity [5, 11, 40, 10]. Agile_MANTEMA was introduced to provide benefits to the small companies [23]. Extreme programming was initially meant for small companies but later it was extended for medium and large organization [39]. Less complex systems also makes use agile methodologies [40].

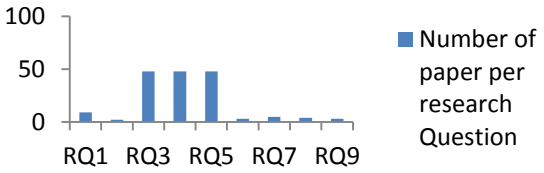

Figure 10: Number of research paper per research question.

Research questions	**Number of papers**	**Authors**
RQ1	9	The Agile Manifesto[3], Szalvay[2], Ilieva [11], Meso[12], Serena [13], Vijayasarathy [15], Hinchey [16], Dagnino[10], Madison [14]
RQ2	2	Mohammad[17], Koch [18]
RQ3	All are included	All are included
RQ4	All are included	All are included
RQ5	All are included	All are included
RQ6	3	Szalvay [2], Sureshchandra[4], Huo[43]
RQ7	5	Serena[13], Majumdar [20], Meng*[46]*, Saiedian [31], Szalvay [2]
RQ8	4	Huisman [5], Ilieva[11], Svensson [40], Dagnino [10]
RQ9	3	Svensson[40], Piattini [23], Qureshi [39]

Table 8: Number of papers per research questions.

5 Conclusion and future directions

Our current survey study is the review of 30 research studies. After observing the evidences from the research studies, it was observed that by introducing agile software development methodologies there has been a continuous improvement in the field of software development. Various methodologies have been used and practiced by practitioners. Agile uses product backlog, sprint backlog and carries work in iterations. The small products are released after every iteration to help the customers to add more requirements according with their needs. As maintenance is very tedious job and is the most expensive phase of the software lifecycle, this has always been a concern under the traditional waterfall approach and is something that's the introduction of agile methodology has addressed in terms of visibly reduced cost.

This helps the organizations to minimize the cost and concentrate on the provision of greater productivity and quality. Extreme programming is the most practiced and used methodology and provides productivity not only in small but also in medium as well as large organizations. There is, however, more research that is required in this field to provide clear path of implementation of agile methodologies in software maintenance.

Some of the future works which can be done in this field are:

- As per the analysis, author's observed that improvement in the pair programming will help the programmer to make up for theory lack of training. Although this is an advantage but still it needs to be incorporated in a company so that it become part of its fabric.
- In future we are planning to compare the quality of a product that can be achieved through the use of waterfall alongside that of agile methodologies.
- To the best of author's knowledge, analysis of detailed metrics should be done with the help of agile methodologies and this analysis could be extended to consider not only the number of defects but also severity.
- There is a strong need for the Private case studies to be replicated with the general cases so that results can be verified.
- Authors are also planning to focus on the refinement of the extreme programming process model with the help of different case studies.
- As far as the author's knowledge is concerned, Comparison of the number of hours required for maintaining the software by the developers using agile as well as some traditional lifecycle modeling has not yet conducted.
- There is a strong need for creation of Automated tools for agile which can be prepared for future refinements in the projects.
- The Authors observed that formalized validation of the data is needed needs so that projects can be validated easily.

References

[1] Walker W. Royce, "Managing the Development of Large software system" *in Proc. IEEE WESTCON*, Los Angeles, IEEE Computer Society Press, 1970, pp. *328-338*.

[2] Victor Szalvay (2004). *An Introduction to Agile Software Development* [Online]. Available: http://www.danube.com/system/files/CollabNet_IntroToAgile_wp_0710.pdf

[3] Kent Beck et al. *Mainifesto for Agile Software Development* [Online]. Available: http://www.agilemanifesto.org/

[4] K. Sureshchandra and J. Shrinivasavadhani, "Moving from Waterfall to Agile" in Agile Conference, [2008] © IEEE. doi: 10.1109/Agile.2008.49

[5] C. Poole and J.W. Huisman, "Using Extreme Programming in a Maintenance Environment," *Proc. IEEE J. Software* vol. 18, Issue 6, pp. 42-50, Nov. 2001.

[6] B. Kitchenham et al., "Guidelines for performing systematic literature review in software engineering," © 2008 Elsevier B.V, doi:10.1016/j.infsof.2008.09.009

[7] B.W. Boehm, in *Software Engineering Economics*, 1st ed. USA: Prentice Hall PTR Upper Saddle River, NJ, 1981.

[8] K.Beck, in *Extreme Programming explained: embrace change*, USA: Addison-Wesley Longman Publishing Co., Inc. Boston, MA, 2000.

[9] S.W. Ambler ," Examining the cost of change curve," in *The Object Primer: Agile Model-Driven Development with UML 2.0*, 3rd ed. USA: Cambridge University Press, 2004.

[10] A.Dadnino ,"An Evolutionary lifecycle Model with Agile practices for software development at ABB" in *ICECCS '02 Proc. 8th Int. Conf. on Engineering of complex computer systems*,© IEEE computer Society, USA, pp. 215.

[11] S. IIieva et al., "Analyses of an agile methodology implementation," in Proc. *30th EUROMICRO Conference*, ©IEEE, 2004, pp. 326-333.

[12] P. Meso, "Agile Software Development: Adaptive Systems principles and Best practices," *Information System Management*, doi: 10.1201/1078.10580530/46108.23.3.20060601/93704.3.

[13] Serena software Inc., *An introduction to Agile Software Development*,[Online], Available: http://www.serena.com/docs/repository/solutions/intro-to-agile-devel.pdf.

[14] J. Madison, " Agile Architecture Interactions," *IEEE Software*, ©IEEE Computer Society,[2010], Vol. 27, no. 2, doi: http://doi.ieeecomputersociety.org/ 10.1109/MS.2010.27.

[15] L.R. Vijayasarathy and D.Turk, "Agile Software Development: A survey of early adopter," *Journal of Information Technology Management*, Vol. 11, no. 2, 2008, pp. 1-8.

[16] S. Black et al., "Formal Versus Agile: Survival of the fittest,", ©IEEE Computer Society, [2009], Vol. 42, no. 09, pp. 37-45.

[17] A.H. Mohammad et al.," Agile Software Methodologies: Strength and Weakness,", *Int. J. of Engineering Science and Technology*, Vol. 5, No. 03, March 2003, pp. 455-459.

[18] A.Koch, " 12 Advantages of Agile Software Development,", ©Global knowledge Training LLC. , [2011], pp. 1-10.

[19] K. Waters, Disadvantages of Agile Development [Online], Available: http://www.allaboutagile.com/disadvantages-of-agile-development/.

[20] M. Agarwal and R. Majumdar," Software Maintainability and Usability in Agile Environment,", *Int. J. of Computer Application*, Vol. 68, No. 4, 2013, pp. 30-36.

[21] P. Upadhyay," Modeling software Maintainibility and Quality Insurance in Agile Environment,", Int. J. of Database Theory and Application, Vol. 7, No. 3, 2014, pp. 83-90.

[22] B. Kumar," The Sway of agile Processes over Software Maintainability,", *Int. J. of Computer Application*, Vol. 109, No. 1, 2015, pp. 25-29.

[23] F.J. Pino et al. ," A software maintenance methodology for small organizations: Agile_MANTEMA,", *J. Software Maintenance and evolution*, ©John Wiley & Sons, Ltd., 2011, pp. 851-876.

[24] J.H. Hayes et al.,"Observe-mine-adopt(OMA): an agile way to enhance software maintainability,", *J. Software Maintenance and evolution*, ©John Wiley & Sons, Ltd.,2003, pp. 297-323.

[25] M. Zanker and S. Gordea," Measuring, monitoring and controlling software maintenance efforts,", *Proc. 13th Int. Symp. on Temporal Representation and Reasoning*, ©2006 IEEE.

[26] M. Najafi and L. Toyoshiba," Two case Studies of User Experience Design and Agile Development,", *IEEE Agile Conf.*, ©2008 IEEE, pp. 531-536.

[27] J.Heidenberg," Towards increased productivity and quality in software development using Agile,Lean and Collaborative Approaches,",Ph. D Dissertation, Dept, of Information Technology, Abo Akademi Univ., Turku, Finland, 2011.

[28] S. Jeon et al.," Quality Attribute driven Agile Development,", *9th Int. Conf. on Software engineering Research, Management and Applications*,©2011 IEEE, doi:10.1109/SERA.2011.24.

[29] W.Reyes, "Agile Approaches to Software Maintenance: An exploratory Study of Practitioner Views,", *Managing worldwide Operations& communications with Information Technology*, ©2007, Idea Group Inc.

[30] H.Hulkko and P.Abrahamsson, "A Multiple Case Study on the Impact of Pair Programming on Product Quality,",*ICSE*,©2005 ACM, pp. 495-504.

[31] D.S.Janzen and H.Saiedian, "Does Test Driven Development Really Improve Software Design Quality?," *IEEE Software*, ©2008 IEEE, Vol. 25, No. 2.

[32] K.R.Schougaard et al., " SA@Work A field Study of Software Architecture and Software Quality at work,",*Proc. 15th Asia-Pacific Software engineering Conf.*, ©IEEE Computer society, pp. 411-418.

[33] G.K.Hanssen, "Maintenance and Agile Development: Challenges, Opportunities and Future Directions,", *Proc. ICSM, Canada*, ©!EEE, pp. 487-490.

[34] M.K.Mattsson and J.Nyfjord, "A Model of Agile Evolution and Maintenance Process,",*Proc. 42nd Hawaii Int. Conf. on System Sciences*,©2009 IEEE.

[35] F.K.Y. Chan and J.Y.L. Thong, "Acceptance of Agile Methodologies: A critical Review and conceptual framework,",*J. Decision Support Systems*, Vol. 46, No. 4, 2009, pp. 803-814.

[36] A. Ahmed et al., "Agile Software Development, impact on Productivity and quality,", *Int. Conf. on Management of Innovation and Technology*,2010©IEEE, doi: 10.1109/ICMIT.2010.5492703.

[37] D.Knippers, "Agile Software Sevelopment and Maintainability,", *15th Twente Student Conf.*,2011,the Netherlands,©University of Twente.

[38] K.N.Rao, " A study of the agile software development methods, applicability and implications in industry,"*Int. J. of Software Engineering and its Applications*,Vol. 5, No. 02,2011, pp. 35-46.

[39] M.R.J.Qureshi, "Agile software development methodology for medium and large projects,", *IET Software*, Vol. 6, No. 4,©2012 The Institution of engineering and technology, pp.358-363.

[40] H.Svensson and M. Host," Introducing an agile process in a software maintenance and evolution organization",*Proc. 9th European Conf. on Software Maintenance and Reengineering*,©2005 IEEE.

[41] M.Singh," U-Scrum: An agile methodology for promoting usability,",*Agile Conf.*, ©2008 IEEE, doi: 10.1109/Agile.2008.33.

[42] K.Opelt and T. Beeson, "Agile teams require agile QA: how to make it work, An experience Report,",*Agile Conf.*,©2008 IEEE,doi: 10.1109/Agile.2008.59.

[43] M.Huo et al., "Software Quality and Agile methods,", *Proc. 28th Annual Int. Computer Software and Applications Conf.*, ©2004 IEEE.

[44] C.R.Jakobsen and K.A.Johnson, "Mature agile with a twist of CMMI in Agile,", *Agile Conf.*,©2008 IEEE,doi: 10.1109/Agile.2008.10.

[45] J.Choudhari and U. Suman, "Iterative Maintenance Life Cycle using Extreme programming,", *Int. Conf. on Advances in Recent Technologies in Communication and Computing*,©2010 IEEE, doi:10.1109/ARTCom.2010.52.

[46] X.Meng et al., "A process pattern Language for Agile Methods,", *14 th Asia-Pacific Software Engineering Conference.*

[47] R.Moser et al.,"Does XP deliver quality and maintainable code?,", *8th Int. Conf., XP*, Como, Italy, June 18-22, 2007.
[48] L.Williams, "A Survey of Agile Development Methodologies", ©Laurie 2007.
[49] T.Mens and T. Tourwe," A survey of software refactoring in Software Engineering,",*IEEE transaction on Software Engineering*,Vol. 30,No. 2, ©IEEE Computer Society,2004,doi: 10.1109/TSE.2004.1265817.
[50] A.Sampaio et al.," Towards reconciling Quality and agility in web application development,", *ICWE Workshops'04*.
[51] J.Prochazka,"Agile support and Maintenance of IT services,",Information Systems Development ©Springer Sciences,doi:10.1007/978-1-4419-9790-6_48.
[52] G.Concas et al., "An empirical study of software metrics for assessing the phase of an agile project,",*Int. J. of Software Enineering and Knowledge engineering*,Vol.22,no. 4,2012,doi: 10.1142/S0218194012500131.
[53] J.Verdugo et al., "Using Agile methods to implement a laboratory for software product quality evaluation,",©Springer International Publication Swizerland 2014, pp. 143-156.
[54] S.Nerur et al., "Challenges of migrating to agile methodologies,",Magazine Communications of the ACM-Adaptive Complex enterprises, Vol. 48,No. 5,pp. 72-78.
[55] R.Moser et al., "Does refactoring improve reusability?,", Proc.9th Int. Conf. on Reuse of Off-the-Shelf Components,©2006 Springer, pp.287-297.
[56] 70+Comprehensive Agile Project Management Tool list,[Online], Available: http://www.softwaretestingclass.com/70-comprehensive-agile-project-management-tools-list/.
[57] Top Agile and Scrum Tools, [online], Available: http://agilescout.com/best-agile-scrum-tools.