Agile Software Development

Jinjin Li
Technische Universitt Berlin
Berlin, Germany
E-mail:kimljj@mailbox.tu-berlin.de

***Abstract*—With the further development of computer technology, the software development process has some new goals and requirements. In order to adapt to these changes, people has optimized and improved the previous method. At the same time, some of the traditional software development methods have been unable to adapt to the requirements of people. Therefore, in recent years there have been some new lightweight software process development methods, That is agile software development, which is widely used and promoted. In this paper the author will firstly introduces the background and development about agile software development, as well as comparison to the traditional software development. Then the second chapter gives the definition of agile software development and characteristics, principles and values. In the third chapter the author will highlight several different agile software development methods, and characteristics of each method. In the fourth chapter the author will cite a specific example, how agile software development is applied in specific areas.Finally the author will conclude his opinion. This article aims to give readers a overview of agile software development and how people use it in practice.**

I. Introduction

Before we know the definition of agile software development, what's exactly agile software development, let's first look at its origins and development, as well as its achievements in the field of computer technology.

A. Background and progress

Software Engineering gives the procedures and practices to be followed in the software development and acts as a backbone for computer science engineering techniques[1].

Software development process is a structure imposed on the development of a software product[2],which based on the theory of software engineering.People use it to implement a variety of different software.

Software development methods are attempting to offer an answer to the eager business community asking for lighter weight along with faster and nimbler software development processes[3].

The word agile software development comes from the project management, so what is project management? What is the definition of project management. As William R. Duncanillam[4] in the book "A Guide To The Project Management Body Of Knowledge" said, "Project management is the application of knowledge, skills, tools, and techniques to project activities in order to meet or exceed customer's requirements and expectations from a project". Every project aim to product and delivery products or services to meet customer requirements in the process, in the process people will invest resources and then convert it to outputs of project.

Before the 1960s, computers had just put into practical use, the software design was often only for a particular application in the specified design and preparation. The scale of software was relative small and usually didn't have documentation, rarely use a systematic metohd to development. Design and programming was often equated.

In mid-1960s, large capacity, high speed computers have enabled the rapid expansion of computer applications, the quantity of software development has increased dramatically. The Appearance of High-level programming language and operating system, causing of changes in the way of computer applications. Large amounts of data processing led to the birth of first generation database management system. The software systems became more and more complex and large, software reliability problems were also more prominent. The original personal design, personal method can no longer meet the requirements, software need to change the mode of production. "Software crisis" broke out[5].

Fig. 1. Process of software development

B. Traditional development methods

Software crisis has more or less promoted the maturity of software engineering. In the 1990s, the software development began to use repeated process with the documentation, based on the theory of software engineering theoretical system. At that time, some traditional models came out. Waterfall model was as the representatives of the traditional software project management and had occupied a very important position.

According to the waterfall model (as shown in Figure 2 is a model which was developed for software development; that is to create software. It is called as such because the model develops systematically from one phase to other in a downward fashion, like a waterfall"[]. Waterfall model emphasizes the software development cycle shown in figure

2. And the cycle stage of each step and should be planned and the investment of time, manpower and the use of related technologies in each step should be thoughtful deployed. In the end of each step the results should be reviewed. When customer is satisfied with results, then the next step can be continued. Waterfall method is best suited to the the user whose needs is fixed or results is predictable. Advantages and disadvantages of waterfall model shown in figure 3.

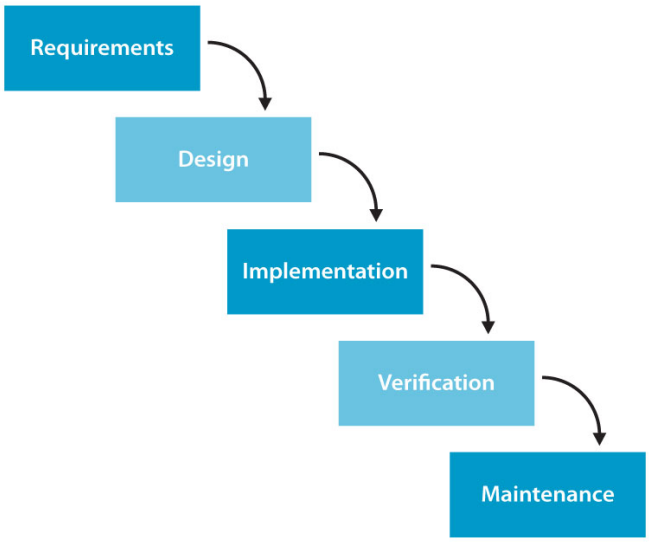

Fig. 2. Waterfall Model

Structured Analysis and Structured Design (SASD)[6] is a software development method that was put forward in the 1970s by Yourdon, Constantine. It emphasizes that divide the whole project or task into a sub-projects or sub-tasks, depicting the various relationships between sub-projects or sub-tasks . This saves time and greatly improved efficiency. SASD became one of the most popular software development method in the 1980s, and then IBM also incorporated in this approach to their software development process. Of course some people have criticized for SASD, because it ignore the participation of users.

Rational Unified Process (RUP) is "a software engineering process framework that captures many of the best practices in modem software development in a form that is suitable for a wide range of projects and organizations. It embeds object-oriented techniques and uses the UML as the principal notation for the several models that are built during the development."[6],was developed by the Rational Software Corporation (acquired by IBM in 2003). It is also an iterative style of development, it introduces a lot of success examples in software development, so it is very suitable for large-scale software development and project. RUP put focus on internal and external communication and exchange of engineering,so it is very valuable for projects which focus on the exchange. Its disadvantage is that customers must have some knowledge of UML, because this method owned by a company. RUP also needs a lot of documentation.

With the further development of software project management, software project management began to emphasize self-adaptive to face various requirements of the market. At that time the traditional software project management has been unable to meet all aspects of the requirements. And then agile software development came out.

The word agile development was mentioned in the Agile Manifesto since 2001. Later agile development achieved great success. More and more people started to pay attention on it and will to use agile development to complete their own projects. Today, agile development has also very many types, for example,Extreme Programming, Adaptive Software Development, Lean Software Development and so on.

Advantages	Disadvantage
The result is predictable	A large number of documents
Strict control	The product will appear in the end of project
High quality	Difficult to modify

Fig. 3. Advantages and disadvantages of waterfall model

II. What is exactly Agile Software Development?

The concept of agile development was proposed in 2001 by the agile team, and then many software development teams and companies recognized and accepted it, and gradually been widely used in many projects. Agile Software Development[7] published the Agile Manifesto shwon in figure II at the same time, on behalf of software development has entered a new era.

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
That is, while there is value in the items on the right, we value the items on the left more.

Fig. 4. Manifesto for Agile Software Development

A. 12 principles behind the Agile Manifesto

1) Our highest priority is to satisfy the customer through early and continuous delivery of valuable software: Agile development is an iterative method using an incremental and delivery of valuable software. Continuous delivery, reflecting the continued iterative process of agile development. Through the early delivery of valuable software in order to listen to customers'advice as soon as possiblethe .That avoid to deviation at the understanding of the users' requirements. The earlier error was found, smaller was the cost at the correction of deviation. Customers can with this principle fully experience software company's efficiency and attention, satisfaction.

2) Welcome changing requirements, even late in development: Even in later time of development, agile development is also willing to make the appropriate changes according to requirments. When software really meet the needs of user and market, is a valuable software. As agile development has reserved the space in system design for changes, so agile development can minimize costs arising from changes in requirments.

3) Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale: Short period of iterations can ensure that the project team to make cooperation more closely with customers. In each new delivery, the project team will deliver improvements to the software or add new features on basis of previous delivery. And these improvements and new features must be tested, can work and achieve the quality standards that can be released.

4) Business people and developers must work together daily throughout the project: Information in the transmission process would inevitably lead to the case of distortion. When business people describe the requirments of customers to the developers, developers may have misunderstanding of the business. Therefore, throughout the project development, business and developers would need more frequently and meaningful interaction to identify problems as early as possible. Project team members work together every day, from the time and space, to ensure that the communication between business and developers are more convenient.

5) Build projects around motivated individuals.Give them the environment and support they need, and trust them to get the job done: Only with such a trust, motivation and support full potential of all team members would be released.

6) The most efficient and effective method of conveying information to and within a development team is face-to-face conversation: In large team, in order to transfer knowledge in the form of the document, it is more appropriate to communicate with each other, he can make people accept both knowledge and information at the same time. But the agile team has normally only 7 to 10 people, the use of document to communicate will waste a lot of time by writing the document. Face to face conversation between team members could transmit information more quickly and efficiently.

7) Working software is the primary measure of progress: In agile development, each iteration is to deliver a working software, so the measure of progress is no longer the number of lines of code was written, the number of test cases was implemented, but the number of software that was tested, achieve release standards and can work.

8) Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely: In software companies, work overtime is a very common thing. However, agile development oppose to in the form of work overtime to complete iterative tasks. Overtime will lead to team members to become fatigue, boredom, it is difficult to ensure efficiency.

9) Continuous attention to technical excellence and good design enhances agility: In agile development need to respond positively to change, communicate with each other is also important, people pay more attention to good design and technology, better agile ability will become.

10) Simplicity–the art of maximizing the amount of work not done–is essential: Anyone could be completely expected changes in requirments accuratly. Agile teams advocate that everyone should pay attention to what is the easiest way to complete the current problems, rather than to build a future software features that may be required. Agile development does not advocated using complex technology to implement software.

11) The best architectures, requirements, and designs emerge from self-organizing teams: Self-organizing team is able to positive communicate with each other to form a common work ethic and culture.They do not need detailed instructions, this let team members have more confidence.

12) At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly: When the projects is during the progress, not only the requirements may change, but also there are many uncertain factors, such as changes in team members, According to a predefined plan to work is difficult to achieve agility. It requires in a certain time intervals, the team need to reflect on their work, and make appropriate adjustments.

Of course,just like Scott Ambler in his book wrote[8],"These principles form a foundation of common sense upon which you can base successful software development efforts." That means these priciples are the foudation of succeed, they are not enough to let your project or software to work.

B. Differences between the agile development and traditional development

So What is exactly agile software development and the What is the differences between it and traditional software development model ?Before giving the definition, we firstly compare differences between the agile development and traditional development based on Agile Manifesto.It is not difficult to summarize that there are two points[**?**] which agile devolopment obviously emphasis on.The one is adaptability, the another one is teamwork.

Model	Adaptability	Teamwork
Traditional	Too much reliance on documentation Design and impelment are separete	Lack of communication Manager take charge of process
Agile	Documentation, design and demand can be changed during the implement	Divide into small group Flexibel and efficient

Fig. 5. Differences between the agile development and traditional development

1) Adaptability: The difference to traditional model, agile development take more emphasis on adaptability, rather than the predictability of traditional model. The person who choose the traditional development mode , when starting a project, always think of a very detailed and complete documentation. They will analyze the entire development process and details of each sub-process for example, how many times or how many people will be invested in, finally the results are written in the document. The most important thing is that the document can not be change after it once was identified. All project developers are requird to strictly follow the document.When someone want to change the document or timeplan, that is not allowed. Although this development model also has its advantages when the demand from the beginning to the end did not change. But gradually people also found many problems.

Firstly, the part of the design document takes a lot of time, because a very comprehensive documentation required all the details of the project have been implemented. In this way to create a documentation in itself is no problem, but this development efficiency will be lower. Most of the modern project hope that in whole phase the demand didn't change, it is unrealistic, too many incentives will cause the changes in requirments.

Agile willing to accept changes, even in the latter prozess of software development. Its own methods of system design and system builders can quickly respond to changes in customer demand. It ensure that the results of the last iteration is the customer's really needs, and it meets changes of market. Differs from the waterfall model, agile development fully comply with plan. Agile development would at the beginning of a project to develop a rough plan, providing more space to changes of project.

Secondly, If there is no real users that participate in requirements definition process, the definition may be difficult to meet the needs of end-user's work habits. Even if the requirements is defined, and then show it to the end user, let them to confirm the result, but there is a risk that demand would be changed. The reason is that everything is imaginary until the user can run the system, because users could not understand programming, when there is something unreasonable, he can not see it. Let's assume that the user had already carefully participated in confirmation of requirments and most of the requirments were identified in this stage, but some non-functional requirements have no way to show the user, such as performance, smooth operation and so on. It's just like that someone give an iPhone manual and an andorid phone manual, let you to evaluate which cell phone is better. Besides the beautiful look and user interface of iPhone, you can not really imagine which execellent advantages does iPhone has that make so many young people to seek after it. Until you actually hold it in your hand when using and experience the thrill of the iPhone's smooth operation. Then you can feel on iPhone what is exactly distinguishes.

Agile development is not completely closed, the develop team seek positive communication with customers, even to understand the changing needs of customers. Cooperation with the customer does not stay in the contract negotiations, but more concerned about the investigation in customer-related business of software products and technical aspects.

Thirdly, traditional model would completely separate code and design, it is unrealistic for many projects. However, if the demand in the project later with a greater risk have to be changed, so do it early in more detail may be inappropriate. So, assuming that demand can be determined, but can design also be determined? My understanding is that in case when we are very familiar with the technology in the area is possible. But how should the technology of one field is very familiar with? Like some small companies which just implement some office systems (OS), they have implemented a OS for company A, When they also implemented a similar system for company B, The used technology is almost the same, but there are some similarities and only small differences in the business. However, when a company want to develop a projects in a unfamiliar field. Before you investe many people into the features of pre-design, you would find it hard to refine the design. Pre-desig for the project usually takes a long time. Completing design refinement at the beginning of a project may lead to backfire.

Agile will be divided into multiple sub-projects, the completion of each project is a small phased achievements. By using visualize, actionable phased achievements instead of comprehensive documentation. Let customers to have a more intuitive feel about the prozess of development and put forward a more accurate advice on the software in the next step. Agile development focus on that in a short time to deliver valuable software to customer to make them satisfied and during development ongoing delivery of runable software. The time for delivery should be as short as possible.

2) Teamwork: The goal of traditional process management is to ensure that the process within the organization as expected execute and the defined process is strictly adhered to. Document-centric process tend to define people's roles as interchangeable, reliable machine parts.

Agile software process is people-centered rather than process-centric.They believe that individuals and their interactions important than processes and tools. Practice is the life of methodology[9]. A key point of Agile development is let people to accept a process rather than impose a pro-

cess. Developers must have the right to make all decisions of the technical aspects. Process is the second point, so it should be minimized. The center of Agile development is to establishment project team with positive staff. Give them the necessary environment and support, having full of confidence to their work. In the project group, the most useful and most effective way of communication is face to face conversation. It embodies the principles of human-centered.

This form of oral communication compared to document-centric that the interaction is more faster and efficient to transmit the information and sovle the problem. It requires also during the entire project development, developers and business people should always be together. Focusing on communication between team members, this is the embodiment of agile development project. The end of research for business people is not to write a requirement analysis and then send it to developers. But should more effectively communicate with developers to ensure that developers understand the business correctly.

After we have a full understanding of agile development und also understood the difference between it and traditional development, now we give the definition of agile development. Agile software development is a capability that can rapidly response to changing needs of the software, which focus on rapid delivery of high quality software, and achieve customer satisfaction[9].

III. Which Agile Software Developments are being widely used

We know from the previous chapter's introduction, agile development is a way that with a human-centered, focusing on iterative, step by step to development approach. In agile development, software project is split into several subprojects, the subprojects will be developed in a certain period. In other words, agile development divide a large project into the small projects that multiple interrelated, but can also be run independently, and then separately completed.

The initial concept of agile development is agile process, including Extreme Programming, Scrum, Lean Development, and so on. Extreme Programming is a plan for the project management practices, Scrum is an agile project management framework.

A. Extreme Programming

Extreme Programming is a lightweight software development methodology.Extreme Programming is from practice, it is also a summary of the practice, its main feature is to adapt to changes in the environment and requirment and give full play to the subjective initiative of the developers. Extreme Programming commitment to reduce software project risk, improve responsiveness to business changes, increase productivity during development, the software development process to increase the fun, I believe that is enough to attract everyone.

In Extreme Programming, at first the four variables are introduced, cost, time, quality and scope, by studying the interaction between variables, we can get a beeter thorough analysis of project development.In order to successfully implement Extreme Programming, Extreme Programming has defined four values and twelve principles. Extreme Programming is a very large knowledge base, each of which is a scholarship worth studying.

Extreme Programming was proposed by Kent Beck in 1996. It is the development methodology that suitable for small and medium team and rapid changes in demand. It developed software that should meet the changing needs of customer, it is the goal of Extreme Programming. Extreme Programming is similar to spiral development, which divide the development process into a relatively simple cycle. Through positive communication, feedback and a range of other methods, developers and customers can have a very clear understanding of software development progress, change, problem to be solved and the potential difficulties, and adjust development process according to the actual situation in time.

Compared with the traditional way of project development, Extreme Programming emphasis on that do the best by its list of methods and ideas, other Extreme Programming are not advocated, flatly ignored. A strict implementation of Extreme Programming project, its development process should be efficient and fast, able to do one week in 40-hour instead of delay in project progress.

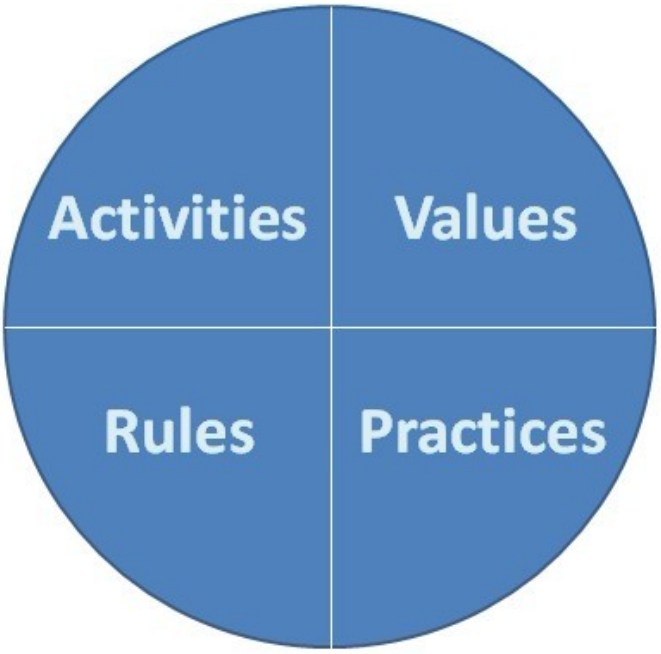

Fig. 6. Components of Extreme Programming

1) Extreme Programming Process: Extreme Programming consists of four components: values, principles, practices and behavior. Principles from the value; the values and principles are based on 12 practices; 12 practices associated with the four main software development activities. Figure 6 shows the dependencies between them, this four-part behavior is throughout the entire life cycle[10].

2) Extreme Programming Values: Extreme Programming provides a global, values-driven development process view. To achieve high iterative process, the team need to have four values: communication, simplicity, feedback and courage[9].

Communication. A tacit understanding is the biggest challenge of the development team need to face, the most effective way to solve the problemto is to strengthen communication.

Simplicity. There is often a feedback in iterative, so that developers do not need to plan the expansion of system design, only need to consider the simplest possible solution and try to simplify the steps to complete the work. At a later stage, if necessary, and then make a change, this avoids the resources spent on the program that is complex and not meet real needs of customer.

Feedback. Developers and customers need to communicate constantly and through continuous, clear feedback, when they meet problems in EXtreme Programming project. At the same time, the software developers need to quickly implement some functionality, presentation to customers, get customers' high quality, accuracy feedback on the current progress.

Courage. Once an error or changes in requirment occur, developers may need to drop some previous work and then reimplement. In this case, the developers undoubtedly need for courage. Ignored for the past mistakes is to against the the principles of high-iterative process. Courage can be likened to playing the maze game, when the wrong way was founded, the developers must immediately turn back, this does not mean just a waste of the time, but rather the price to find a maze's exit.

Extreme Programming is actually driven by these four values. In other words, if you want to achieve an Extreme Programming practice, and did not get the four values, then it will lose the meaning of Extreme Programming.

3) Extreme Programming Principels: Built on the basis of the values of Extreme Programming, there are five key principles to guide software development: quality work, incremental change, rapidfeedback, assume simplicity and embracing change[9].

Auality work refers to the quality of work and product can not be ignored, developers can not give up the software quality because of the development progress, and make concessions.

Incremental change. Developers should through a series of small changes to solve a problem. In the planning, development and design process, this principle should be adopted .

Rapid feedback. Developers through the short iterative cycles to get feedback and quickly check the current product if it meets customer requirements.

Assume simplicity. Try to solve each issue with the simplest way. This means that developers only need to considered the problem within the current iteration. Design should be as simple as possible, just to meet the needs of the current iteration.

Embracing change. In solving pressing problems, adopt an tolerant strategy. Developers need to understand customer needs are constantly changing, so the features and functions of the priority needs are constantly changing. To embrace change, developers need to solve pressing problems, and often make appropriate adjustments to changes in reruirements.

4) Extreme Programming Activities: Traditional software development consists of four main activities: requirements, design, coding and testing. The four activities of Extreme Programming have the same name with these, but their essentially is different. The following is brief description of the four activities of Extreme Programming[9].

Listening. Extreme Programming is based on communication. It depends not so much on formal written document. Developers not only need to listen to customer requirements, but also the development team needs to listen to opinions from other people of team. Developers need grasp good communication skills.

Designing. Extreme Programming has a rather novel point of view, the design is followed for the development of the project. Design is no longer a static or a project phase, but instead of a dynamic process. It is no longer the task the developer.

Coding. In Extreme Programming, many of the practices such as pair programming, code refactoring, test driven development are focus on programming, to get high-quality program. Extreme Programming developer s considered code as interesting text, rather than the encrypted text which difficult to understand. So the meaning of the code should be obvious, that means with high readability to make the other team members can understand the logic, algorithms and processes.

Testing. In Extreme Programming, Test is one not easy to perform[11].Good test is the key to ensure the quality of software product. Extreme Programming test is run through the entire development process. From the beginning the code could be tested,the most of the issues of software should be resolved at an early stage, rather than to wait until a later stage. In this way developed software's quality is guaranteed, the development costs will be reduced.

5) Extreme Programming Practices: 12 Extreme Programming practices is summed up by the developer of Extreme Programming, it reflects well the principles of Extreme Programming[12].

Planning game. The purpose of the Planning Game is to guide the product into delivery[13]. In the project planning stage, you need to get the user's needs. The way how Extreme Programming get the requirements from customers is different form traditional way. In Extreme Programming customers get story card from developer and then write down all their requirements on the cards. Developers predict the time that required to achieve on each card, and then According to the priority from users and the estimated time developers would select the the tasks which in first iteration of should be completed. After the first iteration and the result will compare with the original story card. Before the start of the next iteration, the user can modify the user card, and then start the next iteration.

Simple design. Extreme Programming consider code should be designed as simple as possible, just meet the requirements of the current function. Traditional software development is a

top-down design, emphasizing the design first, before you start writing the code, there must be a perfect model. In contrast, in Extreme Programming developers according to the present requirements to work hard to find the simplest design.

Small releases. Extreme Programming emphasizes that every 3 weeks should give an iterative version of the distribution system to the user. It also gives customers more opportunities to communicate with developers. Customers will provide feedback based on this small release, telling developers where are problems, which would allow developers in the next release to make adjustments in time.

Metaphor. System metaphor is through the system's description let all developers have a very clear outline of the project to help everyone understand the system's basic elements and their relationships. Because developers are not familiar with the term of the business, customers also do not understand.term of software development. Through these metaphors to enhance understanding between customers and developers to avoid misunderstanding.

Pair Programming. It means that there are two developers in each small group, the one of role is driver and the another is navigator, they can chang their role with each other during programming. Two programmers to do "a programmer's work" seem inefficient, but in fact is just on the contrary. In addition to providing better code and tests, it also provided for the transfer of knowledge in the team. Everyone can also learn new knowledge from each other. Work will become fun.

Coding standards. In order to be able to understand each other developers must follow a common coding standard when whey write code. In Extreme Programming the document should be minimized, better understanding code are considered as the best documentation.

Coneetiveownership. No one can have any part of the code alone. All the code are released into the version of repository. Anyone has the right to modify code and does not require any other person's permission.

Continuous integration. It means that put continuously the complete modules together. Purpose is to get the ongoing customer's feedback, and to early detect errors. Traditional method works as follows: perform big-bang integration after all code were written, and then spend a long time to correct the problem. Obviously continuous integration is conducive to detect problems early. Integration should be done many times a day.

Refactoring. It is a process that improve the code has been done. It optimize the code's internal structure, when in case that the external behavior does not change. Its purpose is to improved software design and code quality and maintainability.

Testing. Extreme Programming emphasizes test should be written before the start of coding. The tests are mainly performed is unit testing and functional testing.

On-site customer. Extreme programming requires at least one representative of the customers in the entire development cycle responsible for ensuring requirements, answering questions and writing functional testing. Experience has shown that if a customer is on site, can improve the accuracy and efficiency of the development.

40 hours a week. Extreme programming requires developers to work every day for no more than eight hours working time per week does not exceed 40 hours, no overtime continuously for more than two weeks, so as not to affect efficiency. Smooth development process ensure that programmers can continue to complete the task, team can continue to deliver release to its customers. Continue to work for a long hours will kill efficiency, fatigue developers will make more mistakes.

B. Scrum

Scrum is proposed by Ken Schwaber and Jeff Sutherland[14], seeking full play to object-oriented development approach, is also a improvement of iterative object-oriented method. Scrum is from Rugby (in the game each player should keep the overall judgments of the court, and through collective action, struggling to achieve the same goal–victory). Scrum was practiced for the first time in Easel (1993), is suitable for software development, which requirements are difficult to predict[15]. Scrum's meeting, sprint, backlog, Scrum Master, Scrum Team have been used by PLOP as the standard of Organizational and Process Pattern[16].

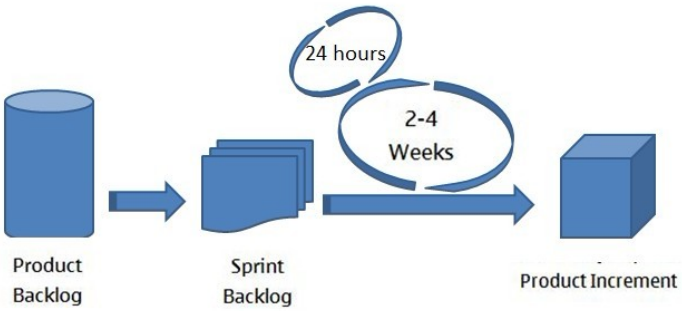

Fig. 7. Scrum process

Scrum's basic idea is: To develop software like new products development, you can not at the beginning define the final product specifications. Process requires research, creativity, attempt and error, so there is no process can guarantee a fixed success of the project. Scrum has a clear ultimate goal, familiar with the best sampel model and technology, which are necessary for the development process. With a high autonomy and flexibility, closer communication and cooperation to solve the challenges. Ensure that every day, every stage has a clear increment. Therefore, Scrum is ideal for product development projects.

Scrum development process is usually an iterative cycle of 2-4 weeks[14],shown in figur 7. Each iteration cycle is called a Sprint, a new products start with customer requirement. Development team must make every effort to deliver results in each cycle. Team hava a meeting for 15 minutes a day to view

each member's progress and plans, in order to understand the difficulties and try to figur out, then make decesion to arrange the next day's tasks.

1) Role: Only three roles in Scrum: Scrum master, Scrum team and Product Owner

Scrum master is a little different from project manager. His duty is to help Scrum team to handle the things besides developing, such as arranging and hosting meetings with customers, management. Scrum master is to guide the team, rather than control them. He is also a interface between development team and external.

Scrum team is self-organization, that means everyone in the team can disscus that which task is by whom, each member has the same responsibility and authority. The size of each Scrum team is between 5 and 9 person. What is the specific structure within the team has not been defined, but the actual project team determine the size and complexity of the structure.

Product owner is responsible for all requirements, ROI (Return Of Investment), project Objectives and the entire project. He is also responsible for updating product backlog and the order of requirements' priority.

2) Sprint: Sprint is usually a 30-day cycle. During a sprint people do development and test. After the end of a Sprint, the another one will begein till all of the tasks were finished, this process is a iteration. At the end of each sprint, should be able to demonstrate a working product. If some work is not done at the end of sprint, then they should be written back to the Product Backlog in order to discuss them in the next Sprint Planning Meeting. If the sprint task is completed before the end, with the help of product owner some tasks would be selected from the product backlog into Sprint Backlog.

Sprint Planning Meeting will be divided into two parts. In the first part Scrum master, Scrum team, management, Product owner discuss what and how will task developed in next Sprint. Only Scrum master and Scrum team participate in the second part. Participants refine the selected task into several small steps, then write Sprint Backlog.

There is a Review meeting in the end of each Sprint for Product Owner. In the meeting Scrum team show in the Sprint what have they done. The merits of the solution should be described and analyzed at the meeting. Results of presentation would be compared with goals identified in the Sprint Planning Meeting to determine whether the development team have complete their tasks. At the meeting, the customer can have the opportunity to change the direction of future product development.

After Review Meeting there is a retrospective meeting. Purpose is to summarize the Sprint. Each participant should answer the following two questions: What is good in previous Sprint and what could be improved in next Sprint.

3) Scrum advantages and limitations: In the process of the project, all project team members have a very strong sense that Scrum brought many positive changes such as: Adapt quickly to changes in requirements and released on time, improve efficiency and reduce risk and product quality.

What are the limitations of that Scrum? There are two points. It Support not so good for distributed development environment and lack of support for large and complex software.

IV. A real example of Agile Software Development

The railway of Netherlands transported 1.2 million passengers every day. They use a new information system, provide more accurate information on the train, reducing human intervention. AgileDo have developed the PUB distribution system, it controls the information display and audio broadcasting of all stations[17].

A. Start

Someone tried to develop this PUB system with Waterfall model. Just like the way of Waterfall, they have written a detailed documentation and given it to developers. After 3 years, this project was cancelled, developers haven't finished it because of some reasons, maybe changs in requirement or something unexpected. Then the railway company employed AgileDo to develop it.

The project started with a 3 weeks preparation phase that prepare everything they need in sprint. One project manager, one architect and one Scrum master were responsible for this phase. They have choose two business analysts as project owner who with experiences on PUB, because they found it not enough for one people to be owner in this project.

Project manager will decide the priorty of tasks in each sprint. But he was often absent. Therefore, the manager had modified the priority for several times on the day he was absent. Ideally, every person who has the final decision on the priority should participate in sprint planning meeting.

B. Group work

This project have 7 member. They had formulated some rules in Wiki which show how the team member works together, for example, about development tools, work time, quality etc. With Wiki the members can have a better consensus. When someone want to modify or update something, they must write it in Wiki. It's good when new menber join in.

In the first few iterations they had build, tested and verfied the user stories. That makes customer very satisfied. After several iterations they had expanded the project. The indian member came back to India, then the project had two Scrum team, every team had 5 people, the 2 team shared a tester. Then the project was divided into 3 teams, each had a tester. Each team had dutch and indian.

How each team contace with each other? Firstly, They used Skype to have one to one or whole meeting. To ensure the progress of meeting can be successful holded, they had used UPS. Pair programming was builded with the member who comes from the same country. They felt that no matter with which tools the partner in pair programming should sit together.

They used Scrumworks to make record about sprint process and who had finish something. When they disscus something with product owner, Scrumworks is very helpful and useful.

Thet had met some problems. For example, product owner can't speak english. According to Scrum, plan meeting was divided into 2 parts. In the first part, product owner tell the user story only to dutch member, then they discuss the user story together and estimate it with skype. After sprint was showed in Netherlands, the member in Netherlands will write a report to the team in India.

C. Team for architecture

They also need to know the non-functional requirements from user. Although the product owner was familiar with the core functional requirements, but he didn't know well about the security, log, usability and so on. So they need build a team to contact with the other department in company. Their job is only pay attention to architecture and non-functional requirements, then change it into user story in backlog.

D. Requirements management

The product owner was responsible for requirement documentatin and backlog, because the user requires a deteiled documentation. Usually the team just need the user story in backlog and product owner can explain it, that is enough.

This documentation is also useful for external tester.

E. Test

They had made a test after each sprint to ensure they can hand out the runnable software. The external tester hadn't found many bugs.

The test had two parts, unit testing and acceptance testing. They did unit testing with JUnit and Clover. The target was that 80

They had a trouble to test a complex client. It is more harder than server site. So they did test manual. But the test time become longer and longer, the even worse thing is, the external tester founded bugs only in this part. This problem would be solved with auto testing. So they advise that it deserve to do auto testing, espacially in later time of project.

F. Result

Customer was very satisfied with the result. The improtant thing was that they discuss with customer how to improve the software.

Customer found a audit outsourcing company to audit the software. The conculusion was maintainability of the system is very good and the source code was with a very high quality.

G. Experience

It is difficult to find a product owner with wealth of knowledge about requirement and also know how to set the priority. The project team need at least one product owner, espacially in big project. To ensure the project will be finished on schedule, it need a good backlog and estimation. Although the software development process does not require a lot of documentations, but the customer may need. Scrum is also suitable for distributed development. Before the project start a meeting should be holded to let all the member have a consensus.

V. Agile Software Development in university

Programmers, or people who do something refers to project management. Is that a truth that peoople could really understand or use agile software development only only have participated in specialized training or gone through some real agile development? The answer is no. The university student have already got the chance to experience what will Agile Software Development bring to them.

Project has already become to one of favorite course in many universities. Becasse in project student could learn some thing practical and useful. Just like the student in majority in Computer Science and Technology, they can apply their knowledge in project and learn experience, rather than write "hello world" in their own mind. As university apply the project more mature, people are think about how could they make it better.

University School professors and teachers began to think, in addition to technical factors, how could they improve project efficiency and increase student's communication between each other, so that they can learn from each other, have a good understanding of each other. Ultimately it will improve the quality of project products, reduce errors, save time.

VI. Conclusion

Agile Software Development have brought us many many good things in software development. The most intuitive is the improved quality of products, improved efficiency of developers and less errors. But we can not ignore its limitations. Especially in distributed development and large projects Agile Software Development can still not good show the its advantages. In my opinion, Agile is an attitude which is positive, efficient, and cooperative.

Acknowledgment

I would like here to thank my superviser Ulrich Bareth, he helped me a lot by writting this paper.

References

[1] D. N. M. S. V. TAPASKAR, "Enacted software development process based on agile and agent methodologies," *International Journal of Engineering Science and Technology*, vol. 3, no. 11, 2007.

[2] D. D. Jamwal, "Analysis of software development models," *IJCST*, vol. 1, no. 2, 2010.

[3] J. R. J. W. Pekka Abrahamsson, Outi Salo, "Agile software development methods - review and analysis," *VTT Elektroniikka*, 2002.

[4] W. R. Duncanillam, *A Guide To The Project Management Body Of Knowledge*, 1996.

[5] W. Royce, *Software Project Management:A Unified Framework*, 1998.

[6] P. Kruchten, "Introduction to the rational unified process," *Proceedings of the 24th International Conference on Software Engineering*, p. 703, 2002.

[7] K. B. Marten Folwer, James A.Highsmith, Manifesto for Aigle Software Development, http://agilemanifesto.org/, 11 2011.

[8] S. Ambler, *Agile Modeling:Effective Practices for eXtreme Programming and the Unified Process*. Wiley Computer Publishing, 2004.

[9] J. Highsmith, *Agile Software Development Ecosystems*. Addison Wesley, 2002.

[10] M. L. Back R. J., Hirkman P., "Evaluating the xp customer model and design by contract," in *Euromicro Conference*, 2004, pp. 318–325.

[11] D. Karlstrm, "Introducing extreme programming an experience report." *Proceedings 3rd Conference on extreme programming XP 2002*, 2002.

[12] K. Beck, “Embracing change with extreme programming,” *IEEE*, pp. 70–77, 1999.
[13] M. G. M. Frank, “Introducing agile methods: Three years of experience,” *IEEE*, pp. 334–341, 2004.
[14] K. Schwaber, *Agile Project Management with Scrum*. Microsoft Press, 2004.
[15] J. Sutherland, “Agile can scale: Inventing and reinventing scrum in five companies,” *Cutter IT Journal*, 2001.
[16] Y. S. K. S. J. S. Mike Beedle, Martine Devos, “Scrum: An extension pattern language for hyperproductive software development,” in *PLoP: The 1998 Pattern Languages of Programs Conference*, 1998.
[17] A. Yuan, *A real example about Agile Software Development*, AgileDo, 2009.