A Survey on Efficient Agile Development Methods

M. Vijaya Bharathi[1]
[1]*Asst.professor, CSE Department, GMR Institute of Technology, A.P, India,*

V.Spurthi[2]
[2] *M.Tech, CSE Department, GMR Institute of Technology, A.P, India,*

Abstract

The crux of Agile Methodologies is that the changes in the requirement can be managed by software even during the development cycle of the software development.Eight Agile Methodologies had been studied and surveyed in current scenario of software development. These are Extreme Programming, Scrum, Crystal Methodologies, Rational Unified Process, Adaptive Software Development, Feature Driven Development, Dynamic Systems Development Method and Lean Development.In this paper we are going to systematically review the existing literature on agile software development methodologies and review of six agile approaches including Extreme Programming, DSDM , Crystal , Lean Development ,Feature-Driven development and SCRUM.

Index Terms*: Agile Methodology, Scrum, Agile Teams, Extreme programming, DSDM (Dynamic Systems Development Method), Feature-Driven Development, Lean Development, Software Development.*

1. Introduction:

Agile methods are a subset of iterative and evolutionary methods. Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. "Plan-driven methods work best when developers can determine the requirements in advance and when the requirements remain relatively stable, with change rates on the order of one percent per month**".**Plan-driven methods are those that begin with the solicitation and documentation of a set of requirements that is as complete as possible. Based on these requirements, one can then formulate a plan of development. Usually, the more complete the requirements, the better the plan. The introduction of the extreme programming method (better known as XP) has been widely acknowledged as the starting point for the various agile software development approaches.

There are several other methods like Crystal Methods, Feature Driven Development and Adaptive Software Development belongs to the same family of Agile Methodology. Custom software development is a Privileged methodology in which the system development is a linear, sequential, managed and controlled process. This approach needs to be feasible, adaptive and flexible enough for the developers to make late changes in the specifications. Agile Software Development is random, opportunistic, simultaneous and overlapping process. The following factors resulted in the "Agile Movement" in the software industry.

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation
- Responding to change over flowing a plan

When a software development is incremental, Co-operative, Straight forward and Adaptive (able to make last moment changes) then agile methodology provide a best solution. As such the agile methods are principle-based, rather than rule-based. They have predefined rules regarding the roles, relationships, and activities, the team and manager are guided by these principles:

1. The highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter time scale.
4. Business people and developers must work together daily through the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of

progress.
8. Agile processes promote sustainable development.
9. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
10. Continuous attention to technical excellence and good design enhances agility.
11. Simplicity – the art of maximizing the amount of work not done – is essential.
12. The best architectures, requirements, and designs emerge from self-organizing teams.
13. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

2. Agile Methodologies:

Here we provide a brief introduction to six agile methodologies. The six were chosen to demonstrate the range of applicability and specification of the agile methodologies. For each methodology we provide an overview process.
Agile methodologies include:

- Extreme Programming
- SCRUM
- Crystal
- DSDM
- Lean Development
- Feature-Driven Development

2.1. Extreme Programming(XP):

A particularly popular agile methodology is extreme Programming (XP). XP was developed by Beck, Cunningham, and Jeffries and is a "lightweight discipline of software development based on principles of simplicity, communication, feedback, and courage". These four key values are described in further detail below:
Simplicity: Rather than attempting to predict future requirements of the software system at the outset, "extreme programmers do the simplest thing that could possibly work," and "leave the system in the simplest condition possible" . This improves the overall speed of development while still retaining an emphasis on working software.
Communication: Poor communication in software teams is one of the central causes of failures within projects. In XP, good communication is stressed between all project members—customers, team members, and project managers. A representative from the customer should be present on site at all times to answer questions and clarify project requirements. Pair programming is used, so each programmer can constantly review the other's work.
Feedback: There should always be some way of getting information about the system, to accurately determine the state of the development process. Such feedback serves as an indicator of the project's progress and informs project leaders when changes need to be made.
Courage: XP programmers are encouraged to experiment and rewrite code if they are dissatisfied with the existing code or design. This helps maintain morale about the project and supports further communication with other project members.
Advantages:
1. Lightweight methods suit small-medium size projects.
2. Produces good team cohesion.
3. Emphasis's final product.
4. Iterative.
5. Test based approach to requirements and quality assurance.
Disadvantages:
1. Difficult to scale up to large projects where documentation is essential.
2. Needs experience and skill if not to degenerate into code-and-fix.
3. Programming pairs is costly.

2.2. SCRUM

The SCRUM development process was introduced to make development of software systems more flexible and lightweight than traditional heavier methods. "Scrum is an iterative, incremental process for developing any product or managing any work".[3] The primary characteristic of SCRUM relative to more traditional development methods is that it assumes an element of chaos in the development process. Compared to the traditional methods such as Waterfall, Spiral, and Iterative, SCRUM is best prepared to handle changes in the environment, and can more easily respond to changes in requirements, schedule, and other externally or internally defined updates. "SCRUM acknowledges that the underlying development processes are incompletely defined and uses control mechanisms to improve flexibility". SCRUM achieves this flexibility and adaptability through a well-defined development process designed to recognize and Respond to changes in the environment. The key features of the SCRUM methodology are the Planning phase, Sprint phase, and Closure phase.
Planning Phase: During the planning phase, a Product Backlog is created, which a list is containing the features desired by the customer. The delivery date for the final product is specified; prioritization of system components are laid out, project cost is estimated, and potential risks to the product development are assessed. All information regarding what the intended product should be is

determined at the outset, before any development begins.

Sprint Phase: The actual project development occurs in the Sprint phase. A Sprint is an "iterative cycle of development work," and generally lasts between 1 and 4 weeks. At the outset of a Sprint, a subset of features from the initial Product Backlog is assigned to be completed during the Sprint. During the course of a Sprint, no other features can be added, but features within the Sprint's Backlog can be updated or changed depending on environment variables

Closure Phase: Once a product has met expectations of the development team, management and customer, the product is prepared for general release as part of the Closure phase. At this point, the product is final and tasks such as preparing training materials, adding user documentation, and preparing marketing material are completed

Advantages:

- Communication can improve across all the teams.
- It provides for an open forum, where everyone knows who is responsible for which item.
- Scrum can increase team efficiency by as much as 20 percent.
- Problems are more transparent.

Disadvantages:

- Decision-making is entirely in the hands of the teams.
- There has to be constant, hands-on management.

2.3. CRYSTAL

Crystal Methods were developed to address the variability of the environment and the specific characteristics of the project. All the Crystal Methods emphasize the importance of people in developing software.[4] The graph in Figure 1 is used to aid the choice of a Crystal Method starting point (for later tailoring). Along the x-axis is the size of the team. As a team gets larger (moves to the right along the x-axis), the harder it is to manage the process via face-to-face communication and, thus, the greater the need for coordinating documentation, practices, and tools. The y-axis addresses the system's potential for causing damage. The lowest damage impact is loss of comfort, then loss of discretionary money, loss of essential money, and finally loss of life. Based on the team size and the criticality, the corresponding Crystal methodology is identified. Each methodology has a set of recommended practices, a core set of roles, work products, techniques, and notations.

Life(L)	L6	L20	L40	L100	L200	L500	L1000
Essential money (E)	E6	E20	E40	E100	E200	E500	E1000
Discretionary Money(D)	D6	D20	D40	D100	D200	D500	D1000
Comfort(C)	C6	C20	C40	C100	C200	C500	C1000
	1-6	-20	-40	-100	-200	-500	-1000

Figure 1: Crystal

Crystal focuses on people, interaction, community, skills, talents, and communication as first order effects on performance. Process remains important, but secondary".[4] There are only two absolute rules of the Crystal family of methodologies. First, incremental cycles must not exceed four months. Second, reflection workshops must be held after every delivery so that the methodology is self-adapting. Currently, only Crystal Clear and Crystal Orange have been defined. Summaries of these two methodologies are provided be

2.3.1. Crystal Clear:

Crystal Clear is an optimization of Crystal that can be applied when the team consists of three to eight people sitting in the same room or adjoining offices. The property of close communication is strengthened to "osmotic" communication meaning that people overhear each other discussing project priorities, status, requirements, and design on a daily basis. Crystal Clear's model elements are as follows:

Documents and artifacts: Release plan, schedule of reviews, informal/low-ceremony use cases, design sketches, running code, common object model, test cases, and user manual.

Roles: project sponsor/customer, senior designer-programmer, designer programmer, and user.

Process: incremental delivery, releases less than two to three months, some automated testing, direct user involvement, two user reviews per release, and methodology-tuning retrospectives. Progress is tracked by software delivered or major decisions reached, not by documents completed.

2.3.2. Crystal Orange:

Crystal Orange is targeted at a D40 project. Crystal Orange is for 20-40 programmers, working together in one building on a project in which defects could cause the loss of discretionary money (i.e., medium risk). The project duration is between one and two years and time-to-market is important. Crystal Clear's model elements are as follows:

Documents and artifacts: requirements document, release plan, schedule, status reports, UI design

document, inter-team specs, running code, common object model, test cases, migration code, and user manual

Roles: project sponsor, business expert, usage expert, technical facilitator, business analyst, project manager, architect, design mentor, lead designer programmer, designer-programmer, UI designer, reuse point, writer, and tester

Process: incremental delivery, releases less than three to four months, some automated testing, direct user involvement, two user reviews per release, and methodology-tuning retrospectives.

2.4. Feature - driven development:

Throughout, FDD emphasizes the importance of having good people and strong domain experts. FDD is build around eight best practices: Domain object modeling; developing by feature; individual class ownership; feature teams; inspections; regular builds; configuration management; reporting/visibility of results.UML models are used extensively in FDD.

Process: The FDD process has five incremental, iterative processes, as shown in Figure. Guidelines are given for the amount of time that should be spent in each of these steps, constraining the amount of time spent in overall planning and architecture and emphasizing the amount of time designing and building features. Processes 1 through 3 are done at the start of a project and then updated throughout the development cycle. Processes 4 and 5 are done incrementally on two week cycles. Each of these processes has specific entry and exit criteria, whereby the entry criterion of Process N is the exit criteria of Process N-1.

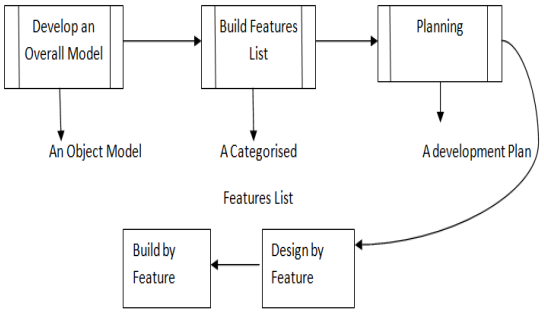

Figure 2: Overview of Feature Driven Development

Process 1: Domain and development team members work together to understand the scope of the system and its context. High-level object models/class diagrams are developed for each area of the problem domain. Model notes record information about the model's shape and why some alternatives were selected and others rejected.

Process 2: Complete list of all the features in the project; functional decomposition which breaks down a "business activity" requested by the customer to the features that need to be implemented in the software.

Process 3: A planning team consisting of the project manager, development manager, and chief programmer plan the order in which features will be developed. Planning is based on dependencies, risk, complexity,
workload balancing, client-required milestones, and checkpoints. Business activities are assigned month/year completion dates. Every class is assigned to a specific
developer. Features are bundled according to technical reasons rather than business reasons.

Process 4: The chief programmer leads the development of design packages and refines object models with attributes. The sequence diagrams are often done as a group activity. The class diagrams and object models are done by the class owners.Domain experts interact with the team to refine the feature requirements. Designs are inspected.

Process 5: The feature team implements the classes and methods outlined by the design. This code is inspected and unit tested. The code is promoted to the build. [6] Progress is tracked and made visible during the Design by feature/Build by feature phases. Each feature has six milestones, three from the Design by feature phase (domain walkthrough, design, and design inspection) and three from the Build by feature phase (code, code inspection, promote to build). When these milestones are complete, the date is placed on the Track by Feature chart which is prominently displayed for the team. When a feature has completed all six milestones, this completion is reflected on the "Burn Up" chart. All features are scoped to be completed within a maximum of two weeks, including all six mile stones.

Agile Methodology	Distinguishing Factor
Extreme Programming	1. Intended for 10-12 co-located, object-oriented programmers 2. Minimal archival documentation 3. Rapid customer and developer feedback loops
Crystal	1. Methodology dependent on size of team and criticality of project 2. Emphasis of face-to-face communication 3. Consider people, interaction, community, skills, talents, and communication as first-order effects 4. Start with minimal process and build up as absolutely necessary
Scrum	1. Project management wrapper around methodology in which developer practices are defined. 2. Daily Scrum meeting of Scrum Team. 3. Burndown chart to display progress.
Feature Driven Development	1. Scalable to larger teams 2. Highly-specified development practices 3. Five sub-processes, each defined with entry and exit criteria 4. Two-week features

2.5. Lean-development:

Lean development: is an iterative methodology which is developed by Mary and Tom Poppendieck.Lean Software Development owes much of its principles and practices to the Lean Enterprise movement; this also focuses the team on delivering value to the customer, and on the efficiency of the "Value Stream," the mechanisms that deliver that Value.
The main principles of Lean include:

1. Amplifying Learning
2. Deciding as Late as Possible
3. Delivering as Fast as Possible
4. Empowering the Team
5. Building Integrity In
6. Seeing the Whole

Lean development eliminates waste through such practices as selecting only the truly valuable features for a system, prioritizing those selected, and delivering them in small batches. It emphasizes the speed and efficiency of development workflow, and relies on rapid and reliable feedback between programmers and customers. Lean uses the idea of work product being "pulled" via customer request. It focuses decision-making authority and ability on individuals and small teams, since research shows this to be faster and more efficient than hierarchical flow of control. Lean also concentrates on the efficiency of the use of team resources, trying to ensure that everyone is productive as much of the time as possible. It concentrates on concurrent work and the fewest possible intra-team workflow dependencies.

Advantages:
1. The elimination of waste leads to the overall efficiency of the development process.
2. Delivering the product early
3. Empowerment of the development team helps in developing the decision making.

Disadvantages:
1. The project is highly dependent on cohesiveness of the team and the individual commitments of the team members.

2.6. Dynamic System Development Method:

The DSDM, Dynamic System Development Method is a blend of, and extension to, rapid application development and Iterative development practices. Martin Fowler, one of the writers of Agile Manifesto, believes, “DSDM is notable for having much of the infrastructure of more mature traditional methodologies, while following the principles of the agile methods approach”. The fundamental idea behind DSDM is to fix time and resources, and then adjust the amount of functionality accordingly rather than fixing the amount of functionality in a product, and then adjusting time and resources to reach that functionality. DSDM consists of five phases

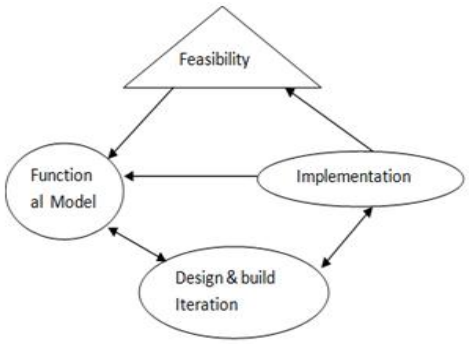

Figure: DSDM Process

Feasibility Study: In this phase a decision is made whether to use DSDM or not. This is determined by judging the type of project and, organizational and people issues. In addition, two work products are produced; a feasibility report and an outline plan for development.

Business Study: The recommended approach to this phase is to organize a workshop to help understand the business domain of the project. The key outputs of this section are System architecture definition and an Outline prototype plan.

Functional Model Iteration: First iterative phase. This phase involves analysis, coding and prototypes. The results gained from these prototypes are used in improving the analysis models. The key output is a functional model which consists of the prototype code and analysis models.

Design and Build Iteration: The system is mainly built in this phase. The design and functional prototypes are reviewed by the users and further development is based on the users' comments.

Implementation: In this final phase the system is handed over to the users. Training is provided. User Manuals and a Project Review Report. However, the DSDM iterative and incremental nature means that maintenance can be viewed as continuing development. Instead of finishing the project in one cycle, the project can return to any of the phases, Design and Build phase, Functional Model Iteration, or even Feasibility phase so that previous steps can be refined.

Conclusion:

Agile methodology is a sound choice for software development and web design projects. This paper explains agile methods, advantages and disadvantages of each method. These methodologies exhibit optimum results when there are a strong communication between the developer and the customer. While designers and developers look at the world from different viewpoints, the Agile philosophy is flexible enough to sustain the approaches and views of both professions. There are still ways to fit quality design work into an agile world.

References:

[1]http://en.wikipedia.org/wiki/Agile_software_development, "Agile software development".

[2] Sheetal Sharma,"Agile Processes and Methodologies: A Conceptual Study", ISSN, 05 May 2012.

[3] Van de Vyer.,Koronois.,&Lane (2003*).Agile methodologies and the emergence of the agile organization: A software development approach waiting for its time?.7 th Pacific Asia Conference on Information Systems,10-13July 2003, Australia, Page 1344-1358*

[4] J. Highsmith, *Agile Software Development Ecosystems*. Boston, MA: Addison-Wesley, 2002.

[5] B. Boehm, "Get Ready for Agile Methods, with Care," *IEEE Computer*, vol. 35, no. 1, pp. 64-69, 2002

[6] F. P. Brooks, *The Mythical Man-Month, Anniversary Edition*: Addison-Wesley Publishing Company, 1995

[7] K. Schwaber and M. Beedle, *Agile Software Development with SCRUM*. Upper Saddle River