Volume-3, Issue-7, July-2014 • ISSN No 2277 - 8160

Research Paper | Computer Science

Implement Agility to Improve Today's Business in Software Developing Companies

Hemina C. Bhavsar Lecturer in S.S.Agrawal College Of Computer Science, Navsari-396445.

ABSTRACT

Today's software development companies deliver software in a shortest time. Software systems that once took years to deliver can now be created in months. The enabler of this transformation is the agile software process. As changes arise in software, developer has to implement changes in a correct manner. For impalement changes agility is the most important method. This paper shows actual meaning of agility. I had written why agility necessary in development. This paper also shows agile process, agile methods and advantages. Developers who apply the agility into development have to follow some agility principles which are also mention in this paper.

KEYWORDS : XP, FDD, ASD, LSD, AM, AUP, DSDM

1: Introduction to software development

Software development is the computer programming, documenting, and testing involved in creating and maintaining applications and frameworks involved in a software release life cycle and resulting in a software product.

2: What is Agile and Agility?

- Agile is the today's buzzword. Everyone is Agile today. Agile is a set of values,

– Individuals/Interactions
– Working software
– Customer collaboration
– Responding to change.

- Now day's people's mentality is change, they want changes every time. And the thing or product which is able to respond immediately when user wants change that is called agile.

- Agility is the strength of the members of the team who appropriately respond to change.

3: Reason to implement Agility into software development

- Every customer needs change in his software, whether software is developing or developed. In fear of scope creep and a never-ending project, we resist changes and put people through a change control committee to keep them to the essential minimum.
- Because the one thing that's certain in life is change. Instead the timescale is fixed and requirements emerge and evolve as the product is developed. Of course for this to work, it's imperative to have an actively involved stakeholder who understands this concept and makes the necessary trade-off decisions, trading existing scope for new.

4: How Agile Process should be?

Any agile software process is characterized in a manner that addresses a number of key assumptions about the majority of software projects:

1. It is difficult to predict in advance which software requirements will persist and which will change. It is equally difficult to predict how customer priorities will change as the project proceeds.

2. For many types of software, design and construction are interleaved. It is difficult to predict how much design is necessary before construction is used to prove the design.

3. Analysis, design, construction, and testing are not as predictable (from a planning point of view).

- Given these three assumptions, an important question arises: How do we create a process that can manage unpredictability?

The answer is that an agile process, therefore, must be ***adaptable***.

- But continual adaptation without forward progress accomplishes little. Therefore, an agile software process must adapt incrementally.
- To accomplish incremental adaptation, an agile team requires customer feedback. An effective catalyst for customer feedback is an operational prototype or a portion of an operational system. Hence, an incremental development strategy should be instituted.
- Software increments (executable prototypes) must be delivered in short time periods so that adaptation keeps pace with change.

5: Agility Principles

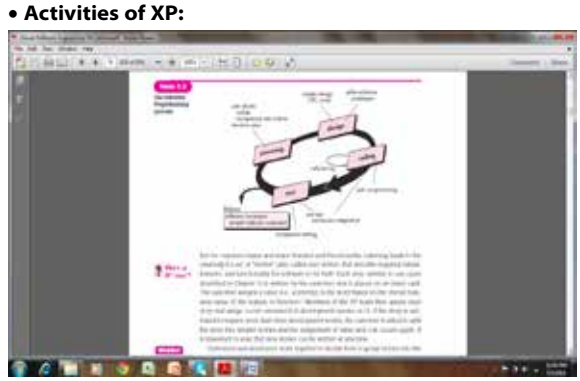
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals.
6. Within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development.

6: Agile Methods

A. Extreme Programming

- Extreme Programming (XP) developing methodology suitable for "object-oriented projects using teams of dozen or fewer programmers in one location."
- The methodology based upon five underlying values: communication, simplicity, feedback, courage, and respect.

• Activities of XP:



(I) Planning:

The planning activity begins with ***listening*** requirements that is gathering activity that enables the technical members of the XP team to understand the business context for the software. Listening leads to the creation of a set of "stories" that describe required output, fea-

tures, and functionality for software to be built. Each *story* is written by the customer and is placed on an index card. The customer assigns a *value* (priority) to the story based on the overall business value of the feature or function. Members of the XP team then assess each story and assign a *cost* measured in development weeks to it. If the story is estimated to require more than three development weeks, the customer is asked to split the story into smaller stories and the assignment of value and cost occurs again.

(II) Designing

XP design rigorously follows the KIS (keep it simple) principle. The design of extra functionality is discouraged. XP encourages the use of CRC cards as an effective mechanism for thinking about the software in an object-oriented context. CRC (class-responsibility collaborator) cards identify and organize the object-oriented classes that are relevant to the current software increment.

(III) Coding

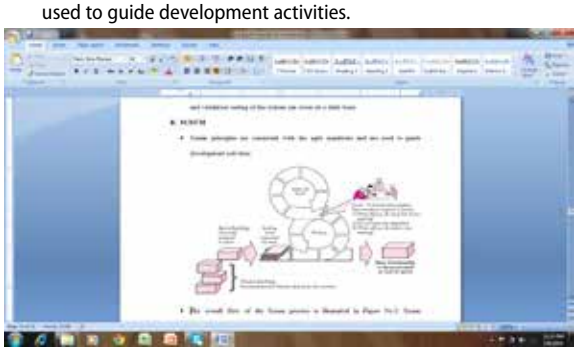
- After stories are developed and preliminary design work is done, the team does not move to code, but rather develops a series of unit tests that will exercise each of the stories that is to be included in the current release (software increment). Once the unit test has been created, the developer is better able to focus on what must be implemented to pass the test.
- A key concept during the coding activity (and one of the most talked about aspects of XP) is pair programming. XP recommends that two people work together at one computer workstation to create code for a story. This provides a mechanism for real time problem solving and real-time quality assurance. It also keeps the developers focused on the problem at hand.

(IV) Testing

- The unit tests that are created should be implemented using a framework that enables them to be automated. This encourages a regression testing strategy whenever code is modified known as XP refactoring philosophy.
- As the individual unit tests are organized into a "universal testing suite" , integration and validation testing of the system can occur on a daily basis.

B. SCRUM

- Scrum principles are consistent with the agile manifesto and are used to guide development activities.



- The overall flow of the Scrum process is illustrated in Figure No.2. Scrum emphasizes the use of a set of software process patterns:
- Backlog—a prioritized list of project requirements or features that provide business value for the customer. Items can be added to the backlog at any time. The product manager assesses the backlog and updates priorities as required.
- Sprints—consist of work units that are required to achieve a requirement defined in the backlog that must be fit into a predefined time-box14 (typically 30 days).
- Scrum meetings—are short (typically 15 minutes) meetings held daily by the Scrum team. Three key questions are asked and answered by all team members:
- What did you do since the last team meeting?
- What obstacles are you encountering?
- What do you plan to accomplish by the next team meeting?
- A team leader, called a Scrum master, leads the meeting and assesses the responses from each person. The Scrum meeting helps the team to uncover potential problems as early as possible.
- Demos—deliver the software increment to the customer so that functionality that has been implemented can be demonstrated and evaluated by the customer.

C. Crystal

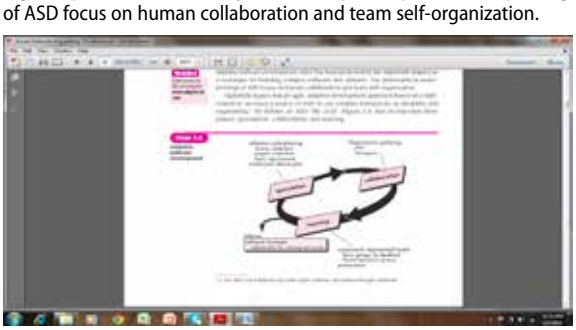
Crystal method is used to achieve a software development approach that puts a premium on "maneuverability" that is "a resource limited, cooperative game of invention and communication, with a primary goal of delivering useful, working software and a secondary goal of setting up for the next game".

D. Future Driven Development

Like other agile approaches, FDD adopts a philosophy that (1) emphasizes collaboration among people on an FDD team; (2) manages problem and project complexity using feature-based decomposition followed by the integration of software increments, and (3) communication of technical detail using verbal, graphical, and text-based means.

E. Adaptive Software Development

Adaptive Software Development (ASD) has been proposed for building complex software and systems. The philosophical underpinnings of ASD focus on human collaboration and team self-organization.



- During speculation, the project is initiated and adaptive cycle planning is conducted. Adaptive cycle planning uses project initiation information the customer's mission statement, project constraints and basic requirements to define the set of release cycles (software increments) that will be required for the project.

- Motivated people use collaboration in a way that multiplies their talent and creative output beyond their absolute numbers.
- As members of an ASD team begin to develop the components that are part of an adaptive cycle, the emphasis is on "learning" as much as it is on progress toward a completed cycle.
- ASD teams learn in three ways: focus groups, technical reviews, and project postmortems.

F. DSDM

The *Dynamic Systems Development Method* (DSDM) is an agile software development approach that "provides a framework for building and maintaining systems which meet tight time constraints through the use of incremental prototyping in a controlled project environment". The DSDM philosophy is borrowed from a modified version of the Pareto principle "80 percent of an application can be delivered in 20 percent of the time it would take to deliver the complete (100 percent) application". DSDM is an iterative software process in which each iteration follows the 80 percent rule. That is, only enough work is required for each increment to facilitate movement to the next increment.

G. Lean Software Development (LSD)

***Lean Software Development* (LSD)** has adapted the principles of lean manufacturing to the world of software engineering. The lean principles that inspire the LSD process can be summarized as eliminate waste, build quality in, create knowledge, defer commitment, deliver fast, respect people, and optimize the whole.

H. Agile Modeling (AM)

There are many situations in which software engineers must build large, business critical systems. The scope and complexity of such systems must be modeled so that (1) all Constituencies can better understand what needs to be accomplished, (2) the problem can be partitioned effectively among the people who must solve it, and (3) quality can be assessed as the system is being engineered and built.

I. Agile Unified Process (AUP)

The ***Agile Unified Process*** **(AUP)** adopts a "serial in the large" and "iterative in the small" philosophy for building computer-based systems. By adopting the classic UP phased activities: ***inception, elaboration, construction,*** **and** ***transition***.

7: Agile Advantages and disadvantages

(I) Revenue
(II) Speed-to-market
(III) Quality
(IV) Visibility
(V) Risk Management
(VI) Cost
(VII) Business Engagement/Customer Satisfaction
(VIII)More Enjoyable!

8: Conclusion

Agility is today's required concept which we must have to implement in development of software for faster completion of software with good quality. Instead of big specs, we discuss requirements in workshops. Instead of lengthy status reports, we collaborate around a taskboard discussing progress. Instead of long project plans and change management committees, we discuss what's right for the product and project and the team is empowered to make decisions. In my experience this makes it a much more rewarding approach for everyone. In turn this helps to create highly motivated, high performance teams that are highly cooperative.

REFERENCES

1. The Characteristics of Agile Software Processes, Granville G. Miller, rmiller@togethersoft.com, TogetherSoft | 2. Essential Scrum Presented by Tobias Mayer, 11/7/2008, at Baldwin-Wallace College Professional Development, co-sponsored by Cleveland Scrum Alliance & NEOPMI | 3. An Introduction to Agile Software Development, June 2007, serena.com | 4. Engineering of Unstable Requirements Using Agile Methods , James E. Tomayko | Carnegie Mellon University, jet@cs.cmu.edu | 5. Impact of Agile Software Development on Quality within Information Technology Organizations Robert Imreh , Mahesh S. Raisinghani, ISSN 2079-8407 | 6. http://www.allaboutagile.com/10-good-reasons-to-do-agile-development/#sthash.qGFWREkv.dpuf | 7. Software Engineering, A Practitioner's Approach, Roger S. Pressman, Ph.D. |