Agile Methodology Adoption: Benefits and Constraints

Radha Shankarmani
Information Technology Department
Sardar Patel Institute of Technology Mumbai, India

Renuka Pawar
Information Technology Department
Sardar Patel Institute of Technology Mumbai, India

S. S. Mantha, PhD.
Chairman
AICTE
New Delhi, India

Vinaya Babu, PhD.
Principal & Professor
CSE JNTU
Hyderabad, India

ABSTRACT
Agile philosophy is to deliver working versions of the software in short iterations, then update the software according to customers' feedback. Applying this philosophy will help to overcome the problems faced in traditional development. By welcoming changes and involving the customer when planning for iteration helps to improve the quality of the product, enables faster development, and in the end, users will have just the system they need. This paper also discusses some of the potential risk and constraints in agile adoption.

General Terms
Agile methodology.

Keywords
Agile manifesto; scrum; traditional software development; agile adoption; risk.

1. INTRODUCTION
Agile practices have recently gained popularity among a large number of companies as a mechanism for realizing early return on investment (ROI) and increasing ability to handle changes in dynamic market conditions. Agility in short means to reduce as much of the heaviness commonly associated with traditional software development methodologies, as possible, in order to promote quick response to changing environments, changes in user requirements**,** accelerate project deadlines, and the like [1]. Agile methodologies prefer software product development over documentation.

Agile methodologies include- Figure 1[14]

- Scrum
- Scrum/XP Hybrid
- Custom Hybrid
- Don't Know
- Kanban
- Scrumban
- Feature-Driven Development
- Extreme Programming XP
- Lean
- Other
- Agile Unified Process (AgileUP)
- Agile Modeling
- Dynamic Systems Development Method

If we compare some process tools on the prescriptive vs adaptive scale: Figure 2 [15]

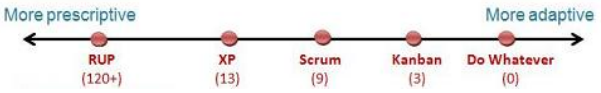


Figure 2: Prescriptive vs Adaptive scale

RUP is pretty prescriptive – it has over 30 roles, over 20 activities, and over 70 artifacts. XP (eXtreme programming) is pretty prescriptive compared to Scrum. It includes most of Scrum + a bunch of fairly specific engineering practices such as test-driven development and pair programming.

Scrum is less prescriptive than XP, since it doesn't prescribe any specific engineering practices. One of the main differences between Scrum and RUP is that in RUP you get too much, and you are supposed to remove the stuff you don't need. In Scrum you get too little, and you are supposed to add the stuff that is missing.

The agile methodology is broken down into 5 steps – Figure 3 [2]

1. In the "Version Scope" phase, the requirements are identified. Even if the scope can evolve throughout the project, the scope is clearly defined at the start. The project terms of reference detail the project costs and benefits, taking into account the implications of identified risks. This first step is important for the specification of the requirements and for setting priorities.
2. Based on the initial plan and the achievements from the previous iterations, the cycle plan describes activities for the next iteration in detail. Each new iteration begins with a planning stage.
3. The Build cycle is realized in an iterative way, making sure that each iteration delivers a concrete and usable result for the user.
4. The Client checkpoint includes a quality review at the end of each iteration. The work done during the iteration is validated.
5. The Post-Version review generates feedback resulting into improvement plans. The effective realization of initial expectations is evaluated.

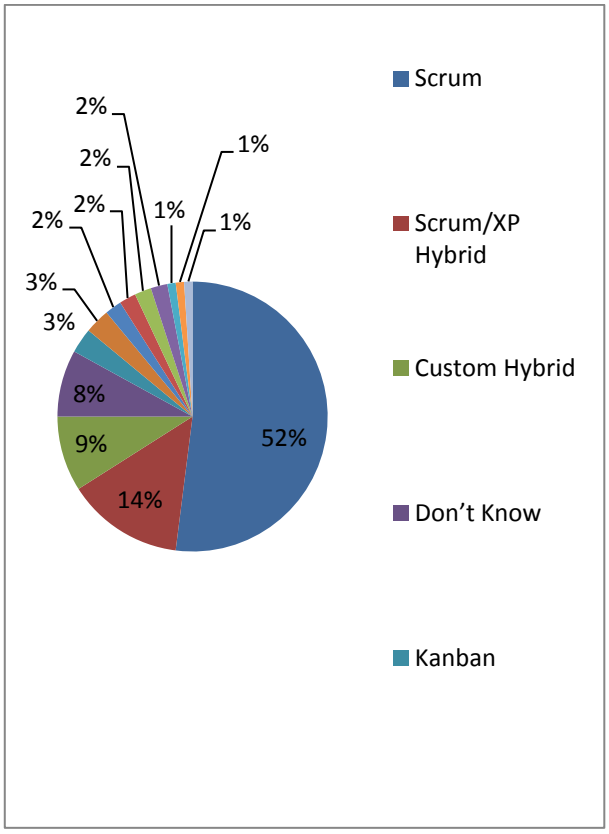


Figure 1 : Agile Methodologies

2. BENEFITS OF AGILE METHODOLOGY

The very first advantage is that the company get to see with the Agile Methodology is the saving of time and money. There is less documentation required, the ones which are used for quick design and development along with the necessary test data. This helps to a great deal in verifying and validating the requirements, also focus more on the application rather than documentation. Since it is iterative in its form, it tends to have a regular feedback from the end user so that the same can be implemented as early as possible. [3]

With the transparency afforded by agile development projects, customers have witnessed stronger results and have benefitted from being provided with real-time updates on the status of development.

By leveraging the agile method, development teams experience a lightweight process that supports a focus on the rapid delivery of business value.

This helps organizations significantly reduce the overall risk associated with development process and ensure that business value is maximized. By continuously aligning the delivered software with desired business needs, teams can easily adapt to changing requirements throughout the project. It can be implemented without any budget constraint since agile manifesto states that the team is self-organizing. [4]
Scrum is an agile project management method. Daily meetings and discussions for the project following this approach can help to determine the issues as early as it arises and work on it accordingly. Quick coding and Testing makes the management aware of the gaps existing in either requirements or technology used and can try to find the ways to work-around the same.

Hence, with the quicker development, testing and constant feedbacks from the user, the Agile methodology becomes the appropriate approach for the projects to be delivered in a short span of time.

2.1 Some characteristic of agile methodology– [5]

- Deliver frequently.
- More iteration.
- Less defects.
- Test frequently.
- Collaborative approach.
- Maximum ROI

3. AGILE ADOPTION CONSTRAINTS

Constraints on adoption of agile software development exist in an organization for a variety of reasons. Such challenges are likely to occur in large organizations in which the stakeholder population is high and interaction among departments and teams is heavy.

3.1 Some of the constraints are given below [6]

- Agile skills within the group are at a novice level, and taking on all the agile practices in one swoop is too challenging. Therefore, the team has to follow staging adoption.
- There are constraints imposed by the organization that prevent certain agile practices from being adopted.
- The team is sticking to (or holding on to) old ways or being half-hearted about Agile adoption, or is wrongly adopting Agile practices.
- Ability to change organizational culture[8]
- Availability of personnel with right skills
- General resistance to change
- Management support
- Project complexity
- Confidence in ability to scale
- Customer collaboration
- Perceived time to transition
- Budget constraints

3.2 Agile methods have three major potential risks: [7]

- Agile methods are easy to misunderstand.
- It's easy to think you're doing Agile right, and be wrong.
- Agile methods make value visible which is embarrassing for the team but adds value to the customer.

3.2.1 Agile Methods Are Easy To Misunderstand

Agile philosophy includes heavy customer involvement, responding to change versus following a plan, releasing software early and often, and focusing on individuals and interactions over processes and tools. Those Agile philosophies tie into agile methods such as pair programming, the planning game, time-boxed iterations, test-driven development, and refactoring. (XProgramming.com is a wonderful resource on agile methods.)

The methods exist in order to enable the philosophy. For example, test-driven development combined with refactoring

can make software malleable so it can change more easily, allowing the plan to change. Automated testing combined with continuous integration makes it possible to release often. Without automated testing and continuous integration, frequent releases create a huge manual testing burden that's often unbearable.

Agile practices are like any other practices, though they're learned through example, application, and training. If the practices are disconnected from the philosophy, the result just won't work. For example, if a company that wanted to be more "agile," would still want to know exactly when projects would start and stop, before the requirements were defined. That's a fine idea—in fact; agile development can without automated testing—with the result that the testing burden grows exponentially—or move to iterative development but keep the hard deadlines and the expectation of full delivery on a ship date. Pair programming without communication and compatibility is just someone breathing over your shoulder. Without testing skills, a developer won't be able to do automated testing; worse, he won't even realize that he has wasted hours of writing "test scripts." support this plan for a single iteration.

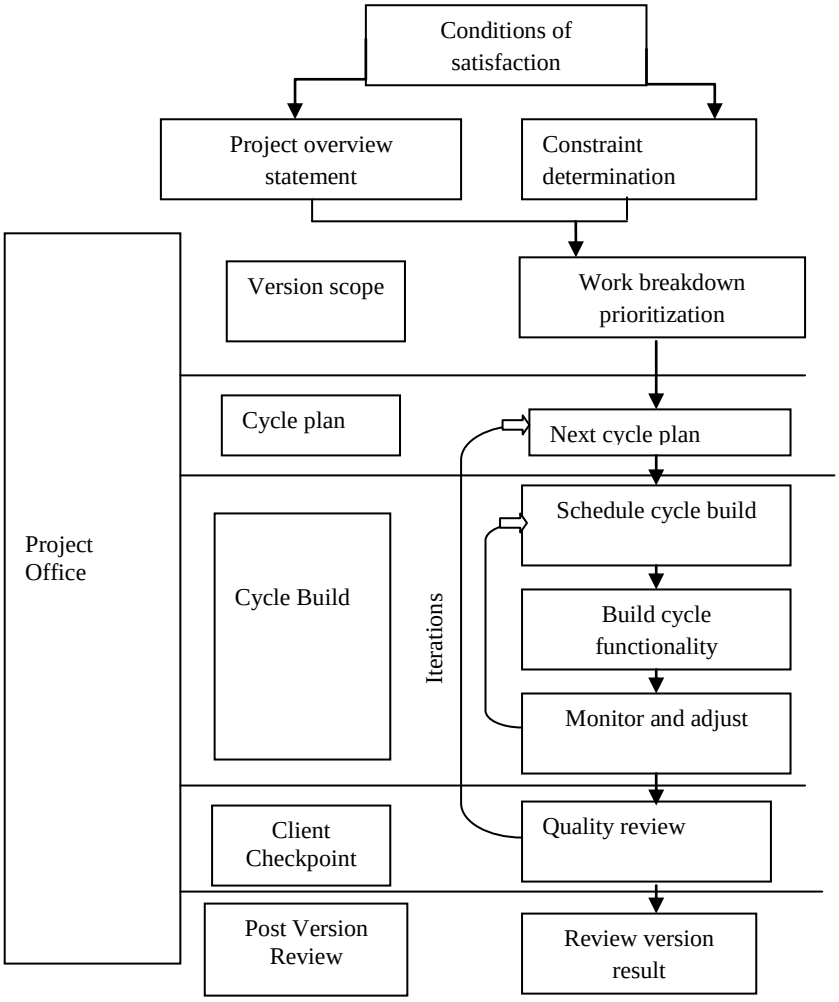


Figure 3: Agile structure

Features are placed in a priority order; the team works on the highest priority first, iterating until time runs out (time-boxed iteration). The problem occurs when the organization wants an estimate for all the work to be done—before they knew all the requirements.

3.2.2 You Have Agile Principle In Mind But Do Not Practice

It's really not so easy to throw out the long time practiced methodology and the heavyweight requirements documents, but agile development expects that something will replace them. Many organizations move to iterative development

Agile techniques require good skill set (Domain, technology and fairly good knowledge of agile practices): the ability to know the right techniques for the current project, and the ability to choose among them. Without direction, if a team was asked to throw away its waterfall method it will simply devolve into "code and fix". That's chaos and not agile.

3.2.3 Agile Methods Make Value Visible

This is the idea that leads to a potential risk of agile adoption. It's actually a political risk. In most large organizations, employees do not want to be accountable for everything they do each day. This creates lot of pressure and resistance among employees.

Agile methods make progress visible. Unlike the complex, bureaucratic waterfall organization, which is opaque, if someone in an Agile team isn't contributing, the fact will come to limelight very fast.

3.3 Concern about adopting Agile – [8]

- Lack of up-front planning
- Loss of management control
- Management opposition
- Lack of documentation
- Lack of predictability
- Lack of engineering discipline
- Dev team opposed to change
- Quality of engineering talent
- Reduced software quality
- Regulatory compliance
- Inability to scale
- No concerns

4. WORKING TOWARDS ADOPTION

No organization can be 100% agile or not agile at all. An organization always moves towards adopting agile methodology.

In this context, few queries were posted to a handful of product development, Service – Application development and Consulting companies.
Mostly the questions were quantitative, but there were also some binary yes-or-no questions and qualitative questions for non-quantifiable variables [9].

4.1 Initial survey

Initial survey was done with few queries. Queries were answered in yes/no format along with some remarks, the summary of it is stated below

Methodology:

- Query: Do you follow agile methodology in your organization?
- Response: Yes. Projects that followed agile-scrum varied from 40%-80%

Scrum Impediments:

- Query: Can you list some of the impediments that you faced related to problem domain, solution domain, technology and tools?
- Response: Most of them were Infrastructure problems, Lack of Domain knowledge, unclear business requirements

Skill set:

- Query: Do you get the right skill set of developers for the projects?
- Response: No. Though they preferred a good mix of generalist and specialist;(this question does not arise for consulting companies since they hire contractors, but if client insists on agile methodology they provide product owners from their consulting firm and hence need to have knowledge about agile methods)

Product Backlog:

- Query: What percentage of user stories decided for the current sprint goes to product backlog because of impediments - not being solved?
- Response: Percentage of user stories decided for the current sprint goes to product backlog because of impediments varies from 10%-25%

Documentation optimum for maintenance:

- Query: Is documentation optimum for maintenance? If yes what is the method used?
- Response: Yes. Some of them use in- house development tool or rational tool.

Reusability:

- Query: What percentage of reusable components do you really use?
- Response: Present Ratio is around 5% -10%.

Customer consents for agile:

- Query: Do old customers agree to agile methodology?
- Response: Yes, customers are oblivious of the methodology used.

Reduced scrum:

- Query: Can daily scrum be reduced to twice a week, if in case enough assistance are provided to solve impediments?
- Response: Al most 80% said yes.

Training in Tool / Technology:

- Query: How do you train agile developers in new technology / tool? Is it by pair programming? Do you think if eMentors are provided (not escalating the cost) will it be of help?
- Response: Small training sessions or pair programming is done for tools/technology training; consulting firms hire contractors. E-Mentors are welcome by all of them

Self- organizing and lessons learned:

- Query: Do you look for patterns for solving issues during retrospective meeting? Do you use lessons-learned repository to improve process?
- Response: They look for patterns during retrospective meetings; but the same is not automated since most of them are caught by team's senior members by experience.

4.2 Macro level factors for agile project

Table 1 represents some macro level factors to increase the quality of agile project. Some questions based on organization involvement, proficiency of team. If tem player is present in the project then give reward to encourage the person.[11] To increase the productivity observes the matured story point and to increase the technology maturity organize the expert for it.

In agile involvement of customer does matter. So to increase the quality find whether the customer representative is IT savvy. Also need to consider the tool training provided as a quality factor [10]

Table 1: Macro level factors for Agile project.

	Macro level factors
MLF.1	How has Organizations Shown an increased concern with quality?
MLF.2	How much proficiency of Team skill set?
MLF.3	How many Team players are organized in a team?
MLF.4	How many maturated requirements of user story?
MLF.5	How to organize expert for technology maturity?
MLF.6	How Tool training is provided?
MLF.7	How can expert will handle multiple roles?
MLF.8	Is Customer representative an IT savvy?
MLF.9	How to get maturity for requirement architecture?
MLF.10	How to make daily standup effective?

4.3 Agile development assessment questions

Table 2 represents the answers to these questions with a corresponding point scale to reflect the characteristics above. Development Table 2 presents questions that identify evidence for best practices in agile development. A well working agile development process has the following characteristics:
[11]Matured requirement for each user story, then how the splitting is done for each subtask. After release find the features that need to be pushed back to the product backlog for each user story. Estimation to each user story and source code committed to the version control.
The development process should consist of frequent, time-boxed iterations that deliver code that is ready to deploy to the customer at the end of each iteration.

The co-operation between the development and the testing teams should be close. Code should be committed to the version control system at least nightly and a new build of the software should be available for testing daily.[12]
Table 2 presents the answers to the questions with a corresponding point scale to reflect the characteristics discussed above.

Table 2: Agile development assessment questions

DEV.1	How many matured requirement of user story do you have?
DEV.2	How many subtasks is each feature split into?
DEV.3	In general, how many features must be pushed back to the product back-log at the end of each user story?
DEV.4	How many open bugs do you have for this user story?
DEV.5	Is the estimation of user story is right or not?
DEV.6	How often is source code committed to the version control system?

4.4 Agile testing assessment questions

The questions in table 3 identify evidence for best practices for agile testing. A well working agile test process has the following characteristics:

- The system is proven to work by passing repeatable tests that the customer helps to specify.
- Tests should be updated and run constantly on daily builds of the system. The tests should preferably be automated.
- Communication between the testers and developers should be seamless, but the testing process itself should be autonomous - development should not interfere with how the tests are run on a feature.
- All features must pass the test cases before they are delivered to the customer. If a feature fails a test, the feature is not shipped and it is pushed back to the product backlog.

Table 3: Agile testing assessment questions

	Assessment question for testing
TST.1	What is the velocity of sprint for User story point?
TST.2	How many are ready for release?
TST.3	How many issues are reported for user story?
TST.4	How many issues go to product backlog??
TST.5	For how many issues solution is found?

5. Survey Chart [16]

In the survey, we first found problems can occur to any degree in the dimensions for adoption. We conducted the survey for three different projects. Survey was carried out based on the different queries and the result of which is shown in the figure 4. In table 5 some suggestions are provided to make agile adoption more feasible.

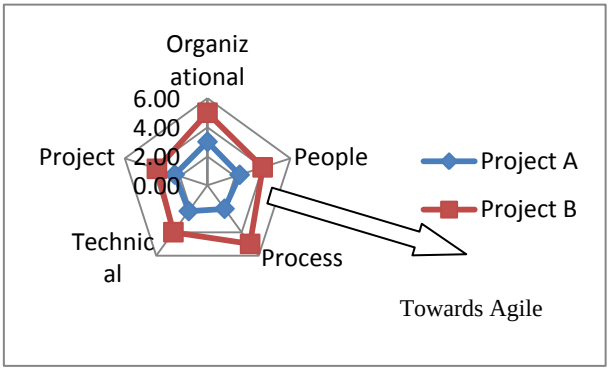
Table 4: Survey Chart

Dimensions	Factors
Organizational	**O01**. Effectiveness of product owner.
	O02. Management commitment.
	O03. Organizational size.
	O04. Working on agile logistical arrangement.
	O05. Organizational culture
People	**P01**. Necessary skill-set
	P02. Project management competence
	P03. Team work
	P04. Team Player
	P05. Product owner relationship
Process	**PR01**.Defined project scope
	PR02. Defined project requirements
	PR03. Defined project planning
	PR04. Agile progress tracking mechanism
	PR05. Customer participation.
Technical	**T01**.Set of correct agile practices.
	T02.Appropriateness of technology and tools
	T03.Regular delivery of software
	T04.Delivering most important features first
	T05.Appropriate technical training to team
Project	**PRJ01**.Project type being of variable scope with emergent requirement
	PRJ02.Projects with dynamic, accelerated schedule
	PRJ03. Projects with small team
	PRJ04. Projects with up-front cost evaluation done
	PRJ05. Projects with up-front risk analysis done

Table 5 some suggestions towards agile adoption

Sr no.	Queries	Suggestions
1.	Impediments that you faced are they related to problem domain, solution domain, technology and tools?	Provide enough resources to support and improve skill set of team
2.	Increased concern of Organizations with quality?	Use of refactoring tools
3.	Deficiency of skill set of developers for the projects?	Pair programming
4.	Can daily scrum be reduced to twice a week, if in case enough assistance is provided to solve impediments?	Use e-assist to help developers solve impediments.
5.	Team players are organized in a team?	Frequently change the pair and role
6.	User story well understood?	Quality person should tag to business analyst to avoid deficiency in non functional requirements
7.	Is Product owner IT savvy?	In consulting company, product owner is consultant, query will not arise. For others prototype/demo can help.
8.	Making daily standup effective?	Create general folder for assist to put all references like FAQs which can be shared by team,.
9.	Feature split into subtask?	May require scrum of scrum
10.	How many product backlogs at the end of each user story?	Time line can go wrong if estimation go wrong. Involve team while estimation.
11.	Source code committed to the version control system?	If release has bug, task is taken to next iteration
12.	What is the velocity of sprint for User story point?	Initially allow this to progress

Figure 4: Moving Towards agile adoption



Project A the application which is less agile e.g. military where, there is always a strong traceability from tests to requirements and from requirements to specifications that can be automatically checked in both directions

Project B is more agile the online cloths store, which sells their product across the city. This faces extremely volatile requirements with major changes week by week.

6. CONCLUSION

Agile methodology is followed by small project teams which are co-located since it is easy to coordinate and work as a team much better than a large distributed team**.** Organizations that are trying to adopt agile methods for the first time should be aware of the risks if agile is not well understood because agile has not set any standard process to follow as agile manifesto says "people over process". The team decides on the process to be followed and is self-organizing.

7. ACKNOWLEDGEMENT

Our sincere thanks to all the companies that responded to the agile-scrum queries posted to them. We would like to convey our sincere thanks to the contributors of agile articles and their various blogs which gave us an exposure of industry practices on agile methodology

8. REFERENCES

[1] J. Erickson, K. Lyytinen and K. Siau, Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research. In Journal of Database Management, 16(4), 2005, 88-100.

[2] http://www.qualified-auditpartners.be/index.php?cont=767&lgn=2

[3] http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-agile-software-development.html

[4] http://softlinksolutions.com/development_methodology.aspx

[5] http://www.isoftwaretesting.com/2012/01/what-is-agile-testing-advantages-of.html

[6] http://ovum.com/2011/09/19/enterprise-agile-constraints-implications-of-scrum-but-and-agilefall/

[7] http://www.informit.com/articles/article.aspx?p=441115

[8] http://www.versionone.com/state_of_agile_development_survey/11/

[9] http://softlinksolutions.com/development_methodology.aspx

[10] http://epf.eclipse.org/wikis/scrum/Scrum/guidances/supportingmaterials/scrum_overview_610E45C2.html

[11] http://www.ambysoft.com/essays/agileLifecycle.html

[12] http://smalldogsoft.com/wordpress2/?p=368

[13] http://www.versionone.com/state_of_agile_development_survey/11/

[14] http://www.infoq.com/minibooks/kanban-scrum-minibook

[15] A survey study of critical success factors in agile software projects by Tsun Chow, Dac-Buu Cao published in the Journal of Systems and Software 81 (2008) 961–971