Chapter 4

RESULTS AND DISCUSSION

This chapter presents and discusses the findings of the research done on agile development in Indian organizations. In order to achieve its objective this chapter discusses the difference between heavyweight and agile software development in Indian organizations and analyzes the results obtained.

4.1 Result analysis

Present research work emphasize on an important question that what is agile? Which has been explained in the Chapter-1. Another important aspect about the difference between different agile methods is described below (Table 4.1):

<table>
<thead>
<tr>
<th>S. No</th>
<th>Name of agile methodology</th>
<th>Features</th>
<th>Drawbacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Extreme Programming</td>
<td>Customer driven development, small teams, daily builds.</td>
<td>While individual practices are suitable for many situations, overall view and management practices are less attractive.</td>
</tr>
<tr>
<td>2.</td>
<td>Scrum</td>
<td>Independent, self-organizing and small development teams, 30 day release cycle.</td>
<td>Integration and acceptance tests are not detailed.</td>
</tr>
</tbody>
</table>
3. **Dynamic System Development**
   - Application of controls to RAD, use of timeboxing and empowered DSDM teams. First truly agile software development method, use of prototyping.
   - While the method is available only consortium members have access to white papers dealing with actual use of the method.

4. **Feature Driven Development**
   - 5 step process, object oriented component based development, method simplicity, design and implement the system by features, object modeling.
   - It focuses only on design and implementation, it needs other supporting approaches as well.

5. **Lean Software Development**
   - Focuses on people and communication, follow 7 principals, elimination of waste reduces project time and cost.
   - Decisions have to be made promptly.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td>Dynamic System Development</td>
<td>Application of controls to RAD, use of timeboxing and empowered DSDM teams. First truly agile software development method, use of prototyping.</td>
</tr>
<tr>
<td>4.</td>
<td>Feature Driven Development</td>
<td>5 step process, object oriented component based development, method simplicity, design and implement the system by features, object modeling.</td>
</tr>
<tr>
<td>5.</td>
<td>Lean Software Development</td>
<td>Focuses on people and communication, follow 7 principals, elimination of waste reduces project time and cost.</td>
</tr>
</tbody>
</table>

Table 4.1 Differences Between Various Agile Methods

Every method has its weakness and strengths. So is with agile and older methods. One important aspect was “To identify the difference between agile and heavyweight methods in Indian organizations”. For this researcher first tried to
compare it with different models and then tried to find common differences between the two. The comparison with different models is discussed below:

4.1.1 Agile v/s Waterfall Model

Adoption of Waterfall has helped to drive down the failure rate of software development projects, but even with rigorous project management and processes, a full 70 percent of software projects using this methodology fail to meet their objectives. Organizations tried to cut the failure rate by insisting on more detail in the requirements and design phases. This process of requiring extensive, even exhaustive, documentation culminated in 1988 with the publication of the Department of Defense Standard for software development. One of the most important differences between the agile and waterfall approaches is that Waterfall features distinct phases with checkpoints and deliverables at each phase, while agile methods have iterations rather than phases. The output of each iteration is working code that can be used to evaluate and respond to changing and evolving user requirements [41].

Waterfall assumes that it is possible to have perfect understanding of the requirements from the start. But in software development, stakeholders often don’t know what they want and can’t fix their requirements. With waterfall, development rarely delivers what the customer wants even if it is what the customer asked for [104]. Agile methodologies embrace iterations. Small teams work together with
stakeholders to define quick prototypes, proof of concepts, or other visual means to
describe the problem to be solved. The team defines the requirements for the
iteration, develops the code, and defines and runs integrated test scripts, and the
users verify the results. Verification occurs much earlier in the development process
than it would be with waterfall, allowing stakeholders to fine-tune requirements
while they’re still relatively easy to change [105].

4.1.1.1 Why are people abandoning waterfall and moving to agile [106]?

1. Agile is adaptive

   For the project team, as well as the business, agile enables us to make quick
changes in right direction so that our software product and our business can respond
to a rapidly changing business environment. Agile teams use two-to-four week
iterations, often called sprints, in which they develop and then release a working
product. At the end of each sprint, the team uses retrospectives to look back on the
work completed and see how productivity can be improved; the team also works
with the customer to determine which work should be accomplished during the next
sprint. One technique enables continuous improvement; the other enables the
business to re-prioritize work based on changes in the business climate. Together,
they make agile highly adaptive as compared to a waterfall approach that
effectively locks the team to both a process and business strategy for a number of
months.
2. **WYSIWYG (What you see is what you get) development**

   In reality, many of developers have a rough idea of what they want and often less of an idea of what their customers want, particularly with software products that serve the masses. Agile takes an entirely different approach to requirements gathering. Product features are identified for development and then the team works together with the business customers to build the features cooperatively. In many cases, user stories are written, screen mockups are drawn and simple business rules are written, but nothing too sophisticated. Instead, the agile team relies upon heavy interaction with the customer or product owner.

3. **Shorter time to market**

   Using a waterfall approach, delivering anything to the marketplace takes months and sometimes years. But, by taking an agile approach, the bare-bones features of a new product can be delivered in weeks, then, further enhanced to provide a truly robust solution. Again, the secret to shorter time-to-market lies in the use of iterations (sprints), with the end of each sprint another opportunity to deliver more features to the customer. Agile has this but Waterfall doesn’t.

4. **Greater employee satisfaction**

   Agile development provides greater employee satisfaction. Research by Grigori Melnick and Frank Maurer from the University of Calgary showed 82% of employees at agile-adopting businesses were satisfied or very satisfied with their jobs, while only 41.2% were satisfied or very satisfied in non-agile shops [107].


5. Higher quality

Adopting agile reduces defects and results in higher product quality. According to a 2008 survey by Version One, 3rd Annual Survey on the State of Agile Development, 68% of respondents to a survey on agile adoption and corresponding results reported improved product quality as one of the benefits. Similarly, according to David Rico’s, report there is 75% improvement in quality by adopting agile [108].

6. Higher ROI

If there’s one single reason for the corner office to be sold on agile, it has to be higher ROI. Because agile reduces project overhead, delivers beneficial work more quickly and produces higher quality products, agile also delivers a higher ROI to the businesses who adopt it.

It’s sometimes hard to understand why waterfall is still practiced. But the truth is, adopting agile takes a paradigm shift in thinking that is not easy for individuals, much less organizations, can make it. It also takes experience not only in practicing agile, but also in managing organizational change.

4.1.1.2 Agile v/s Iterative enhancement model

Iterative model has the same phases as the waterfall model, but with some restrictions. The phases occur in the same order as in Waterfall model, but these
may be conducted in several cycles, with each release providing additional functionality.

Some of the major differences between iterative enhancement model and agile are [109]:

- Each agile team consists of one customer representative. Customer representative is the one who is involved as an end user of the project or he can be product distributor or he can be the representative of the company who will use product. But iterative enhancement model does not consist of customer representative.
- Agile does requires less documentation but iterative method require a lot of documentation.
- Agile methodology requirements can change till the end of the project but in iterative enhancement method once requirements are made they do not change.
- Agile is based on iterative and incremental method both but an iterative method is only based on iterations.
- In agile at the end of each iteration, the iteration does not consist of all functionalities but consists some functionality but they are bug free. In iterative enhancement model all functionalities are provided and if some bug remains then they are removed in further iterations.
• Iterative is predictive whereas agile is adaptive.

• Time required in agile is in weeks whereas in iterative enhancement it is in months.

4.1.1.3 Agile v/s Spiral Model

Spiral model is one of the popular process model used by industry. It is based on evolutionary approach which couples the iterative nature of prototyping with the controlled and systematic aspects of linear sequential model. The project is reviewed and the next phase of the spiral is planned [10]. It was proposed by Bhoem in 1988 and is a popular model used for large size projects. Key difference between agile and spiral method [110][111] are:

• Agile methodology is for smaller projects which consist of smaller teams but spiral model is for larger projects involving larger teams.

• Agile consists of experienced team members but spiral can consist of fresher’s as well.

• There is less documentation in agile but spiral consists of heavy documentation.

4.1.1.4 Agile v/s V-shape Model

V-shape model was developed to relate the analysis and design activities with the testing activities and thus focuses on verification and validation activities
of the product. Main differences between V-shape model and agile methodology are [112] [113] [114]:

- V-shape model does not support iterations of phases and dynamic changes in requirements throughout the lifecycle but agile does support.
- It does not take into account risk analysis but agile does support this.
- Testing cycle time of agile is relatively short compared to V-model, because testing is done parallel to development in agile.
- Agile model delivers a working version of the product very early compared to V-model. As more features are delivered incrementally, customer can realize some of the benefits early on.
- Agile is a proactive model (due to its very short cycles) compared to the much more reactive V-model.
- V-model is very rigid and relatively less flexible than agile model.
- This model does not provide a clear path for problems found during testing phases.

4.1.1.5 Agile v/s Prototyping Model

A prototype is made first and based on its final product is developed. A prototype is a model or a program which is not based on strict planning, but is an early approximation of the final product or software system. A prototype acts as a sample to test the process. From this sample we learn and try to build a better final
product. This prototype may or may not be completely different from the final system we are trying to develop. This type of System Development Method (SDM) is employed when it is very difficult to obtain exact requirements from the customer. While making the model, user keeps giving feedbacks from time to time and based on it, a prototype is made. Completely built sample model is shown to user and based on his/her feedback; the SRS (System Requirements Specifications) document is prepared. After completion of this, a more accurate SRS is prepared, and now development work can start using any model. There are some differences between agile and prototyping model such as [115]:

- Prototyping is usually done at the cost of the developer. So it should be done using minimal resources. It can be done using Rapid Application Development (RAD) tools. Sometimes the start-up cost of building the development team, focused on making prototype, is high.
- Once we get proper requirements from client after showing prototype model, it may be of no use. That is why; sometimes we refer to the prototype as "Throw-away" prototype.
- Prototyping is a slow process.
- Too many changes can disturb the rhythm of the development team.
4.1.1.6 Agile v/s Rapid Application Development (RAD) model

Rapid Application Development (RAD) model is incremental software development process model that emphasizes short development cycles. The Rapid Application Development model was proposed by IBM in 1980’s and later on was introduced to software community by James Martin through his book Rapid Application Development. There are some differences between RAD and agile such as [116]:

- One of the agile methodologies DSDM uses RAD model. DSDM grew to provide an industry standard project delivery framework for what was referred to as Rapid Application Development (RAD).

- RAD was extremely popular in the early 1990’s, the RAD approach to software delivery evolved in a fairly unstructured manner. As a result, the DSDM consortium was created.

Six heavyweight models have been considered for research. More models can be included. An important outcome from the present work is comparison of agile with heavyweight methodology in Indian organizations which is discussed (Table 4.2).
<table>
<thead>
<tr>
<th>S. No</th>
<th>Criteria</th>
<th>Traditional Methods</th>
<th>Agile</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Assumption</td>
<td>Problem is well understood and the desired output is well defined from the beginning.</td>
<td>The desired output is not known completely until the solution is delivered.</td>
</tr>
<tr>
<td>4.</td>
<td>Aim</td>
<td>To deliver software after a defined period.</td>
<td>To deliver software quickly [117].</td>
</tr>
<tr>
<td>5.</td>
<td>Planning</td>
<td>Detailed planning of time-line with clearly defined products and documents to be delivered.</td>
<td>High-level plan for the overall product development life-cycle with details only planned for current iterations.</td>
</tr>
<tr>
<td>6.</td>
<td>Requirement engineering</td>
<td>Clearly defined specification phase; requirements specification of overall product with sign-off; detailed requirements specification often part of the contract, requirements change is a</td>
<td>Welcoming change to requirements specially leading to continuous evolution; relaxed change request process; communication with customer over detailed product specifications.</td>
</tr>
</tbody>
</table>
| 7. Benefits | (a) With frequent requirement changes it cannot work.  
(b) Cannot adopt new ideas in-between stages.  
(c) These do not deliver product in-between production.  
(d) Does not allow specification changes as per user’s requirement. | (a) Can work when requirements constantly changes.  
(b) At the end of each stage it can adopt new ideas.  
(c) Agile team has launchable product at the end of each tested stage.  
(d) Allow specification changes as per user’s requirement. |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8. Team size</td>
<td>Large</td>
<td>Small</td>
</tr>
<tr>
<td>9. Documentation</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>10. Architecture</td>
<td>Specification of architecture and designs is comprehensive and detailed; Architecture design concentrated on one phase. Architecture is designed for current and foreseeable requirements.</td>
<td>Minimal draft of architecture and design specification and re-evaluation of architecture continuously throughout development life-cycle. Architecture is designed for current requirements.</td>
</tr>
<tr>
<td>11. Success Measurement</td>
<td>Confirmation to plan</td>
<td>Business value</td>
</tr>
<tr>
<td>12. Management style</td>
<td>Autocratic</td>
<td>Decentralized</td>
</tr>
<tr>
<td>13. Implementation</td>
<td>Programming work concentrated in one phase and coders concentrate mainly on the programming task; programming is specification driven.</td>
<td>Programming work throughout the entire project; programmers have the possibility to interact with customers; collective code ownership and egoless programming; pair programming</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>15. Emphasis</td>
<td>There exists a defined method, process oriented</td>
<td>Cowboy coding (i.e. absence of defined method), People oriented</td>
</tr>
<tr>
<td>16. Cycles</td>
<td>Limited</td>
<td>Numerous</td>
</tr>
<tr>
<td>17. Return on investment</td>
<td>Late</td>
<td>Early</td>
</tr>
<tr>
<td>18. Risks</td>
<td>Well known risks</td>
<td>Unknown risks</td>
</tr>
<tr>
<td>19. Reviews and inspection</td>
<td>Formal roles in the review process (e.g. inspection); Use of quality doors to approve software artifacts for hand over between phases.</td>
<td>No explicit roles for reviews and inspections; no formal reviews (except in e.g. feature driven development, but not as formal as the inspection process)</td>
</tr>
</tbody>
</table>

Table 4.2 Comparison of Agile with Heavyweight Methods
Frequency tables are also a form of basic analysis. Frequency tables (Annexure III, Table 4.3 to 4.45) show the possible responses and the total number of respondents for each part. Frequency tables are useful when a large number of response options are available. In most cases, pie or bar charts are easier to work with than frequency tables.

*Use of agile practices*

Respondent asked survey respondents to tell about the extent of their usage of various agile practices. The top agile practices that teams followed were Scrum and other (when asked in detail then it was Scrum/XP hybrid). The least followed practices were DSDM and FDD. Results obtained by different organizations are illustrated in upcoming pages.

**Question 1. Size of organization (Small, Medium or Large)?**

The pie chart (Fig. 4.1) presents the diversification of the 70 responders participating in the study based on their organization size. Participants were asked about their organization size. Most of the organizations were of medium size targeting around 40.5%. Moreover 26.08% were of small size and 33.33% of large size.
Question 2. What type of business or organization are you employed in?

According to pie chart (Fig. 4.2) maximum respondents are from Information Technology type organization. IT organizations are 68.57%, telecommunications are 12.86% and others are 11.42%. So it can be concluded that most of people who currently use agile are in Information technology field.
Question 3. Which of the following best describes your position in the organization?

Maximum employees who responded are at position of software engineer (24.29%) and at second position are analyst as well as others (20%) (Fig. 4.3).

![Person's Position in Company](image)

Question 4. Approximately how many software professionals are employed by your organization?

Maximum people replied that number of software professionals does not come in the range that is specified by researcher.
**Question 5.** Do you use any software capability quality standard (such as ISO 9000, SPICE, CMMI)?

75% companies are using software capability standard such as ISO 9000, SPICE OR CMMI (Fig 4.5).
Question 6. *When it comes to adopting new technologies and methods, your company is?*

47.82% companies are market leaders and next highest are market follower which are 30.43%. Moreover 18.84% companies are conservative and only 2% are of static nature (Fig. 4.6).

![Figure 4.6: Company Adopts New Technology or Not](image-url)
Question 7. How would you rate your knowledge of agile methodologies?

Maximum people were having average knowledge of agile which corresponds to 50.72% and after that the people are having extensive knowledge (20.29%) (Fig 4.7).

Fig 4.7 Knowledge of Agile Methodologies
Question 8. How would you rate your knowledge of heavyweight methodologies?

Most of the persons have average knowledge of traditional methods (Fig. 4.8) which corresponds to 49.27%.
Question 9. Whether you are using agile software development methods?

65.71% people were using agile software development methods.

![Pie chart](image)

Fig 4.9 Using Agile or Not

Question 10. Are you satisfied with the current working technology at your organization?

Maximum people were satisfied with the technology they are using (Fig. 4.10).

![Pie chart](image)

Fig 4.10 Satisfied with Current Working Technology
Question 11. *Which agile methodology do you mostly use for small scale software development?*

There are several ASD methodologies available today. Scrum is most liked software development technology that is used by maximum users for small scale software development (Fig. 4.11). The technology changes as the size of software to be developed increases. In India for small scale projects maximum people 37.3% are using Scrum and next highest technology that has been used is hybrid i.e. Scrum/XP hybrid, Custom hybrid, Kanban, Scrumban etc.

![Fig 4.11 Agile Methodology People Mostly Use for Small Scale Projects](image-url)
Question 12. Which agile methodology do you mostly use for medium scale software development?

In medium scale maximum users are of other technologies (Fig. 4.12) which are other than Extreme programming, Scrum, Feature driven development, Dynamic systems development and Lean software development.

Fig 4.12 Agile Methodology People Mostly Use for Medium Scale Projects
Question 13. Which agile methodology do you mostly use for large scale software development?

In large scale projects maximum users are using other technologies, other than XP, Scrum, FDD, DSDM and Lean (Fig. 4.13).

Fig 4.13 Agile Methodology People Mostly Use for Large Scale Projects
Question 14. Which heavyweight methodology do you mostly use for small scale software development?

When it came to use of heavyweight methodology then Waterfall model is the most well known model used by individuals for small scale projects (Fig. 4.14) with 42.85% users.

Fig 4.14 Heavyweight Methodology Mostly Used for Small Scale Projects
Question 15. Which heavy methodology do you mostly use for medium scale software development?

For medium scale projects Spiral is most widely used model (4.15).

![HEAVYWEIGHT METHODOLOGY]

Fig 4.15 Heavyweight Methodology Mostly Used for Medium Scale Projects
Question 16. Which Heavy methodology do you mostly use for large scale software development?

For large sized projects also Spiral model is mostly used (Fig 4.16).

Fig 4.16 Heavyweight Methodology Mostly Used for Large Scale Project
Question 17. Which of listed aspects of agile methodologies most appeal as compared with heavyweight methodologies, for small scale software development projects?

Agile methodology aspect that most appealed as compared to heavyweight methodologies, for small scale software development was respond to change versus following a plan (Fig 4.17).

![Pie chart showing aspects of Agile Methodology Appeal for Small Scale Projects]

- People oriented versus processes oriented: 27
- Working code versus documentation: 16
- Customer relationship versus contract negotiation: 13
- Respond to change versus following a plan: 5
- Other: 5

Fig 4.17 Aspect of Agile Methods Appeal for Small Scale Projects
Question 18. Which of listed aspects of agile methodologies most appeal to you compared with heavyweight methodologies, for medium scale software development project?

Agile methodology aspect that most appealed as compared to heavyweight methodologies, for medium scale software development was respond to change versus following a plan (Fig 4.18).

![Pie chart showing aspects of agile methodologies appealing for medium scale projects](image)

- People oriented versus processes oriented
- Working code versus documentation
- Customer relationship versus contract negotiation
- Respond to change versus following a plan
- Other

Fig 4.18 Aspect of Agile Methods Appeal for Medium Scale Projects
Question 19. Which of listed aspects of agile methodologies most appeal to you compared with heavyweight methodologies, for large scale software development project?

This question tells as that why people are moving towards agile. Maximum responded that changes can be given by stakeholders at any stage in case of agile. So maximum responded that “Respond to change verses following a plan” appeal them most to adopt agile for large as well as for different type of software development (Fig 4.19).

![Aspect of Agile Methods Appeal for Large Scale Project](image)

Fig 4.19 Aspect of Agile Methods Appeal for Large Scale Project
Question 20. Which aspects of agile methodologies, do you dislike the most for different scale software development?

Low planning is disliked most for different scale of software development (Fig 4.20).

Fig 4.20 Aspect of Agile Methods Disliked Most for Different Scale Projects
Question 21. Which aspects of heavyweight methodologies, do you dislike the most for different kinds of software development?

In case of heavyweight methodologies lack of project structure was disliked most for different types of project development (Fig. 4.21).

Fig 4.21 Aspect of Heavyweight Methods Disliked Most for Different Scale Projects
**Question 22.** How do you believe that the cost of employing agile methodologies compares with heavyweight methodologies for small scale software development project?

41% people believe that the cost by employing agile methodologies decreases (Fig. 4.22) in case of small scale project development.

![Pie Chart](image)

Fig 4.22 Cost of Employing Agile V/s Heavyweight in Small Scale Projects
Question 23. *How do you believe that the cost of employing agile methodologies compares with heavyweight methodologies for medium scale software development project?*

In case of medium scale software development, low decrease in costs is at highest level (Fig. 4.23) and second highest is high decrease in cost.

![Fig 4.23 Cost of Employing Agile V/s Heavyweight in Medium Scale Projects](image-url)
**Question 24.** How do you believe that the cost of employing agile methodologies compares with heavyweight methodologies for large scale software development project?

For large scale software development maximum people responded that there is low increase in costs (Fig 4.24). It might be due to bigger project size as well as team size.

![Fig 4.24 Cost of Employing Agile V/s Heavyweight in Large Scale Projects](image-url)
Question 25. Do you believe that taking on agile methodologies rather than heavyweight methodologies have any effect on software quality for different levels of development?

Maximum 38.80% respondents believed that there is high increase in software quality if agile methodologies are used as compared to heavyweight methodologies for different scale of software development.

![Fig 4.25 Effect on Software Quality on Taking Agile Rather than Heavyweight](image-url)
Question 26. What do you believe is the most common problem experienced while practicing agile methodologies for small scale project development?

Most common problem experienced while practicing agile methodologies for small scale project development is lack of skilled people, who can follow agile methodologies with 55.38%, but for large scale projects, project size or project complexity became a dominant factor.

Fig 4.26 Problem Faced While Practicing Agile for Small Scale Projects
**Question 27. What do you believe is the most common problem experienced while practicing agile methodologies for medium scale project development?**

Most common problem experienced while practicing agile methodologies for medium scale project development is lack of skilled people, who can follow agile methodologies with 34.84%.

![Pie chart showing the percentage of issues faced while practicing agile methodologies for medium scale projects.](chart.png)

**Fig 4.27 Problem Faced While Practicing Agile for Medium Scale Projects**
Question 28. What do you believe is the most common problem experienced while practicing agile methodologies for large scale project development?

For large scale projects, project size or project complexity became a dominant factor (Fig. 4.28).

Fig 4.28 Problem Faced While Practicing Agile for Large Scale Projects
Question 29. What is the average size of teams that work on software development in small scale project category, in your organization?

For this question maximum 85.93% respondents reply was that maximum team size for small scale project category is 2 to 15 (Fig 4.29).

![Fig 4.29 Average Team Size for Small Scale Projects](image-url)
Question 30. What is the average size of teams that work on software development in medium scale project category, in your organization?

For medium scale 63.49% respondents replied that their team size is 15 to 50 (Fig 4.30).

Fig 4.30 Average Team Size for Medium Scale Projects
**Question 31. What is the average size of teams that work on software development in large scale project category, in your organization?**

For large scale projects maximum 47.69% respondents replied that their team size is 50 to 200 team members (Fig 4.31).

![Fig 4.31 Average Team Size for Large Scale Projects](image-url)

- 2 to 15 team members: 6%
- 15 to 50 team members: 3%
- 50 to 200 team members: 43%
- More than 200 team members: 48%
Question 32. What do you believe is most suitable methodology for small scale software development?

70.14% respondents believed that most suitable methodology for small scale software development is agile (Fig. 4.32).

Fig 4.32 Most Suitable Technology for Small Scale Projects
Question 33. What do you believe is most suitable methodology for medium scale software development?

44.77% respondents believe that agile is most suitable methodology in medium scale software development (Fig. 4.33).

Fig 4.33 Most Suitable Technology for Medium Scale Projects
Question 34. What do you believe is most suitable methodology for large scale software development?

46.37% respondents believed that heavyweight is most suitable methodology for large scale software development (Fig 4.34). This is due to the reason that as project size grows, team size also grows and to manage large teams is a difficult job.

Fig 4.34 Most Suitable Methodology for Large Scale Projects
Question 35. Do you use any other methodology, other than agile and heavyweight methodologies for software development?

In total 83.33% respondents are using agile and heavyweight, rest 26.67% are not sure about the technology they are using (Fig 4.35).

Fig 4.35 Other Methodologies for Software Development are Used or Not
Question 36. *To what extent, do you follow different kinds of agile techniques for small scale software development?*

For small scale software development agile is not followed exactly 100%, 75% or 25% but maximum respondents followed different from these exact percentages (Fig 4.36) such as 60%, 65%, 34% etc.

![Pie chart showing distribution of responses for agile techniques follow up](image-url)

**Fig 4.36 To What Extent Agile Techniques for Small Projects Followed**
Question 37. To what extent, do you follow different kinds of agile techniques for medium scale software development?

It is also same with that of small scale project development; here also other response is highest (Fig 4.37).

Fig 4.37 What Extent Agile Techniques for Medium Projects Followed
Question 38. To what extent, do you follow different kinds of agile techniques for large scale software development?

Responses for others are highest here as well (Fig. 4.38).

Fig 4.38 What Extent Agile Techniques for Large Projects Followed
Question 39. Would you like agile to be implemented in your organization?

74.62% respondents replied that they want to implement agile in their organization (Fig. 4.39).

Fig 4.39 Like to Implement Agile in Your Organization
Question 40. *If you are using agile then how effective agile methods are as compared to heavyweight methods in case of productivity?*

In case of productivity 37.87% respondents believed that productivity will be much higher as compared to heavyweight methods on using agile and 30.30% respondents believe that productivity will be somewhat higher on using agile (Fig. 4.40).

![Effectiveness of Agile in Case of Productivity](image_url)

Fig 4.40 Effectiveness of Agile in Case of Productivity
Question 41. *If you are using agile then how effective agile methods are as compared to heavyweight methods in case of quality?*

In case of quality, 33.84% respondents believed that quality is much higher and a little less than this 32.30% believed that quality will be somewhat higher on using agile. So overall we can say that the quality became better on using agile (Fig. 4.41).

![Effectiveness of Agile in Case of Quality](image)

Fig 4.41 Effectiveness of Agile in Case of Quality
**Question 42.** If you are using agile then how effective agile methods are as compared to heavyweight methods in case of business stakeholder’s satisfaction?

Business stakeholder’s satisfaction gets increased on using agile (Fig 4.42).

![Bar chart showing the effectiveness of agile methods in terms of business stakeholder's satisfaction.](image)

**Fig 4.42 Effectiveness of Agile in Case of Stakeholder’s Satisfaction**
Question 43. *If you are using Agile then how effective agile methods are as compared to heavyweight methods in case of cost of system development?*

Maximum respondents were in favour that cost of system development becomes much lower on using agile.

![Figure 4.43 Effectiveness of Agile in Case of Cost of System Development](image)

Another important discussion about the quality, productivity and cost factors using agile are proved using one way ANOVA.
For quality:

Researcher obtained 0.2429 < 2.758, where 0.2429 is calculated value and 2.758 is table value. The results are insignificant at the 5% significance level. There is no evidence that expected values in five groups differ.

So, the calculated value of F is less than the table value then it is insignificant and hypothesis is accepted. In above case the calculated value of the F (0.2429) is less than the table value (2.758) hence it is insignificant at 5% level of significance and Hypothesis is accepted (Annexure I).

For production:

Researcher obtained 1.013 < 2.758 in case of production where 1.013 is calculated value and 2.758 is table value. The results are insignificant at the 5% significance level. There is no evidence that expected values in five groups differ.

The calculated value of F is less than the table value then it is insignificant and hypothesis is accepted. In above case the calculated value of the F (1.013) is less than the table value (2.758) hence it is insignificant at 5% level of significance and Hypothesis is accepted (Chapter 3).

For cost:

Researcher obtained 1.747<2.758 where 1.747 is calculated value and 2.758 is table value. The results are insignificant at the 5% significance level. There is no evidence that expected values in five groups differ.
So, the calculated value of F is less than the table value then it is insignificant and hypothesis is accepted. In above case the calculated value of the F (1.747) is less than the table value (2.758) hence it is insignificant at 5% level of significance and hypothesis is accepted (Annexure I).

So, according to findings it has been observed that quality and production gets increased on using agile and cost gets decreases on using agile (Chapter 3, Annexure I).