Chapter 5

PROPOSED WORK

5.1 Introduction
Agile Methodology based development has been widely accepted in both academia and industry for building software systems. Software development with agile methodology is in demand due to rapidly changing requirements, so on the account of this a new agile methodology is proposed. This proposed methodology is a methodology that provides guidance for the major phases to be followed during development of software.

5.2 Proposed Agile Methodology
Different Agile Methodologies have been proposed and exploited in the literature. These various methodologies exhibit various abilities and disabilities to produce a successful software product. But an agile methodology is required which is having the features like independent of project type, according to the situation select its usage based and then develop a plan based on Feature List / Functionality/ Sprint/ Iteration , so that the maximum return on investment can be attained in minimum time.

The proposed methodology exhibits the following features like small team size, more adaptable to changes, having design phase, work week according to the project need, proper documentation and supporting code-refactoring with code ownership.

Proposed Agile Methodology consists of six phases. It starts by estimating the project requirements, size, complexity and risk of the project. Then the usage, team size as well as flexibility is measured. In the third step it is decided which methodology will be suitable for fulfill the above requirements. In the fourth step, how the project should be developed is decided. In the next step, the final product is obtained. In the sixth step, the developed product is handover to the customer and on the basis of customer suggestions implement/review or maintains the system.
The blueprint of the proposed agile methodology is as follows:

1. Estimate the Project Requirements, Size and Complexity
2. Select Usage Base, Team Size and Flexibility Ratio of the Project
3. Develop a plan based on Feature list/ Functionalities /Sprint/Iteration
4. Test Design / Documentation & Coding
5. Analyse
6. Implement
7. Final System
8. Implement /Review/ Maintain

**Figure 5.1 Proposed Agile Methodology**

**Step1:- Estimate the Project Requirements, Size and Complexity and Risk**

In first step of proposed methodology, all requirements are taken from the customer and after taking customer requirements size, complexity and risk of the project is measured. For achieving the first step, start it from customer requirements.

- **Project Requirements**
  
  First of all in this methodology requirements may be taken from customers who may be static or volatile. If the requirements are static in nature means no change in the customer’s requirements then go to traditional methods otherwise go to agile. Volatile requirements are of various types like
Mutable requirements which change because of changes in the organization’s operating environment.

Emergent requirements which emerge as the customer’s understanding of the system.

Consequential requirements which result from the introduction of the computer system.

Compatibility requirements - requirements which depend on the particular systems or business processes within an organization.

In traditional methods changes in requirements are predictable & a plan can be developed and implemented for this but in agile, focus is given that project should start with minimum possible requirements and that allows the user to change the requirements when needed. The project will be started with minimum requirements which make the project more flexible.

• Project Size

After taking the customer requirements the next work is to measure the project size. To measure the project size various parameters can be used like how many people are needed in the development team, what will be the estimated cost of the project, what is the complexity of the project, what will be the development time frame of the project and on the basis of these parameters project size can be estimated. While classifying projects, project size is very important parameter. No way is considered best to describe the size of a software development project but generally the project which is completed in estimated time of the project is considered best.

• Complexity estimate of Project

Different matrixes are used to measure the complexity and cyclomatic complexity is one of them. It is calculated by counting the number of decision points found in the code and provides a single ordinal number that can be compared to the complexity of other programs. Cyclomatic complexity for a software module is based on the following equation:

\[ CC = E - N + p \]

Where
• CC = Cyclomatic Complexity
• E = the number of edges of the graph
• N = the number of nodes of the graph
• p = the number of connected components

Here following guidelines can be used to understand the equation:

• Start with 1 for a straight path through the routine.
• Add 1 for each of the following keywords or their equivalent: if, while, repeat, for, and, or.
• Add 1 for each case in a switch statement.

Here an example is given to understand how the code complexity is calculated.

Public void Process Pages ()
{

While (nextPage!=true)
{

If ((lineCount<=linesPerPage) && (Status!=Status.Cancelled) && (morePages==true))
{

//....

}
}

}

In the code above, start with 1 for the routine, add 1 for the while loop, add 1 for the if, and add 1 for each && for a total calculated complexity of 5.
After calculating the complexity, risk is evaluated. Some level of risks are available in projects and it is important to identify and manage those risks to save the project from failure. The following table is used to measure the risk assessment.

Table 5.1 Risk Evaluation

<table>
<thead>
<tr>
<th>Cyclomatic Complexity</th>
<th>Risk Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 10</td>
<td>A simple program, without very much risk</td>
</tr>
<tr>
<td>11 to 20</td>
<td>A more complex program, moderate risk</td>
</tr>
<tr>
<td>21 to 50</td>
<td>A complex, high risk program</td>
</tr>
<tr>
<td>&gt;50</td>
<td>An un-testable program means very high risk</td>
</tr>
</tbody>
</table>

2. **Usage Base / Team Size and Flexibility Ratio of the Project**:-

After completing the first step next step is to measure the usage base, team size and flexibility ratio of the project. To achieve second step, start it from the Usage base.

- **Usage base**
  
  Usage base means how the project is utilized for the benefit of the customer. Here it is checked that at which place the project is utilized and who are the potential users of the project, then software is developed according to the usage base of the customer.

- **Team Size**
  
  The team size is selected after considering the usage base of the customer. It is decided that what should be the team size so that project should be completed within time. Different methodologies have different team sizes. Teams should be small and variable in size.

- **Flexibility Ratio of the project**
  
  Project should be highly flexible to the requirements as it is the main feature of agile methodology. Flexibility is valuable (and thus desirable) when new information can be obtained, i.e. one expects to “learn” about the future. The value of this flexibility is
directly related to the value of this information. In decision analysis, this is normally
calculated by reconstructing the decision tree such that all uncertainties occur before
the decision nodes, so that appropriate action may be taken to optimize payoffs.

Flexibility Ratio of Project = (Change in Requirements/Total Requirements)*100

3. **Develop a Plan based on Feature list/Functionalities/Sprint/Iteration**
   During this step, a plan is get developed which decides that the final project will evolve
   with sprints/ iteration/feature driven/functionality based. After this, the actual
development starts.

4. This step involves four phases which repeats until the final project is ready. The details of
   four phases are as follows:
   
   I. First of all requirements are prioritized. Then highly prioritized
      requirements are firstly analyzed for their validity. The possibility of
      incorporating the requirements in the system to be developed is also studied.
   
   II. In design phase overall system architecture is defined, documented and then
      coding is done.
   
   III. In Implementation phase work is started for developing it.
   
   IV. In testing phase the prepared model is tested by user and takes suggestions
      as well as new requirements from the user and on the basis of these process
      repeats until the project is fully completed.

5. **Final System**
   In this phase project is ready and a final system is prepared.

6. **Implement /Review / Maintain**
   The system which is developed in the previous stage is ready for implementing. Project is
   handed over to the customer and takes the reviews of the customer. If still any changes
   are required from the customer then do changes in the product and what steps should be
   taken for maintaining it is also decided in this phase.
   
   So proposed methodology completes a project in the six steps with the faster and easier
development of the system.
5.3 Conclusion

The proposed agile methodology appears to cover the likely phases of the software development. A plan is developed based on Sprint/Iteration/Feature list according to the flexibility ratio of the project which also influence the final output that results faster and easier development of the software. The proposed methodology exhibits the following features like small team size, more adaptable to changes, having design phase, work week according to the project need, proper documentation and supporting code-refactoring with code ownership.

Therefore the proposed agile methodology addresses the issues of developing a new software system with the rapidly changing environments, thus it has great applicability in agile development environment.