**Software Devlopment Model**

**Agile SCRUM Methodology**

**Abstract**  
  
Successful software is one which provides quality product in given cost and time.  
Delivering quality software in time is a difficult task. Traditional software processes are heavy weight, giving importance to documentation and are rigid making them difficult to apply to different software projects. In recent years there has been growing interest in lightweight methodologies which provide a useful compromise between no process and too much process. Agile methods or lightweight methods are less documentation oriented and more code oriented stating that source code is the most important document. Agile method is a method of software development that aims at customer satisfaction through early and continuous delivery of useful software components. These methods are called agile because they are adaptable and not rigid and do not follow a specific plan. This paper discusses a few agile processes, the philosophy driving them and challenges faced while implementing them. 

**Introduction**  
  
In earlier days software development activity followed “code and fix” approach. This approach worked well for small systems but failed when systems grew larger and when there was need to add new features. To avoid this Engineering methodology came into picture. The aim of these methodologies was to make the software process more predictable and efficient by having a strong emphasis on planning activity. This approach works well for other engineering fields, like lot of planning is needed to build a bridge or a house. But the software market is ever changing and brings in greater choices into market. Users and managers must deal with issues like what to include and what to exclude in the software, which technologies to use, what will give the company a competitive edge?. These questions difficult to answer and trying to predict them in a rapidly changing market is even more difficult.   
  
As a reaction to these methodologies, lightweight methodologies like agile methodologies appeared on the scene. Agile methodologies attempt to compromise between no process and too much process.

**• Agile methods are adaptive rather than predictive.**  
  
Engineering methods try to plan out in great detail for long span of time, this approach works well till there are no changes in design. Agile methods welcome change and try to adapt and thrive on change.   
  
**• Agile methods are people oriented rather than process oriented.**  
  
Engineering methods define processes so that they work no matter the skill of the workers. It says that individuals are not as important as their roles. This approach is correct for a factory where workers are not the most intelligent and creative people. Agile methods state that no process will ever make up for the skill and intelligence of the development team. So role of process is to support development team in their work.  
  
**Agile Process Philosophy**  
  
Agile software development philosophy has its roots in the reality of today’s markets. Agile software processes attempt to deal with issues introduced by rapidly changing and unpredictable markets. The “Manifesto for Agile software development” [1] the basic ideas of the philosophy are introduced through four basic values:  
  
**• Individuals and interactions over processes and tools.  
• Working software over comprehensive documentation.  
• Customer collaboration over contract negotiations.  
• Responding to change over following a plan.**  
  
The items to the right have value, however, the items on the left define the agile philosophy. In this paper, we will focus on the left hand side items to explain the agile software development. We will now look at all the four aspects in detail.  
  
**Individuals and interactions**  
  
Using adaptive process requires a very effective team of developers. The team has to work well together to be more effective. Face to face meetings have special importance in agile processes. It is believed that people respond quicker and transfer ideas more rapidly when talking face to face than they can when reading or writing documentation.. Extreme programming introduces the concept of pair programming where two developers develop a module together to provide much better and quicker output than the same job done individually. The concept of synergy (i.e. the interaction of two or more agents or forces so that the combined effect is greater than the sum of their individual effects) takes hold because a few designers, sharing a common space, working together, can produce more code quicker than can the same individuals working alone.  
  
In traditional methodologies treat people as resources that are like replaceable part. As stated earlier they say that individuals are not as important as their roles. They fail to understand that each individual is dynamic and unpredictable. When we are programming a computer, we are controlling a predictable device. But when handling human beings this approach fails. Treating individuals are replaceable resources reduces their morale and they look for much better working environments.

