**Agile Development**

Agile methods have been commonly used in software development since the 1990’s. Traditional plan driven development strategies were inhibited by constant changing of requirements in fast moving environments. It is common in software development that requirements only become clear once the users gain experience after it has been delivered. This created the need for methods that are designed to produce software quickly and adapt to changing requirements.

Within agile methods there is no detailed specification prior to development and documentation is reduced by using informal communication. Thus, specification, design and implementation are combined into a single, interleaved process. The system is often developed in increments, where at each increment a deliverable product is provided to the user to be evaluated. New releases are provided every two or three weeks. Software developers incorporating agile methods also make use of tool support for example, automated testing and continuous integration. This means the output of one stage provides the basis for planning in the next stage.

Agile methods are suitable for small to medium sized development systems. In smaller teams the overheads of planning, designing and documentation in plan driven development dominate the process leaving little time and effort for development.

The agile manifesto states:

* The customer provides and gives priority to system requirements throughout development.
* Requirements are expected to change, and the system must change with requirements.
* A deliverable is provided at each increment and the customer specifies what is desired at each increment.
* Simplicity is essential in both software and process. Complexity should be removed wherever possible.
* Team members should be allowed to incorporate their own ways of working and their individual skills should be exploited.

**User Stories**

To address changing requirements, agile methods integrated requirement generation with development. The concept of user stories was created to represent how a system might be used by the client. The client and development team work together to create stories which incorporate the requirements of the client for the development team to integrate into the system in future increments. Thus, these stories provide the basis for planning iterations.

Once user stories have been created, they can be broken down into tasks and the effort and resources needed for them to be implemented can be estimated. The customer also provides priorities for each task, which assists to identify the most useful functionality in the eyes of the customer. As requirements change, which is often the case with agile development, user stories can be adjusted or removed altogether and new ones can be created.

User stories are much easier to comprehend than traditional requirement documentation in plan driven development.

Structure of a user story:

* As a **user** Iwant **feature** so that **benefit**.

**Scrum**

Scrum was developed to provide a structure for organising agile development and is the most widely used agile method. The Scrum process involves repetition of the sprint cycle where the time is fixed, which is usually two to four weeks. The product owner prioritises items for a backlog to define the aims of a particular sprint.

The Scrum sprint cycle:

1. The work to be done is reviewed and stories are selected from a product backlog which is a list of items to be worked on by the Scrum team. The product backlog may include items which are not directly related to development tasks for example, it may involve refactoring a part of the code or developing documentation.
2. During sprint planning the user stories are broken down to provide the sprint backlog and tasks are assigned to developers. Developers are responsible for selecting the high priority items to be completed in the upcoming sprint and provide an estimate for the time required to complete a given task.
3. During the sprint development is carried out and a shippable product is delivered to the client at the end. Uncompleted items are returned to the backlog as time is never extended, a sprint works to strict deadlines. Daily meetings, known as ‘stand ups’, are held by the team, to review progress, discuss any issues that team members are facing and re-plan where necessary. A Scrum board, which is a whiteboard, provides information on the product backlog, work done and staff. This is shared by the team, and anyone can edit it, it allows for all team members to be kept up to speed on the sprint’s progress.
4. A sprint review occurs at the end of the sprint, which aims to achieve two purposes. Firstly, a review of the technical outcome of the sprint including what has been achieved, and what has not and thus should be returned to the product backlog. Secondly, to improve the process, meaning a reflection on how the team has done and how things may have been done better. This is a review of teamwork and organisation aspects and is generally the goal of a sprint retrospective which may be regarded as a separate entity.

Benefits of Scrum:

* Development is broken down into manageable pieces.
* Unreliable requirements do not prevent progression.
* It is a transparent process which leads to better communication and morale.
* Products are delivered on time without unforeseen delays.
* It invokes a good level of trust between client and development, which in turn leads to a greater chance of success.

**References:**

Sommerville, I. (2016). *Software Engineering*. 10th ed. Hallbergmoos/Germany Pearson.