

## Lecture No. 3

### Requirement Engineering

#### 3.1 Requirement Engineering

We recall from our previous discussion that software development is not simply coding – it is a multi-activity process. The process of software construction encompasses and includes answers to the following questions:

- What is the problem to be solved?
- What are the characteristics of the entity that is used to solve the problem?
- How will the entity be realized?
- How will the entity be constructed?
- What approach will be used to uncover errors that were made in the design and construction of the entity?
- How will the entity be supported over the long term when users of the entity request corrections, adaptations, and enhancements?

These questions force us to look at the software development process from different angles and require different tools and techniques to be adopted at different stages and phases of the software development life cycle. Hence we can divide the whole process in 4 distinct phases namely vision, definition, development, and maintenance. Each one of these stages has a different focus of activity. During the vision phases, the focus is on why do we want to have this system; during definition phase the focus shifts from why to what needs to be built to fulfill the previously outlined vision; during development the definition is realized into design and implementation of the system; and finally during maintenance all the changes and enhancements to keep the system up and running and adapt to the new environment and needs are carried out.

Requirement engineering mainly deals with the definition phase of the system. Requirement engineering is the name of the process when the system services and constraints are established. It is the starting point of the development process with the focus of activity on what and not how.

#### Software Requirements Definitions

Before talking about the requirement process in general and discussing different tools and techniques used for developing a good set of requirements, let us first look at a few definitions of software requirements.

Jones defines software requirements as a statement of needs by a user that triggers the development of a program or system.

Alan Davis defines software requirements as a user need or necessary feature, function, or attribute of a system that can be sensed from a position external to that system.

According to Ian Sommerville, requirements are a specification of what should be implemented. They are descriptions of how the system should behave, or of a system property or attribute. They may be a constraint on the development process of the system.

IEEE defines software requirements as:

- 1 A condition or capability needed by user to solve a problem or achieve an objective.**
- 2 A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.**
- 3 A documented representation of a condition or capability as in 1 or 2.**

As can be seen, these definitions slightly differ from one another but essentially say the same thing: a software requirement is a document that describes all the services provided by the system along with the constraints under which it must operate.

## 3.2 Importance of Requirements

Many of the problems encountered in SW development are attributed to shortcoming in requirement gathering and documentation process. We cannot imagine start building a house without being fully satisfied after reviewing all the requirements and developing all kinds of maps and layouts but when it comes to software we really do not worry too much about paying attentions to this important phase. This problem has been studied in great detail and has been noted that 40-60% of all defects found in software projects can be traced back to poor requirements.

Fred Brooks in his classical book on software engineering and project management “The Mythical Man Month” emphasizes the importance of requirement engineering and writes:

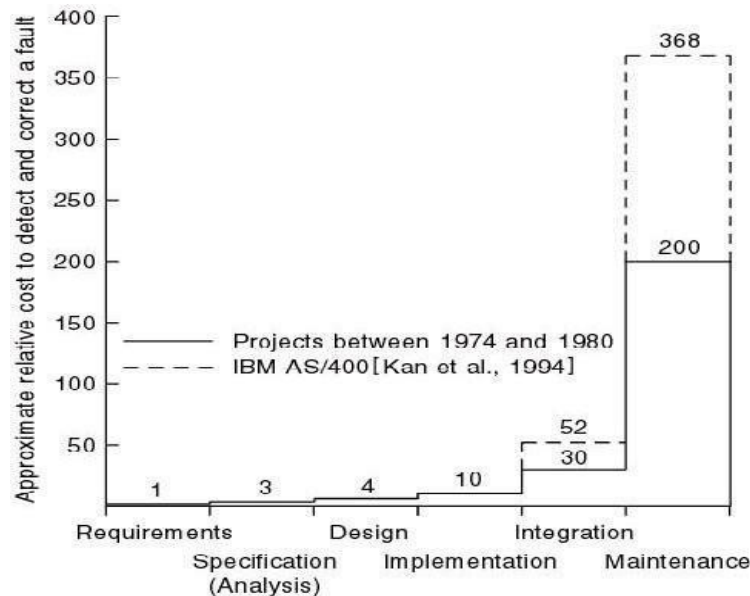
*“The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the system if done wrong. No other part is more difficult to rectify later.”*

Let us try to understand this with the help of an analogy of a house. If we are at an advanced stage of building a house, adding a new room or changing the dimensions of some of the rooms is going to be very difficult and costly. On the other hand if this need is identified when the maps are being drawn, one can fix it at the cost of redrawing the map only. In the case of a software development, we experience the exact same phenomenon - if a problem is identified and fixed at a later stage in the software development process, it will cost much

Dr. Syed Rafaqat Hussain Kazmi



This following graph shows the relative cost of fixing problem at the various stages of software development.



Boehm (1981) has reported that correcting an error after development costs 68 times more. Other studies suggest that it can be as high as 200 times. Since cost is directly related with the success or failure of projects, it is clear from all this discussion that having sound requirements is the most critical success factor for any project.

### 3.3 Role of Requirements

Software requirements document plays the central role in the entire software development process. To start with, it is needed in the project planning and feasibility phase. In this phase, a good understanding of the requirements is needed to determine the time and resources required to build the software. As a result of this analysis, the scope of the system may be reduced before embarking upon the software development.

Once these requirements have been finalized, the construction process starts. During this phase the software engineer starts designing and coding the software. Once again, the requirement document serves as the base reference document for these activities. It can be clearly seen that other activities such as user documentation and testing of the system would also need this document for their own deliverables.

On the other hand, the project manager would need this document to monitor and track the progress of the project and if needed, change the project scope by modifying this document through the change control process.

The following diagram depicts this central role of the software requirement document in the entire development process.

