

REQUIREMENTS MANAGEMENT AWARENESS

Objectives

The objective of the information on this page is to provide an awareness to customers about the benefits of developing clear requirement specifications, and the risks of having no or unclear requirements. It is not meant to be a "how to" guide.

Desired Outcome

Customers will obtain a better understanding about the positive effects that good requirements management practices will have on software system design, quality and integrity, and overall success of a software engineering project.

Benefits of Clear Requirement Specifications

The benefits of developing clear requirements before design and code, and practicing good requirements management include:

- A mutual understanding will be reached between customers and project teams about the requirements of the software
- Agreed-upon and approved software requirement specifications by the stakeholders
- A smooth flow into the system design activities
- Increased functionality, integrity during coding, and performance of the system when developed
- Improved testability, interfaces and maintainability
- Improved overall project management
- Improved communications between the customers and project managers
- Improved customer satisfaction and overall quality
- Cost savings

Importance of Maintaining Traceability

Each requirement should be traceable to a specific project objective described in the

Software Requirement Specifications. This traceability assures that the software product will satisfy all requirements and will not include inappropriate or extraneous functionality. It is important to know the source of all requirements so they can be verified to be necessary, accurate and complete. The traceability is also a key element in establishing auditability of the system during development, and maintaining it after the system is operational.

Risks of No or Unclear Requirement Specifications

There are many risks involved in not developing clear requirements and practicing good requirements management as evidenced in the following reports:

In the September 1994 issue of Scientific American, an article entitled "Software's Chronic Crisis" stated that "out of every 8 large systems development efforts started, 2 will be canceled. Seventy five percent (75%) of those delivered will have operating failures (do not function as intended or not used at all). The average software development project takes 50% longer than planned. Larger projects do worse". In addition, an IBM survey of 24 leading software companies revealed that "55% of projects cost more than expected, 68% overran their schedules, and 88% require substantial revisions".

These reports and field knowledge show that these types of failures can be attributed to the following risks:

- Starting projects with no or poor requirement specifications
- Changing requirements instream
- Adding requirements without conducting a risk analysis to determine impact on project plans
- Adding and changing requirements without reestimating the cost of the project
- Lack of requirements configuration management

Accountability

Identifying stakeholders of a software project in the Project Plan facilitates decision making and unified approval of the Requirement Specifications and other deliverables. The stakeholders should meet for a short time at logical checkpoints (e.g., end of a stage or phase) in the software development or maintenance project to review the progress and approve continuing the project. These assessments improve overall stakeholder awareness of activities and deliverables of a software project. The process promotes open communications between customers, project managers and their managers, and provides prudent focus on a project by all stakeholders. The end result is better accountability of

requirements.

Flexibility of Tailoring and Graded Approach

Realistically, requirements are not rigid and set. Therefore, a graded approach should be used to develop requirements based on the size and scope of the software project. However, the initial requirement specifications should address all functions the system is expected to perform so the risks stated previously can be avoided.

Summarily omitting requirement specifications because a software project is small is a risk. Small systems often grow through popularity, cross organizational lines and evolve to include multiple system interfaces. This often happens to systems that have no requirement specifications, and they grow without requirements being readdressed. These pitfalls often result in increased operational problems, lost code and nothing upon which to base user training.

It cannot be overstated that good requirement specifications are critical to the integrity of the software and providing the customer a product with the functionality requested.