



# Managing Successful Software Development Projects

**Mike Thibado**

**12/28/05**



## Table of Contents

<b>EXECUTIVE OVERVIEW</b> .....	<b>3</b>
<b>STATEMENT OF WORK DOCUMENT</b> .....	<b>4</b>
<b>REQUIREMENTS CHANGE PROCEDURE</b> .....	<b>5</b>
<b>PROJECT ACCEPTANCE PROCEDURE</b> .....	<b>6</b>
<b>PROJECT PLANNING AND SCHEDULING TOOLS</b> .....	<b>6</b>
<b>SOFTWARE REQUIREMENTS AND DESIGN DOCUMENTS</b> .....	<b>7</b>
DETAIL PRODUCT REQUIREMENTS .....	8
SYSTEM ARCHITECTURE SPECIFICATION .....	8
OBJECT DATA MODELS .....	8
<b>SOFTWARE DEVELOPMENT TOOLS</b> .....	<b>9</b>
<b>QUALITY ASSURANCE TOOLS</b> .....	<b>9</b>
<b>SUMMARY</b> .....	<b>10</b>

## Executive Overview

In order to keep pace with continually evolving hardware and software technologies, organizations must have a fluid process to identify, select, build, and implement systems that help improve their business efficiency. Without this process, software development projects can be very difficult to control. They very often result in products that do not meet the original functional and quality level requirements and are delivered late or over budget. Industry statistics have shown that as many as 80% of software projects failed significantly. These project failures are usually not due to lack of technical resources. But rather, result from a lack of applying proven fundamental management concepts.

Since the IT industry is very technical and closely involved with rapid changes in technology, project staffing emphasis is usually placed on technical skills at the expense of project management skills. Additionally, most IT managers building software systems have not been trained to manage large software development projects as they have come up through the technical ranks.

This paper discusses tools and techniques for successfully managing software projects. It emphasizes the need for a disciplined approach and use of proven management methods for defining, planning, monitoring, and delivering software products.

At a high level, the key areas of the project management approach to focus on are:

1. **Defining the project in detail.** Understand and clearly document the end users' needs. Define the client's expectations and identify exactly what will be delivered during the project.
2. **Getting the right people involved.** Identify the appropriate skill sets and experience levels required for the project team.
3. **Estimating the costs and time involved in the project for each phase of the development process.** Break the project into components or smaller pieces of work. Estimate each of those pieces of the project to increase the accuracy of the total project estimate.
4. **Establishing a formal change procedure.** Changes in requirements or assumptions are unavoidable and normal. Have a process and an expectation in place to accommodate the changes that will occur.

**Over 80% of software projects fail significantly - usually due to a lack of applying proven fundamental management concepts.**

5. **Establishing a formal acceptance procedure.** All parties must agree on when the project or components of the project will be considered complete. Have a process in place to acknowledge the delivery of a component and its acceptance as done.

The software development process follows a general set of phases that lead the project from its definition phase through testing and final delivery. There are several software methodologies and best practices available in the industry that may be used to help deliver successful projects. Each methodology or best practice has one thing in common, a set of tools and techniques that facilitate the development of software. The methodology and tool set selected should depend largely on the type of software project being undertaken and the skill level of the project team with the various methodologies and tools. Select the methodology and tool set that is the best fit for your project team.

Tools help facilitate the understanding of the project goals and communications surrounding the development effort. Each tool has a specific structure and use within a development framework. Below is a basic software development methodology with a list of tools to be used. The tools are listed here and defined in greater detail in the following sections.

- Statement of Work
- Requirements Change Procedure
- Project Acceptance Procedure
- Project Planning and Scheduling Tools
- Software Requirements and Design Documents
- Software Development Tools
- Quality Assurance Tools

## Statement of Work Document

The Statement of Work (SOW) may be the most important tool for the entire project. The SOW:

- Defines, in writing, the goals for the project and provides the basis for the project manager to anticipate and resolve problems that occur during the project.
- Contains the high-level definition and boundaries of the project.
- Documents the project team understands the user's environment and requirements.

This document is not a technical document, it should be written in simple business terms so that the customer and all project team members can understand and agree with its goals.

The SOW is critical to the success of the project. It demonstrates that the project team:

- Understands the customer's needs.
- Can describe an approach to meeting the software requirements.
- Is able to develop a reasonable estimate and schedule for the project.
- Has correctly identified the deliverables.

It communicates to everyone involved a mutual understanding of what the project involves and, therefore, what the project does not involve. If this step is not performed the customer's expectations will not be met and it is likely that the project will fail.

The size of the SOW will vary with the complexity of the project. However, it should provide enough definition to describe all of the deliverables for the project and enough detail to ensure it is clearly understood and accepted by the customer. It should also clearly state what is needed to gain approval when the project is completed.

## Requirements Change Procedure

In any project, change is inevitable. Changes occur for a variety of reasons including, modifications in business opportunities, technologies, or errors in communication. Changes need to be planned for and procedures must be in place to deal with them quickly and efficiently. Since change is inevitable and rework or additional work is usually added to the project plan, the overall project cost tends to go up. It is a good practice to add a 15% to 20% "Change Budget" to the baseline cost of the project. It can be specified in either dollars or hours of labor. If this Change Budget is planned for ahead of time and already agreed to by executive management, less time is spent later waiting for approval, which in turn can avoid a slow down in the project and cost overruns. The procedure for handling changes to budget should be included in the Statement of Work, prior to project initiation. The procedure should reflect the following concepts:

- All parties agree who will be responsible for approving changes and in what time-frame they will be approved.
- The person requesting the change fills out the Project Change Request Form and submits it to the project manager.
- Project team members are assigned to review the request and provide a cost analysis and project schedule impact estimate.

**To deal with changes quickly and efficiently add a 15% to 20% "Change Budget" to the baseline project cost.**

- The change request form and the project impact information are then forwarded for approval.

Project changes must be documented and approved by both the customer and the project manager. The changes are then applied against the initial project costs and timelines and those become the target project plans to be monitored at project review meetings.

## Project Acceptance Procedure

The overall goal of a project is to complete a specified task or set of tasks for a customer. To satisfy this goal, it is important that all parties concerned with a project acknowledge completion. An acceptance procedure allows all parties to agree that the task is completed and the project can formally be closed. The procedure for handling project acceptance should be included in the Statement of Work, prior to project initiation. The procedure should reflect the following concepts:

- Everyone on the team understands and agrees to the documented acceptance procedure.
- Agreed upon acceptance criteria.
- Person(s) responsible for acceptance must be identified.
- Projects should be broken into several acceptance components, not just one at the end of the project. For example, each deliverable in the project plan should have a formal sign off step with acceptance criteria well defined and measurable.
- A written Product/Task Acceptance Form should accompany each sign off. If a product or task is not approved it must have specific written comments on why it failed acceptance and what steps are required for approval.

## Project Planning and Scheduling Tools

The basis for good project planning and scheduling includes having a thorough understanding of the detailed product requirements, using good estimating tools and techniques, as well as understanding the skill sets of the team members. Without this foundation, a project will have little chance to be delivered on time and on budget.

The initial product requirements are gathered before the project starts, to help estimate overall project effort included in the Statement of Work. Several estimating tools and techniques are available and should be used when estimating the size of a project. It is important to note that premature project estimates can be very costly. It sets customer expectations, which may not be able to be met.

Tasks or components of the system should be broken down into work units that are no larger than 80 to 120 hours of effort. This helps achieve a more accurate project estimate and provides better project milestones for project management.

Project management tools are critical for setting up and tracking projects with several tasks, timelines, resources, and costs. Select one that is comfortable for the team to use and is available in the project management environment. Microsoft Project is one such tool that is easy to use and provides excellent project planning and tracking capabilities. Use it consistently from the start of the project to the finish. Keep it up to date and use it for the project status review meetings and for work assignments to team members. Make all team members aware of where their tasks are in the schedule and any dependencies that may impact them as a team. Use the project management tool to:

- Track resource loading,
- Print Gantt charts,
- Show task dependencies,
- Highlight critical path flows,
- Denote task completion or slippage, and
- Track project costs and monitor budgets.

A project folder should also be kept for all projects. It should contain all written documents, project plans, budgets, change proposals, and acceptance forms. Basically, everything documenting the project is kept in the project folder.

## Software Requirements and Design Documents

Once the high-level project requirements have been documented in the Statement of Work and the project funding is approved, the Detailed Requirements Gathering and Design phase begins. This is where the business analysts, system architects and designers begin their work.

The details of the system requirements, project scope, internal and external system interfaces, and complexity of the technology involved will dictate what documentation is required to communicate the system design goals. The project team will then need to decide which techniques and tools will help comprise the most effective approach based on the specific project – new build vs. add on. The goal is to find the right tools and not to over-document or under-document the system requirements and design. The following subsections discuss design tools and techniques that are common to software development approaches. Others may be used as needed. The Rational Unified Process (RUP) offers excellent tools and techniques for use in object oriented projects that are common today.

**To achieve a more accurate project estimate, break project tasks down into units of work no larger than 80 to 120 hours of effort.**

## Detail Product Requirements

The goals of detailed product requirements are to communicate to the client and end users that the business analysts clearly understand the full capabilities of the final product and to achieve sign off on the project requirements in order to move on to the next development stage. The product requirements will be used by the system architect, data modeler and application designer to begin the design phase.

The detail requirements for a software development project are gathered by the business analysts by interviewing Subject Matter Experts (SME). During the interview the analyst works to gain a specific understanding of the business products being produced, people that produce the products and business processes involved. Upon completion of the interview, notes are then organized into a high level scheme of related processes and individuals performing specific actions within those processes. The detailed functional requirements of the product capabilities are then defined and documented. The various actions that can be taken at a business level for each function are also defined. The final outcome of this process is a document such as a Use Case (RUP methodology).

**The goal of detailed product requirements is to communicate to the client that the full capabilities of the final product are clearly understood.**

## System Architecture Specification

The System Architecture Specification document is used to show a high-level abstraction of the entire system. It depicts a logical breakdown of the system into a set of related subsystems. This allows the system to be broken down in terms of logical processing or functional areas and to be shown as a picture for easier visualization and understanding. The picture should show interactions between sub-subsystems and include all interfaces to internal or external systems and data storage areas. The document should have descriptions of each of these sub-subsystems, interfaces, and data structures in following sections to discuss the role they play in the overall system.

The general flow of this document should have an Overview as the first section to orient the reader. The remaining sections should have a picture of the system with its sub-subsystems, interfaces, and data storage areas; descriptions of each of the sub-subsystems, interfaces, and data storage areas; and any assumptions or risks with the architecture. There may be several types of pictures used in this specification for showing static or dynamic views of the system.

## Object Data Models

These specifications describe the system in terms of business objects, object interactions, and users. In great detail, it defines the pieces of the system, how they interact, and how data is communicated and stored. There are a variety of tools that assist in defining and modeling the design of the system. The RUP methodology mentioned above is a good source for formal tools and techniques to help with developing this specification.

## Software Development Tools

Selecting the appropriate set of software development tools has several factors that must be taken into consideration. Some factors will limit the options and some will expand the options. One of the factors to be considered is the environment in which the new product must operate. For example, if the product is a browser-based application running in a Microsoft .Net environment, the tools set will be much different than if the project is extending an existing application that is written in PowerBuilder and runs on a mainframe.

Other factors to consider in selecting the appropriate development tools are interfaces to external systems, the development methodology being used, vendor support for a tool set, skills sets of the development team, availability of training for the tools and hardware requirements. These are just a few to be considered, there are many more factors dependant on the individual situation.

Given the number of considerations to be dealt with, it is very important to identify the most important ones early in the development cycle and standardize on the tool set. This will give the team time to get the development tools and environments purchased, installed and tested and complete any training required before the product development phase starts.

## Quality Assurance Tools

Quality assurance is more than just testing the final system. It is a discipline that is employed from the start of the project to its final delivery. One key step in the quality assurance process is having a well-defined approach to documenting and developing software products, as discussed above. The quality assurance process also includes defining project standards and goals that are measurable to help ensure quality is employed and tracked throughout the development process. Finally, the process defines guidelines as to when and where pieces of the system should be inspected or reviewed to gather data about the product's quality throughout the development life cycle.

A test strategy should be written that defines the type of testing to be performed on the application and who will conduct the testing. Test plans and test data can then be developed for all pieces of the system to be tested. Defect tracking mechanisms must be put in place for documenting and fixing problems that arise during the project testing.

Ensuring a high quality software development project requires the use of good quality assurance best practices and tools. Specific standards, goals, guidelines, and tools will vary with each project, but they must be defined and clearly communicated to the project team. These again should be defined and implemented before the system design phase begins or at least before it is completed.

**Quality assurance is a discipline that needs to be employed from the start of the project to the finish.**

## Summary

Successful high quality software projects depend on strong project management and the use of best practices, tools, and methodologies. Technical skills are very important to the success of a project, but without strong project management and proven tools for communicating clear project goals, requirements and changes projects have a very low success rate. Industry studies vary somewhat, but it has been documented that as many as 80% or higher of software projects have failed significantly to achieve their project objectives and/or exceeded their budget by 30%. The term “runaway projects” has been used to describe these projects. The five top reasons identified for the failures were:

- Project objectives not fully specified.
- Bad planning and estimating.
- Technology new to the organization.
- Inadequate or no project management methodology.
- Insufficient senior staff on the team.

Remember, for managing successful software development projects; use a strong project management discipline approach:

1. **Define the project in detail.**
2. **Get the right people involved.**
3. **Estimate the costs and time involved in the project for each phase of the development process.**
4. **Establish a formal change procedure.**
5. **Establish a formal acceptance procedure.**

Having a clear, documented planning methodology and strong project management leadership is the best practice to produce high quality successful software development projects.