

SOFTWARE PROJECT MANAGEMENT: SCENARIOS OF FAILURE AND POTENTIAL SOLUTIONS

Robert J. Boncella
Washburn University
bob.boncella@washburn.edu

Gerrald Reed
Washburn University
gerrald.reed@washburn.edu

Nan Sun
Washburn University
nan.sun@washburn.edu

Abstract

This paper provides the project manager (PM) with three (3) scenarios for examining project failure. For each scenario a decision tree is provided. By asking questions about the task, the PM can determine points of failure, and potential remedies. This is intended to be a learning tool and should affect how a PM prepares for his/her next project.

Keywords: software project failure, project failure diagnosis, project management methods

Fred Brooks, author of *The Mythical Man-Month*, was once asked how software projects fall behind schedule. His response was simple but profound: “One day at a time”. The same can be said about quality failure and cost failure. To avoid putting the entire project in jeopardy, time, quality, and cost problems need to be recognized and addressed promptly.

Inexperienced software project managers (PM) often find themselves confronted with delays and challenges that can lead to project failure. This paper intends to offer some relief to those managers, by illustrating questions that can lead to identification of failure points. The paper focuses on scenarios built around the five key elements of Project Management: requirements, resources, scheduling, budgets and quality.

Requirements are functional requirements explicitly stated by the users and non-functional requirements that are fundamental but not explicitly stated (e.g. the user interface will be assumed to be appropriate for the organization). It is key that the requirements be completely understood by the PM.

Resources are the assets that are needed to complete the requirements. The PM needs to be completely familiar with the capabilities of all the resources available.

Scheduling involves separating the total work in a project into separate activities and judging the time required to complete these activities. In general, it is the act of bringing the requirements and resources into balance and establishing a target delivery date.

Budgets are the resource costs that will be used to meet the delivery date.

Quality is conformance to explicitly stated functional and performance requirements as well as implicit requirements (e.g. ease of use, ease of maintenance, etc.). Lack of conformance is lack of quality.

In addition, the scenarios presented here maybe consider three expert systems. These expert systems are constructed from textbook cases and practitioner reports, and personal experiences. The majority of each scenario is based on the personal experiences of in of the authors Mr. Gerrald Reed. Prior to joining the faculty at WU Professor Reed was a software project manager for a worldwide retail firm whose corporate headquarters are located in Topeka, Kansas. He has approximately 35 years of experience at all levels of software project management. His expertise was crucial to the development of these expert systems. In essence he was the knowledge expert in our expert system development process.

The Figure 1 depicts the schedule as the element to balance the requirements and the resources.

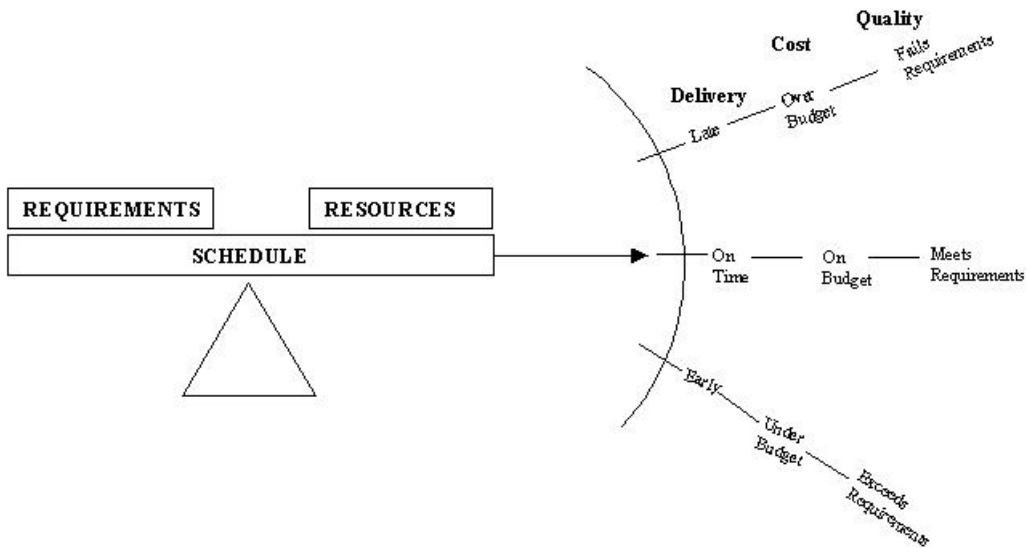


Figure 1 Ideal Balance of Elements of a Software Project

This balancing leads to meeting requirements, on time and on budget. The schedule is created during the planning phase and solidified at the end of the design phase. It is during the design phase that the ‘work breakdown structure’ can be matched against resources and the best schedule created.

The Figure 2 depicts the ‘usual’ imbalance that can occur in a project. Requirements can increase or change, or resources can be lost. This normally results in not meeting a deadline. The imbalance will cause the project deliverable date to shift upward and become ‘late’. Budgets and quality will be adversely affected by the delay. Early delivery, under budget, and exceeding requirements do not present problems. As a result no actions need to be taken.

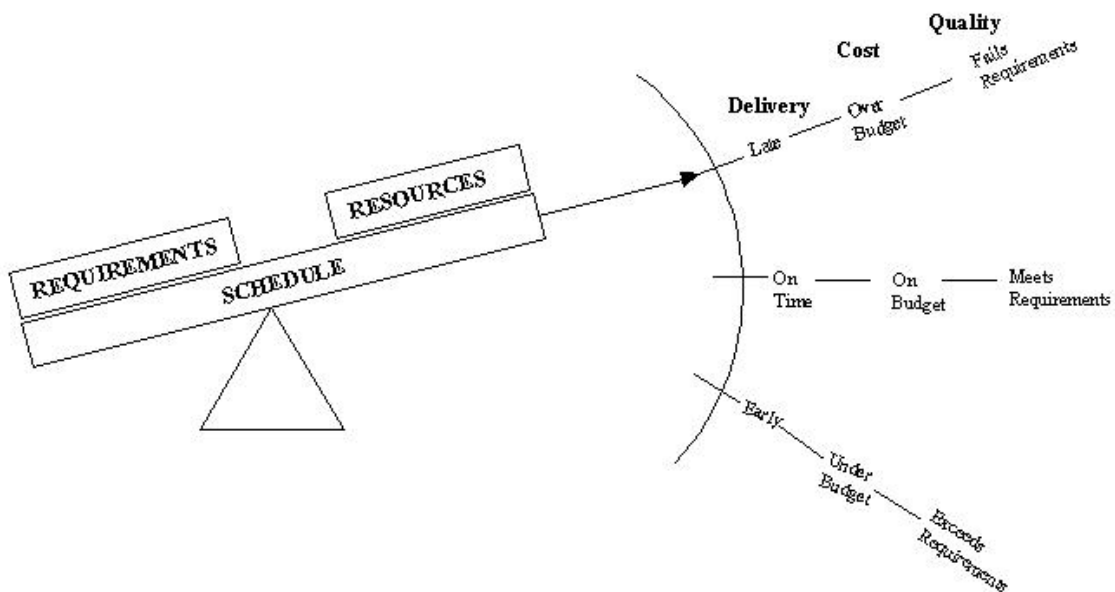


Figure 2 Effects of Out of Balance Project Elements

This paper provides the Project manager with three (3) scenarios for examining project failure. These scenarios can be acted out when the first task in the set of tasks that make up the project is in jeopardy. For each scenario a decision tree is provided. By asking questions about the task, the PM can determine points of failure, and potential remedies. This is intended to be a learning tool and should affect how a PM prepares for his/her next project. Each scenario is triggered by a reported event. Scenario 1 is triggered by the report of a missed deadline, Scenario 2 by the report of a deliverable not meeting requirements and Scenario 3 by a report of a task exceeding budget.

Each scenario is represented as a decision tree. The root node is the immediate question that is asked in response to the reported event. The yes or no response to this immediate gives rise to another question. This sequence of questions, based on a yes or no response, defines a path to a leaf node in the decision tree. This leaf node corresponds to a cause of the reported event. Included with the specification of the cause is a method that will prevent this leaf node from being reached in the future. These paths specified in these scenarios attempt to capture many of the causes of software project failure. And in addition specify prevention methods as well.

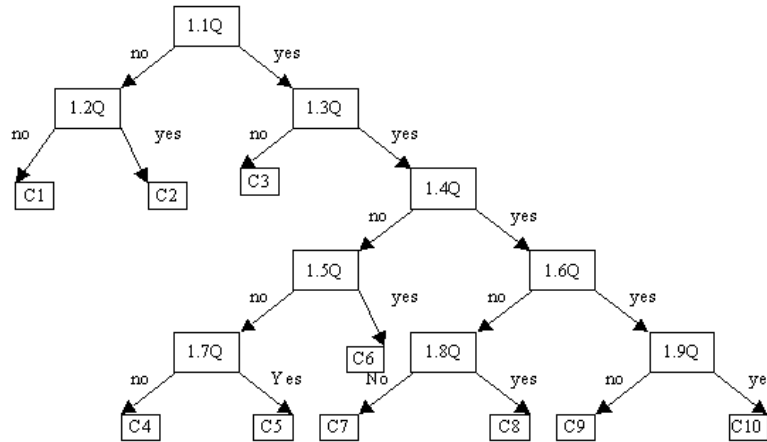


Figure 3 Decision Tree Scenario 1: A Deadline Was Missed

YES or NO Questions - Scenario 1

- | | |
|---|--|
| 1.1Q - Was deadline accepted? | 1.6Q - Was there enough time? |
| 1.2Q - Was this a forced deadline? | 1.7Q - Were priorities selected correctly? |
| 1.3Q - Were the requirements understood? | 1.8Q - Did the requirements change? |
| 1.4Q - Were there enough resources? | 1.9Q - Were the skills adequate? |
| 1.5Q - Was there a staff turn over on the team? | |

Causes and Prevention Methods - Scenario 1

- C1 - Failure to communicate the deadline**
 - P1 -** Make sure the project timeline is shared among all team members ahead of time. Send a reminder a few days before the task is due.
- C2 - Forced deadline**
 - P2 -** if the deadline cannot be changed, make sure that the developer agrees on the timeline and that adequate resources are assigned to finish the task
- C3 - Failure to understand the requirements**
 - P3 -** A requirements document should be prepared in compliance with industry or company standards. This document should be reviewed, understood, and signed off by all parties before work proceeds. It is the developer's responsibility to verify with users that his/her work reflects user needs.
- C4 - Failure to prioritize**
 - P4 -** If a person works on multiple tasks, the tasks need to be prioritized so that he/she knows which one takes first priority. Also, the task priority should be reflected in the schedule.
- C5 - Inadequate resources**
 - P5 -** Acquire and allocate adequate resources.
- C6 - Staff turnover**
 - P6 -** Develop a backup system for key personnel. Too much staff turnover can be a sign of employees unhappy with how the project is going. If that is the case, one needs to improve communication and team moral.
- C7 - Poor scheduling (not enough time allocated)**
 - P7 -** Involve the user and the developer in deciding how much time to allocate towards a given task.
- C8 - Unauthorized change of requirements**
 - P8 -** A change of requirements document should be prepared in compliance with industry or company standards. This document should be reviewed, critiqued, understood, and signed off by all parties.
- C9 - Skills do not match**
 - P9 -** Ensure the person assigned to the task has the right skills, both technical and interpersonal.
- C10 - Wrong person(s) on the team**
 - P10 -** Rebuild the team

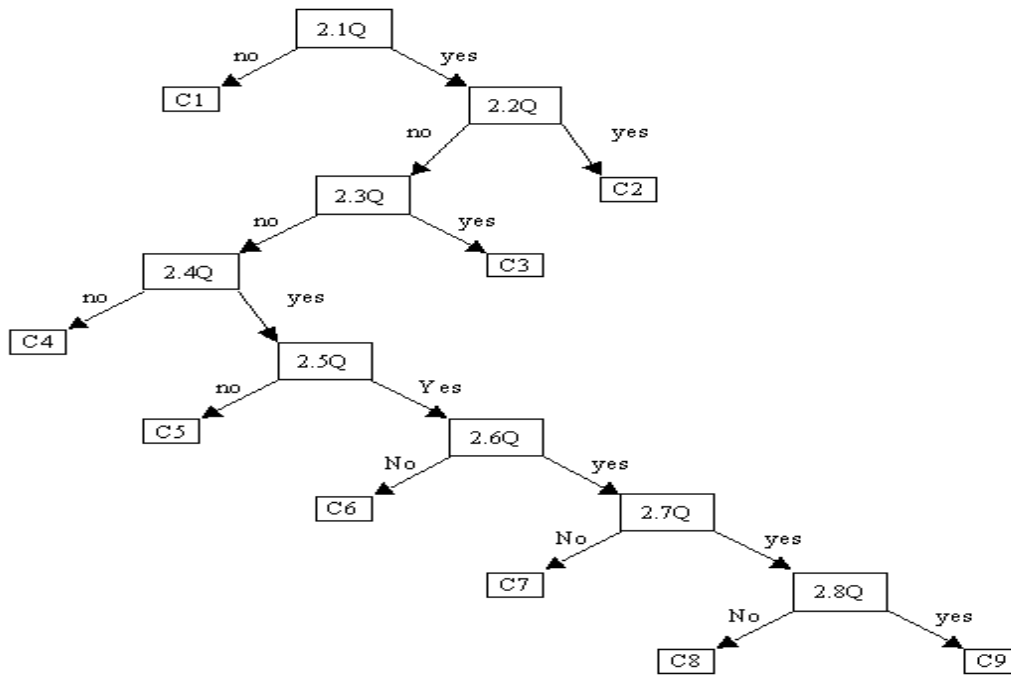


Figure 4 Decision Tree Scenario 2: A Deliverable Does Not Meet Requirements

YES or NO Questions - Scenario 2

2.1Q - Were the requirements accepted?

2.2Q - Were requirements changed?

2.3Q - Were the requirements misunderstood?

2.4Q - Was there enough time scheduled to work on the deliverable?

2.5Q - Were the skills adequate?

2.6Q - Was there a review or inspection process in place?

2.7Q - Did the review or inspection identify quality problems?

2.8Q - Was the test protocol adequate?

Causes and Prevention Methods - Scenario 2

C1 - Failure to agree on requirements

P1 - A requirements document should be prepared in compliance with industry or company standards. This document should be reviewed, understood, and signed off by all parties before work proceeds. If the developers do not accept user requirements, they should explain and negotiate with users until an agreement is reached. In no circumstance should the developers proceed with the project without agreeing on what needs to be done.

C2 - Unauthorized change of requirements

P2 - Establish/enforce change management policy. Any change on the requirements should be reviewed, evaluated, processed, and documented through established channel.

C3 - Failure to understand requirements

P3 - A requirements document should be prepared in compliance with industry or company standards. This document should be reviewed, understood, and signed off by all parties before work proceeds. It is the

developer's responsibility to verify with users that his/her work reflects user needs.

C4 - Poor scheduling

P4 - Involve the user and the developer in deciding how much time to allocate towards a given task. If the deadline cannot be changed, make sure that the developer agrees on the timeline and that adequate resources are assigned to finish the task

C5 - Skills do not match

P5 - Ensure the person assigned to the task has the right skills, both technical and interpersonal.

C6 - Lack of review or inspection process

P6 - Provide training in review or inspection process and make it part of every deliverable.

C7 - Review or inspection process not rigid enough.

P7 - Refine the current review/inspection process so quality problems can be identified at this stage.

C8 - Inadequate test protocol

P8 - Provide training in evaluating and selecting test protocols. Ensure that the right protocol is selected and all the tests are done based on the protocol subsequently.

C9 - Cannot meet requirements with current technology.

P9 - The project team needs to improve risk identification and management techniques. The project team should have identified this as a problem/risk when requirements were being presented and negotiated. Interim reports with warnings could have been helpful. Unless it is a research project full of unknowns, this situation will put the project team's credibility into jeopardy.

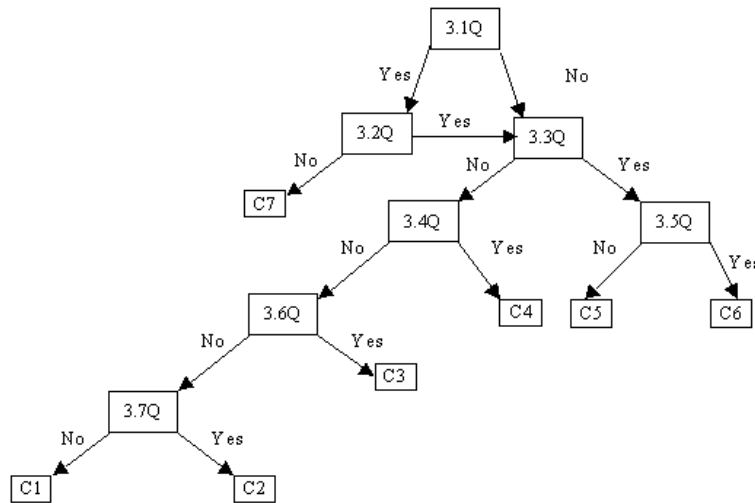


Figure 5 - Decision Scenario 3: A Task Exceeds Budget Requirement

YES or NO Questions - Scenario 3

3.1Q - Were there added requirements?

3.2Q - Were added requirements built into budget?

3.3Q - Was the project due date extended?

3.4Q - Were additional people allocated?

3.5Q - Did staff perform to expectations?

3.6Q - Was unexpected software and/or hardware purchased?

3.7Q - Was there unexpected increase in salaries, software, or hardware?

Causes and Prevention Methods - Scenario 3

- C1** - Poor estimation in the planning phase
P1 - Estimation should be based on historical data, standard methodologies, and employee input. Monitor budget and refine budget throughout life of project.
- C2** - Unexpected increase in salaries, software, or hardware beyond initial risk analysis
P2 - Improve risk management techniques by creating a contingency plan.
- C3** - Poor hardware and software estimation
P3 - The PM should improve architecture implementation estimation techniques by becoming familiar with current technology and cost structure.
- C4** - Poor human resource estimation
P4 - Improve estimation of resources required to finish a task. To do this the PM needs to understand each human resource's capabilities and limitations. Also, understand the team environment and its impact on human resources.

- C5** - HR issue
P5 - The PM needs to reevaluate staff and project management techniques. Identify team or individual issues that adversely affect performance. Once identified solve appropriately.
- C6** - Poor schedule estimation
P6 - Improve estimation time to finish task by using data from successful projects similar to the current project. If this historical data is unavailable use industry standards for projects similar to the current project, if they exist.
- C7** - Budget did not reflect workload
P7 - When requirements are changed, the initial budget needs to be revisited and modified to ensure that the budget reflects the changes made to requirements.

Summary

This analysis of software project failure is based on material found in current project management texts, practitioner articles, and discussions with senior software project managers about their experiences managing software projects.

The results of this analysis are intended for use as a pedagogical aid for the teaching of project management.

This paper aims to equip software project managers and/or students in Software Project Management class with useful tools to analyze and respond to early failures so that preventive measures can be taken to avoid more extensive failures. These tools can also be used by junior project managers and/or serve as a review session for senior project managers. In addition, those who have no formal training in project management but need to oversee that process can use this tool. For example members of a steering committee with project oversight would find this analysis useful.

Future Research

Local practitioners and a Software Project Management class will try out the scenarios provided in the paper in spring 2004. Feedback will be solicited from them and results will be shared in an extended paper. Future work would also include collecting real time data to pin down the causes for project failures and to provide potential solutions.

Bibliography

Boncella, R., Reed, G., Sun, N. “*A Course Sequence to Address Causes of Software Project Failure*” SAIS2003 Conference Proceedings, Southern Association of Information Systems.

Dennis, A., B. Wixom, D. Tegarden, *Systems Analysis and Design: An Object Oriented Approach with UML*, New York, NY: John Wiley & Sons, Inc., (2002).

Forum (1999), “Software Project Failure: Lessons Learned”, Communications of The ACM, November 1999/Vol. 42, No. 11.pp.21 – 24.

Gary, F.C., E. W. .Larson, *Project Management: The Managerial Process*, Boston, MA., McGraw-Hill (2000)

Hofmann, H., F. Lehner, (2001) “Requirements Engineering as a Success Factor in Software Projects”, IEEE Software, July/August 2001 pp. 58 – 66.

Humphrey, W.S., “Why Projects Fail”, Computerworld, May 20, 2002,
<http://www.computerworld.com/printthis/2002/0,4814,71209,00.html> (Current Nov. 15, 2003)

Pfleeger, S.L., *Software Engineering: Theory and Practice*, Upper Saddle, N.J., Prentice Hall Inc. (1998).

Pressman, R., *Software Engineering, a Practitioner’s Approach 5th edition*, Boston, MA., McGraw-Hill (2001).

Sommerville, I., *Software Engineering 5th edition*, Addison-Wesley (1995).