

Check-point on Success of Delivery Management Process and an Attempt to Look Forward and Beyond

Subrahmanyam VN Chinta, PMP

Cognizant Technology Solutions



Leveraging project management for excellence, growth and transformation



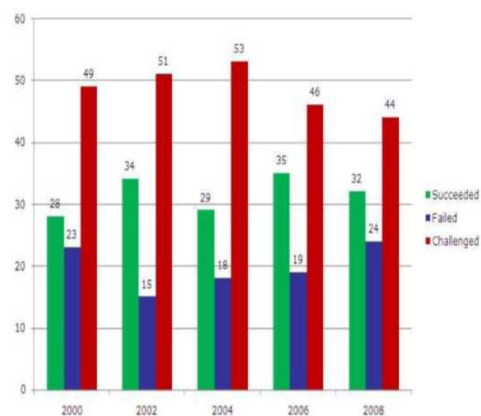
Contents

1.1	The Story So Far...	3
1.2	projects had high Client confidence levels and often paved new business.	4
1.3	Analysis and Root Cause Evaluation..	6
1.3.1	Organizational Factors	6
1.3.2	External Environmental Factors	7
1.4	Inaccurate Estimation and Schedule planning	7
1.5	Incorrect and optimistic status reporting	8
1.6	Unrealistic Schedule pressures	10
1.7	New and Changing requirements during development life-cycle	11
1.8	Inadequate Quality Control	11
1.9	Summary and Conclusion:	13
1.10	Definitions:	13
1.11	Author(s) Profile	14

As we are at the brink of closure of first decade in 21st century, it's the most appropriate stage to look at the way projects were executed during the decade. From a project management perspective, the number of projects that are challenged and failing continue to be high. The graph below provides a snapshot of how the projects have progressed during the period. It's important to diagnose the factors contributing to this scenario and identify solutions.

It's observed that despite significant contributions of technology and tools towards project delivery process, 40 - 50% projects continue to be "challenged". The percentage of projects failing to meet their stated objectives is on the rise. Hence, there is a compelling need to relook at the factors that are contributing to this state.

As overall business environment changes, it's but natural that there will be new set of constraints, factors and dependencies that will influence the project delivery process. The objective of this article is to go beyond the diagnosis of contributing factors and explore practices that can aid project managers to be equipped to handle so called "Challenged Opportunity".



Copyright © 2010 The Standish Group International, Inc.

1.1 The Story So Far...

Research on Successful projects ^[1] identified the reasons for their success - Stakeholder involvement; Clear requirements; Proper Planning and Realistic expectations.

It's observed that for any project which has adequate stakeholder analysis and tasks identified to keep the key stakeholders engaged during the project life cycle had higher probability of success.

SUCCESS CRITERIA	POINTS
1. Stakeholder (End User) Involvement	19
2. Stakeholders and Executive Management Support	16
3. Clear Statement of Requirements	15
4. Proper Planning	11
5. Realistic Expectations	10
6. Smaller Project Milestones	9
7. Competent Staff	8
8. Ownership	6
9. Clear Vision & Objectives	3
10. Hard-Working, Focused Staff	3

Engagement process, meeting stakeholder expectations and involving them at appropriate stages provided the project much needed attention, due importance and smooth approval process at required gateways.

For projects that have clearly stated requirements have seen low variance in scope change, schedule / cost variances, and least rework during development life cycle. Such projects have also seen low change requests.

For projects that had proper planning had smooth execution and delivery. These projects seldom went off track. Project Governance and controlling such projects was done with minimal hassles. Such

1.2 projects had high Client confidence levels and often paved new business.

Analyzing the challenged projects ^[1] for causes and factors that contributed to such an outcome - lack of user inputs, inadequate requirements, scope change, stakeholder support / engagement are major contributors to Challenge the projects.

For projects that lacked stakeholder (end user) inputs had challenges in completing the User / Formal Acceptance stage. Its observed that such projects had to go through an element of rework and changes.

Project Challenged Factors	Percentage
Lack of User Input	12.80%
Incomplete Requirements & Specifications	12.30%
Changing Requirements & Specifications	11.80%
Lack of Executive Support	7.50%
Technology Incompetence	7.00%
Lack of Resources	6.40%
Unrealistic Expectations	5.90%
Unclear Objectives	5.30%
Unrealistic Timelines	4.30%
New Technology	3.70%
Other	23.00%

For Projects that commenced with Incomplete Requirements and Specifications had numerous challenges during execution phase.

- ◆ Changes to Process Model; there by changing to project schedule and acquiring additional Risks
- ◆ Additional Uncertainties; project had challenges in completing critical activities like design, construction. Technical Solution was modeled with serious assumptions that had high impact.
- ◆ Delayed Sign-offs; project schedule was seriously challenged due to non-completion of prior phases. This caused additional risks, changes to schedule and controlling such projects was nightmare.

For projects that had Changing Requirements and Specifications had the following challenges:

- ◆ Changing scope- caused rework of schedule, incompleteness of phases
- ◆ Large volumes of Rework

Poor Communication, Insufficient resource planning, Unrealistic schedules and Poor Project requirements formed the top four reasons for most project failures^[3]. It was observed, that the above factors had a larger influence compared to the other factors. There were other significant observations, like lack of morale in the project team and their commitment towards the project was equally questionable.

Failure Reasons	Percentage
Poor Communication	28.00%
Insufficient resource planning	18.00%
Unrealistic schedules	13.20%
Poor Project requirements	9.80%
Lack of stakeholder buy in	6.70%
Undefined project success criteria	5.20%
Unrealistic budgets	4.80%
Insufficient or no risk planning	4.40%
Lack of control ' change process	4.30%

For projects that lacked communication have several complex challenges during planning and execution. Such projects had inadequate / no stakeholder identification, communication requirements are not identified. This led to inadequate planning and execution. Such projects constantly deviated from the objectives.

For projects that had insufficient resources planned, fared miserably during execution. Such projects had constant schedule / cost slippage. At times, the quality of deliverables was also questionable. For projects that had Unrealistic schedules had constant slippages in milestones. Resources had to put in unreasonable number of work hours to meet the deliverables. Team morale and productivity was low.



1.3 Analysis and Root Cause Evaluation..

Given the above project data, the projects were analyzed for key potential factors for projects to be "Challenged" or "Failed". Research lead to root causes for the respective outcome of the projects.

- ◆ Organizational Factors
- ◆ External Environmental Factors
- ◆ Inaccurate estimating and schedule planning.
- ◆ Incorrect and optimistic status reporting.
- ◆ Unrealistic schedule pressures.
- ◆ New and changing requirements during development life-cycle.
- ◆ Inadequate quality control.

1.3.1 Organizational Factors

What are the factors that an organization can influence on a project outcome? From analysis, the following were found to be the major contributors:


- a. Organizational Culture
- b. Processes / Information System / Process Models
- c. Stakeholder Assessment
- d. Technical Solution Envisaging and Framework
- e. Human Resourcing methodology and practices

The first of these - ORGANIZATIONAL CULTURE. The term 'Culture' means -"how we do things around here". This forms the genesis of any project initiation. Elements like - Identification of Project owner; process orientation; philosophy towards Governance; establishment of role and responsibilities; delegation of authority; establishment of accountability; and Creating an environment imbued with Ethics and Values. The above mentioned factors outlay the principles that govern projects during their life cycle.

The second Processes / Information Systems / Process Models. The emphasis here is on the systems, and processes followed for the execution of the project. Organizational Information system should aid in holding critical project repository; aid estimations; bring out the sensitive aspects of executed projects. It's also important to have complete understanding on the nature of process models available, implemented and their respective history.

The third Stakeholder Assessment - Every organization engages and maintains key stakeholders based on certain parameters and processes. These are intrinsic to the organization. The principles and practices defined will have corresponding impact on the performed project. Establishing and achieving stakeholder expectations has a huge impact on the organization vision, growth, and sustenance.

The fourth Technology Solution Envisaging and Framework - Organizations need to fundamentally be very strong in core technology that forms the basis for business. The mechanism used for envisaging solutions; understanding the nature of business;



ability to work on technology along with adequate knowledge on its strengths and limitations; ability to handle the technology challenges.

1.3.2 External Environmental Factors

What are the factors external to the organization that can influence the project outcome? From analysis, the following factors have serious impact on the project:

- a. Market Conditions / Business Decision Factors
- b. Competitive Business advantages
- c. Stakeholder risk tolerance

1.4 Inaccurate Estimation and Schedule planning


What are the factors triggering software cost estimating problems? From analysis and discussions of estimating issues with several hundred managers and executives between 1995 and 2006 [E], the following were found to be the major root causes of cost estimating problems:

- a. Formal estimates are demanded before requirements are fully defined.
- b. Historical data is seldom available for calibration of estimates.
- c. New requirements are added, but the original estimate cannot be changed.
- d. Conservative estimates may be overruled and replaced by aggressive estimates.

The first of these estimating issues - formal estimates are demanded before requirements are fully defined. Many of the estimation methodologies / frameworks are evolving and do not have a perfect solution for early estimates as of 2006, but there are some approaches that can reduce the risks to acceptable levels. However, there are certain cost estimation tools that enable managers to come up with early estimation in sizing a project that are yet to evolve clear / complete requirements. Using these approaches will also aid in estimating project team needs, infrastructure (including resources), schedules, costs, risk factors and quality parameters. In all such cases, handling risk analysis is the key element. During this approach, most projects are estimated with numerous assumptions and hence early estimates have confidence levels that are initially very low. As the project evolves, information becomes available and requirements are defined and agreed upon, the revised estimates will improve the accuracy and thus confidence levels as well.

As the above analysis clearly explains the challenge for coming up with cost estimates, which are intrinsically difficult and hence early estimates should mandatorily include contingencies for requirements changes and revised project plan needs.

The second estimating issue - historical data is seldom available for calibration of estimates - is strongly related to the first issue. Companies do not provide adequate emphasis on collecting project information from planning to closure phase. Hence, they lack correct / complete / accurate historical information on resources, schedules, costs, quality parameters, and metrics. Hence, using such data is always at risk when it comes to software cost estimation.



One advantage that function points bring to early estimation is that they are derived directly from the requirements and show the current status of requirement completeness. As new features are added / redefined / dropped, the application boundary is redefined so is the size of the application -resulting in change in the function point. As discussed earlier, this methodology will be of significant aid for projects that change process model during the execution phase. Since this is a scientific methodology, even if features are removed or shifted to a subsequent release, the function point metric can handle this situation exceptionally well.

The third estimating issue - new requirements are added but the original estimate cannot be changed - is that of new and changing requirements without the option to change the original estimate. Would like to bring in the analysis done above to look at this very sensitive and challenging situation. Given the project challenge in early estimation and most often scope not clear defined -estimate always carries range of assumptions and is more of budgetary in nature rather than one to define a project life cycle. If the project does not have options to redefine its trajectory as project elaborates then such project is bound to be challenged or fail to meet the project objectives. Organization culture and manner in which engagements are defined play critical role. Estimation models like Function points can be of better aid in defining project size and running projects based on size of the application. This, stakeholder management, in combination of having an opportunity to redefined certain parameters of the engagement during the course of the project life cycle is an ideal option to influence the project outcome.

The rate of requirements creep will be reduced if technologies such as joint application design (JAD), prototyping, and requirements inspections are utilized. Here too, commercial estimating tools can adjust their estimates in response to the technologies that are planned for the project.


The fourth and last of the major estimating issues - conservative estimates may be overruled and replaced by aggressive estimates - is the rejection of conservative or accurate cost estimates and development schedules by clients or top executives. The conservative estimates are replaced by more aggressive estimates that are based on business needs rather than on the capabilities of the team to deliver. There is no easy solution for such cases.

The best solution for preventing the arbitrary replacement of accurate estimates is evaluating historical data from similar projects. While estimates themselves might be challenged, it is much less likely that historical data will be overruled.

1.5 Incorrect and optimistic status reporting

One of the most common sources of friction between corporate executives and software managers is the social issue that software project status reports are not accurate or believable. What has long been troubling about software project status reporting is the fact that this key activity is severely underreported in software management literature. It is also under supported in terms of available tools and methods.

The situation of ambiguous and inadequate status reporting was common even in the days of the waterfall model of software development. Inaccurate reporting is even more common in the modern era where the spiral model and other alternatives such as agile methods and the object-oriented paradigm are supplanting traditional methods. The reason is that these non-linear software development methods do not



have the same precision in completing milestones as did the older linear software methodologies.

The root causes of inaccurate status reporting are:

- ✦ PMs are simply not trained to carry out this important activity.
- ✦ Correct / accurate data not communicated to stakeholders
- ✦ Inadequate Project Planning
- ✦ Inadequate Governance, control and responses to project events

The basic rule of software status reporting can be summarized in one phrase: No surprises! It's important to outlay metrics that needs to be planned and monitored to control the project.

Having a sustained governance plan is adequate for having a smooth project execution. Seven general kinds of information are reported in monthly status reports:

- a. Cost variances (quantitative).
- b. Schedule variances (quantitative).
- c. Size variances (quantitative).
- d. Defect removal variances (quantitative).
- e. Defect variances (quantitative).
- f. Milestone completions (quantitative and qualitative).
- g. Problems encountered (quantitative and qualitative).

Six of these seven reporting elements are largely quantitative, although there may also be explanations for why the variances occur and their significance.

The most common reason for schedule slippage, cost overrun, and outright cancellation of a major system is that they contain too many bugs or defects to operate successfully. Therefore, a vital element of status reporting is recording data on the actual number of bugs found compared to the anticipated number of bugs. Needless to say, this implies the existence of formal defect and quality estimation tools and methods.

The time and effort devoted to careful status reporting is one of the best software investments a company can make. With right discipline to capture correct and accurate data will facilitate healthy collection of historical data and increase the effectiveness and efficiency of future estimates.

In order to have a standardized process across various domains of the industry, a number of organizations and development approaches have included improved status reporting as a basic skill for PMs. Some of these include the Project Management Institute, the Software Engineering Institute's (SEI) Capability Maturity Model® (CMM®), the reports associated with the Six Sigma quality methodology, and the kinds of data reported when utilizing International Organization for Standardization (ISO) Standards.

1.6 Unrealistic Schedule pressures

Unrealistic schedule pressure by executives or clients is a common software risk factor. There are four root causes for unrealistic schedule pressure:

- a. Large software projects usually have long schedules of more than 36 months.
- b. PMs are not able to successfully defend conservative estimates.
- c. Historical data from similar projects is not available.
- d. External factors and business deadline affects the schedule.

Figure 1 shows U.S. industry experiences derived from several thousand software projects. The upper curve shows the average delivery time in calendar months, while the lower curve shows the planned or desired delivery time. The larger the project, the greater the gap between the actual delivery date and the planned delivery date of the application.

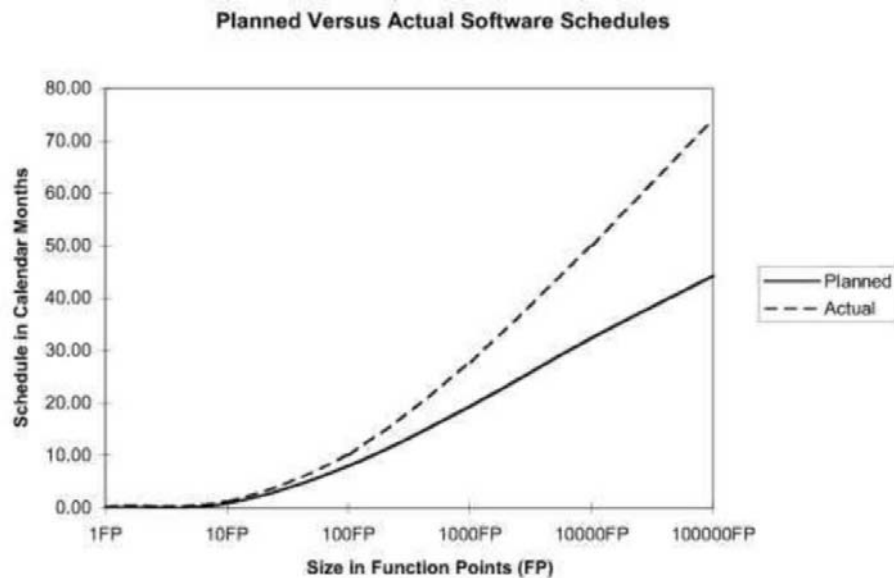


Figure 1: Planned Versus Actual Schedules for Software Projects

The most difficult problem to solve is when some kind of external business deadline affects the project schedule. Unfortunately, these business deadlines are usually outside the control of either PMs or technical personnel. Examples of external business deadlines include contractual obligations, the starting dates of new laws that require software support, or some kind of technical situation such as those associated with the Y2K problem.

Such external fixed dates cannot be changed, or at least not changed by project personnel. Therefore a combination of cutting back on functions, plus staff overtime, remains the most common method for dealing with fixed and unchanging delivery dates. If the mandated schedule is quite impossible to achieve, then a more drastic option would be project cancellation.



1.7 New and Changing requirements during development life-cycle

The root causes of requirements changes are dynamic businesses. Real-world requirements for software must change in response to new business needs. By counting function points from the original requirements and then counting again at the time of delivery, it has been found that the average rate of requirements growth is about 2 percent per calendar month from the nominal completion of the requirements phase through the design and coding phases.

The total accumulated volume of new or changing requirements can top 50 percent of the initial requirements when function point totals at the requirements phase are compared to function point totals at deployment. The state-of-the-art requirements change control includes the following:

- a. A joint client/development change control board.
- b. Use of JAD to minimize downstream changes.
- c. Use of formal prototypes to minimize downstream changes.
- d. Formal review of all change requests.
- e. Revised cost and schedule estimates for all changes less than 50 function points.
- f. Prioritization of change requests in terms of business impact.
- g. Formal assignment of change requests to specific releases.
- h. Use of automated change control tools with cross-reference capabilities.

One of the observed byproducts of the usage of formal JAD sessions is a reduction in downstream requirements changes. Rather than having unplanned requirements surface at a rate of 1 percent to 3 percent every month, studies of JAD by IBM and other companies have indicated that unplanned requirements changes often drop below 1 percent per month due to the effectiveness of the JAD technique.


Prototypes are also helpful in reducing the rates of downstream requirements changes. Normally, key screens, inputs, and outputs are prototyped so users have some hands-on experience with an example of the completed application.

However, changes will always occur for large systems. It is not possible to freeze the requirements of any real-world application. Therefore, leading companies are ready and able to deal with changes and do not let them become impediments to progress; some form of iterative development is a logical necessity.

1.8 Inadequate Quality Control

Effective software quality control is the most important single factor that separates successful projects from delays and disasters. The reason for this success is that finding and fixing bugs is the most expensive cost element for large systems, and it takes more time than any other activity.

The root cause for poor quality control is lack of solid empirical data on the cost effectiveness of a good quality control program. More than 50 years of empirical studies have proven that projects with effective quality control cost less and have shorter schedules than similar projects with poor quality control. However, a distressing number of PMs are not aware of the economics of quality control.



Successful quality control involves defect prevention, defect removal, and defect measurement activities. The phrase defect prevention includes all activities that minimize the probability of creating an error or defect in the first place. Examples of defect prevention activities include the use of the Six Sigma approach, the use of JAD for gathering requirements, the use of formal design methods, the use of structured coding techniques, and the use of libraries of proven reusable material.

The phrase defect removal includes all activities that can find errors or defects in any kind of deliverable. Examples of defect removal activities include requirements inspections, design inspections, document inspections, code inspections, and many kinds of testing.

The phrase defect measurement includes measures of defects found during development and also defects reported by customers after release. These two key measures allow leading companies to calculate their defect removal efficiency rates, or the percentages of defects found prior to release of software applications. Supplemental measures such as severity levels, code complexity, and defect repair rates are also useful and important. Statistical analysis of defect origins and root-cause analysis are beneficial, along with the key measurements of cost and defect repairs.

Some activities benefit both defect prevention and defect removal simultaneously. For example, participation in design and code inspection is very effective in terms of defect removal and also benefits defect prevention. Defect prevention is aided because inspection participants learn to avoid the kinds of errors that inspections detect.

Successful quality control activities include defect prevention, defect removal, and defect measurements. The combination of defect prevention and defect removal activities leads to some very significant differences in the overall numbers of software defects between successful and unsuccessful projects.

One of the reasons why successful projects have such a high defect removal efficiency compared to unsuccessful projects is the use of design and code inspections. Formal design and code inspections average about 65 percent efficient in finding defects. They also improve testing efficiency by providing better source material for constructing test cases.

Unsuccessful projects typically omit design and code inspections and depend purely on testing. The omission of up-front inspections causes three serious problems:

1. The large number of defects still present when testing begins slows the project to a standstill.
2. The bad fixes injection rate for projects without inspections is alarmingly high.
3. The overall defect removal efficiency associated with only testing is not sufficient to achieve defect removal rates higher than about 80 percent.

Fortunately, the SEI, ISO quality standards, and the Six Sigma approach have benefited quality control activities throughout the past 20 years. As a result, an increasing number of large projects have been successful compared to similar projects done in the 1980s.

However, for very large projects above 10,000 function points in size, missed delivery dates, cost overruns, and outright terminations remain distressingly high even in 2006. The industry is improving, but much more improvement is needed.



1.9 Summary and Conclusion:

Most software projects are executed in a challenged and demanding environment. Careful analyses of the challenged and failed projects suggest that fundamentals of Project Management principles are commonly compromised. Given the business needs and overall environment constraints from within and outside the organization, all upcoming projects will have to be executed in a constraint parameters and assumable large set of stakeholder expectations. Looking positive side, this provides an opportunity to implement projects in an effective and efficient manner.

One of the strong ways to achieve is by combining project management principles with tools that will aid the objective. It's aptly appropriate to recall Darwin's Theory of Survival of the fittest.

1.10 Definitions:

1. Successful Project: one that is completed on-time and on-budget, with all features and functions as initially specified.
2. Challenged Project: one that has deviated from their stated objectives by the time of completion. The software project is completed and operational but over-budget, over the time estimate, and offers fewer features and functions than originally specified.
For example, costs were high, huge schedule variance, not meeting the scope or enhanced scope etc.
3. Failed Project: one that could not meet the stated objectives by the time of completion. In some cases, the software project is canceled at some point during the development cycle. References:
 - a. PMI Standards - Project Management Body of Knowledge 4th Edition.
 - b. PMI Standards - Program Management Standard - 2nd Edition.
 - c. Why do software projects fail? Research Article from Outsource 2 India.
 - d. Research Articles from The Standish Group
 - e. Social and Technical Reasons for Software Project Failure - Capers Jones, Software Productivity Research, LLC
 - f. Glass, R.L. Software Runaways: Lessons Learned from Massive Software Project Failures . Prentice Hall, 1998
 - g. Jones, Capers. Patterns of Software System Failure and Success . Boston, MA: International Thomson Computer Press, 1995.
 - h. Garmus, D. and D. Herron. Function Point Analysis - Measurement Practices for Successful Software Projects. Boston, MA: Addison-Wesley Professional, 2001.
 - i. IEEE Software CS Digital Library
 - j. Jones, Capers. Applied Software Measurement. 2nd Edition. New York, NY: McGraw Hill, 1996. k. Harold Kerzner - Project Management: A Systems Approach to Planning, Scheduling, and Controlling

1.11 Author(s) Profile



Chinta presently at Cognizant Hyderabad as Delivery Manager is a connoisseur of Project Management. Delivering highly challenged projects has been his forte. As a PMI Volunteer worked of following standards - Practice Standard for Project Management Body of Knowledge -4th Edition (PMI PMBOK4th Edition); Scheduling 1st Edition; Risk Management 1st Edition. Working with various teams at PMI Chapters and outside mentoring folks on Project management has been another passion.

E-mail: Chinta vns@yahoo.com;
SubrahmanyamVenkata.Chinta@Cognizant.com