Effective Software Management– Where Do We Falter?

SYED JAWAD HUSSAIN[1]
KHALID RASHID[1]
H. FAROOQ AHMAD[2]
SYED FAWAD HUSSAIN[3]
Department of Computer Science, Faculty of Applied Science
International Islamic University
Sector H-10, Islamabad
PAKISTAN[1]
NUST Institute of Information Technology
National University of Science and Technology
Rawalpindi
PAKISTAN[2]
Laboratoire TIMC-IMAG
Institute Nationale Polytechnique
Grenoble
FRANCE[3]

Abstract

Construction industry has been performing better than the software industry in delivering project on time and within budget. A comparative study is presented that shows the difference in development methodologies in the two industries. A new software development methodology is proposed that borrows some concepts from the design-bid-build strategy of the construction industry and merges it with the iterative nature of software intensive development.

1. Introduction

Software industry suffers from an alarming rate of project failure. As low as 12 percent of the total projects get completed on time, within budget and with the required functionalities [3]. Several of the projects are destined for failure right from the day they are started. The question is *where do we falter?* After 40 years of software development, we still haven't reached the level of maturity and confidence [7]. The high rate of failure and the causes of failure have been identified on numerous occasions [8], [9], [13], [14]. Some of the causes pointed out in the literature are (i) Continuous changes in scope, (ii) Incomplete, and ambiguous requirements, (iii) Poor management, (iv) Unrealistic expectations, (v) and wrong software development process model. The software development process models have been blamed on several occasions. Process models like the prototyping paradigm, incremental model, spiral model, win win spiral model, unified process later upgraded to the unified process lifecycle, etc have all been proposed and tried but still we lack the much desired maturity. A simple website expected to be completed in 11 weeks took 8 months and twice the cost to complete. In medium and large projects, the problem is even worse.

Perhaps there is some lesson for us in the more mature, more successful industries. Perhaps we should explore *how they do it*, to achieve a much better success ratio. As Poppendieck et al quote in [7] that when a construction project

is announced, people have confidence that after five years they would have a spectacular art center but the authors fear that same level of confidence is not evident when a company announces its intentions of overhauling its software system.

2. Literature Survey

Several papers on software project management have quoted examples from the construction industry in an attempt to relate the success ratio and the lessons that can be learnt from it. [4], [6], [7]. This paper attempts to compare the two disciplines and pin points certain strategies that can be adopted to the software industry from the construction industry. A new methodology is then proposed based on the Design-Bid-Build methodology presented in the section below.

2.1. Methodology of the Construction Industry

The Construction industry has also tried several management methodologies and have been, on the average, more successful than the Software industry in delivering successful projects [4],[7]. One of the tried and tested methodology of getting projects developed is the Design-Bid-Build strategy where by one company is hired to design the project then bidding takes place to select a contractor to build the project [1]. This strategy divides the project into two and allows the relevant company specializing in the work to take on the task. Though new methodologies have come up in the construction industry, several companies still continue to follow the beaten path.

2.2. Methodology of the Software Industry

The Software Industry normally takes up the complete software development project from requirements gathering to testing to deployment as one project. The development firm starts off with requirements gathering and analysis, then the same firm and sometimes the same set of people make the design and then proceed to development, testing and deployment. Though this approach may propose a lot of advantages, we still need to rethink whether the construction industry's tried and tested methodology of splitting the task into two is better. One firm, the architect, specializes in making the design and architecture of buildings. The owner contacts the architect firm to develop the design [1]. The architect firm specializes in the job, they gather the requirements and develop an architecture for the building [7]. This gives the architect firm the liberty to focus on the actual task of gathering the requirements and proposing the best solution without having to bother about the so called umbrella activities like scope management and such others. The software development firms on the other hand take up the issue of scope management right from day one. The Project Management Body of Knowledge [15] describes scope management as one of the keys to project success least realizing that over emphasizing on scope at an early stage only defer the problem and does not solve it. Software projects still continue to fail. Several reports and studies quoted in [12] point to the high rate of failure in the software industry and some of the main reasons for such a high failure rate. Krasna et. al in [11] attribute poor planning as another reason for project failure.

2.3. Comparison of The Construction and Software Industries

Though there are several dissimilarities between the two industries, the construction industry may still hold certain basic management solutions. One thing that is evident is that construction contractors constructing the buildings are clearer about the details of what is to be developed. Definitely, this accounts, to some extent, to the tangible nature of the product input and output but still decoupling the analysis and design phase from the rest of the project is a basic rule and would work fine with the more intangible software industry. Why do software project managers have to estimate complete project development cost before requirement analysis when most of the details are not clear. If construction

engineers were to work on the methodologies laid down in the software industry, they would surely not be able to maintain their rate of project success.

Some of the reasons of failure quoted in [10] are similar to the reasons of failure of IT projects mentioned in the CHAOS report of the Standish Group [3].

3. Proposed Methodology

The complete project should be divided into two parts, the first dealing with analysis and high level design. After acceptance of the Software Requirement Specification document and an architectural design of the system, the project should be considered complete. The actual development of the project should be treated as a different project. A new cost and schedule estimate should now me made for the actual development and deployment of the project. Definitely, by now, with all the details of the requirements, the company bidding for the project is in a better position to make more realistic estimates. This second part of the project could be a separate company but ideally it should be the same company that completed the first part. This strategy may sound familiar, the Staged Contract or Milestone Based Development methodologies [5] have been quite successful in the 90's.These methods also suggested the division of a project into multiple tasks. Certain Pakistan Government projects do treat the analysis part as a separate project, but their terms and conditions do not permit the organization performing analysis to bid for the development project. This takes away all sense of responsibility from the analysis company involved. It also reduces the charm of the project in terms of future rewards in return of the good work that they may have done in the analysis project. Ideally the project should be open to all organizations including the analyst company; in fact, this company should be given extra weightage as it has a better understanding of the system. This is to reap the benefit of continuity and the sense of responsibility that would lie on the company performing analysis. By treating the analysis and high level design as a separate project from the development project, the company performing analysis would be in a better position to invest more on its analysis team.

Thus the project starts on a linear sequential approach, analysis and high level design are completed in sequence in the first part of the project. Next, the iterative approach is to be adopted for the detailed design, development, testing, and deployment of the project. The proposed solution emphasizes on achieving the benefits of both the Iterative process model and conventional Waterfall software paradigm. The Waterfall model divides the development of software into distinct non-overlapping phases. Each phase is completed before the next goes on floor, thus leaving no option of returning to an earlier phase. Theoretically it seams perfect and targeted towards achieving ultimate quality standards but practice has proved otherwise. Mostly the customer/user is not able specify all the requirements at such an early stage, similarly rectification of any error/mistake in design phase becomes very costly at later stages.

The Iterative method enables faster development [2], but it also carries certain limitations of its own. The proposed process model combines the two. Start of with a proper analysis and requirement gathering phase.

3.1. Analysis

Perform a thorough analysis and get the Software Requirement Specification (SRS) document signed off by all the stakeholders. Maintain a certain level of abstraction and do not go into such minute details that are not feasible to specify at this phase.
The analysis phase should be carried out in the normal linear sequential model style. It is recommend to produce a prototype for the sake of requirements gathering. A storyboard document containing all the screen shots along with a high level description should be attached with the SRS document while getting it signed from customer.

3.2. High Level Design

After analysis, construct a high level design specifying the architecture for the application, development language and tools, coding standards and style, etc. All major design decisions should be done at this stage. This gets you ready for rapid iterations resulting in incremental releases. All major decisions have been made, user requirements and ultimate goal is very clear and high level architecture is ready.

This sets an ideal stage for carrying out the iterations and build on the existing system. Each iteration should ideally add a substantial chunk of functionality and should consist of the following phases.

3.3. Detailed Design

Detailed Design is carried out for each release. The detailed design gets input from SRS and High level design and from the customer's feedback on previous releases. New functionality and changes on previous release are planned and designed at this stage.

3.4. Coding

Here we translate the design into executable application. Coding should be done on standards so as to make it maintainable and easy to change. All changes requested by customer on previous release should be incorporate during this phase.

3.5. Testing

Unit testing is carried out to verify that what ever has been produced is of quality and meets the user expectations. A short integration testing is also carried out to ensure that new release has not caused errors in other parts of the system. All this should be carried out to verify the major functionality of the system and we should leave the minute details for the customer to test and verify.

3.6. Release and Feedback

System should be delivered to customer through a series of releases. Each release should provide a substantial chunk of functionality useful to the user and it should span between two weeks to four weeks. Prepare a separate release document for each release. Share each release with customer and ship it along with a release note document. The

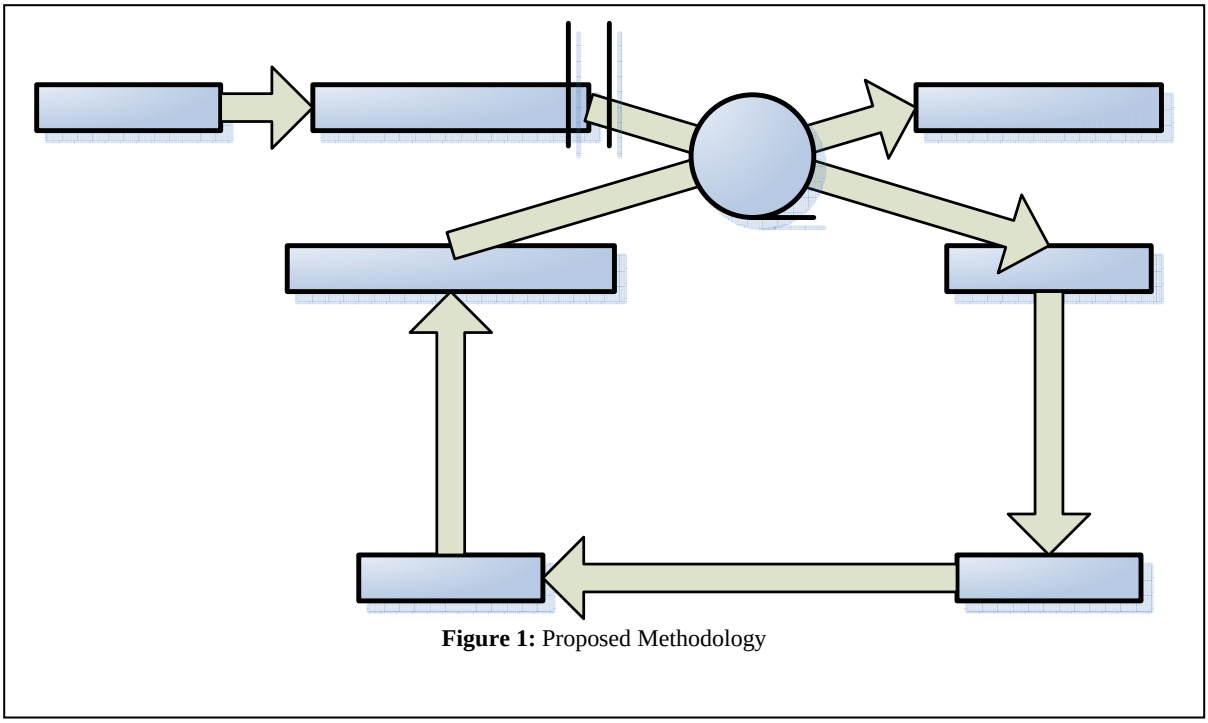

Figure 1: Proposed Methodology

release note document should contain reference to the SRS document and detailed description of functionality covered in the particular release.

Release Plan should be developed in such a way that customer reviews the release and gives feedback before the next release is shipped. Conduct a short feedback meeting with the customer after every release. In this meeting, try to get verification from customer that the release covers all the requirements (implicit, explicit and exiting requirements) and that it functions properly and is to the satisfaction of the customer.

3.7. Project Closure

The closure phase is carried out after all the releases have been delivered. It's a short phase parallel to the Transition Phase in the Unified Process Lifecycle. Conduct a setup review based on the release notes to verify to the client that all the requirements have been covered. If the feedback received after every release has been incorporated and got verified from customer, the closure phase is supposed to be short and smooth.

4. Conclusion

The Design-Bid-Build Strategy of the construction industry divides a project into two allowing the expert to handle the relevant task. The architect designs the building but does not indulge in the actual construction. The Contractor, responsible for construction, does not lay hand on the architecture designing.

By separating the analysis and high level design from the rest of the project, we propose to adopt the design-bid-build strategy like approach to software development. This would improve the correctness of our planning and success ratio of software projects.

5. Future Work

The methodology is being implemented on practical projects to study the results. On one of the projects, the analysis phase output has been comparatively better because of the free hand analysts had to do their job. A complete result shall be shared after completion of these projects.

6. References

[1] Adekunle S. Oyegoke, UK and US Construction Management Contracting Procedures and Practices: A comparative Study, Engineering Construction and Architectural Management, pp 403 – 417, 2001

[2] Antony Powell, Modeling Time-Constrained Software Development, 5th International Workshop on Process Simulation and Modeling, 2004

[3] CHAOS Report, The Standish Group International, 1995

[4] Chris Sauer, L. Liu, Kim Johnston, Enterprise Level Project Management Capabilities: A Comparison of the Construction and IT Services Industries, 2000

[5] Christopher M. Lot, Breathing New Life into the Waterfall Model, IEEE Software, September 1997

[6] David Alev, The Scope went through the roof, http://consultingacademy.com, 2005

[7] M. Poppendieck, T. Poppendieck, A Rational Design Process – Its Time to Stop Faking It, 2001

[8] J. Johnson, CHAOS: The Dollar Drain of IT Project Failures, Application Development Trends, pp 41-47, 1995

[9] Joichi Abe, Ken Sakamura, Hideo Aiso, An Analysis of Software Project Failure, Proceedings of the 4th Software Engineering, 1979

[10] Low Sui pheng, Quek Tai Chuan, Environmental Factors and Work Performance of Project Managers in the Construction Industry, International Journal of Project Management, pp 24-37, 2006

[11] Marjan Krasna, Ivan Rozman, Bruno Stiglic, How to Improve the Quality of Software Engineering Project Management, ACM SIGSOFT, Software Engineering Notes Vol 23, No 3, 1998

[12] M. Bronte-Stewart, Developing a Risk Estimation Model from IT Project Failure Research, 2005

[13] M. Heusser, Managing a Doomed Software Project: Practical Suggestions for Breaking the Bad News, InformIT, 13-Jan-2006

[14] M. Jorgensen, K.J. Molokken Ostvold, How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports, Information and Software Technology, Vol. 40, Issue 4, pp 297-301, 2006

[15] Project Management Institute Inc., A Guide to Project Management Body Of Knowledge, Third Edition