Coombesi, H. et al. (eds.) Proc. of FESMA 98 Business Improvement through Software Measurement, Antwerp, Technological Institute Publications, Antwerp, 305-313.

A CASE STUDY ON SOFTWARE PROJECT MANAGEMENT IN INDUSTRY – EXPERIENCES AND CONCLUSIONS

P. Mandl-Striegnitz[1], H. Lichter[2]

1 Software Engineering Group, University of Stuttgart

2 Department of Computer Science, Aachen University of Technology

Abstract

In this paper we present and discuss the findings of two case studies on software project management in industrial software development projects and the conclusions drawn from it. These studies were motivated to improve software project management capabilities. First, we describe how these studies were organized and performed. In the main part we present our findings and conclusions showing that there are strong deficits in project management quality. Based on these findings we briefly describe the structure of an improvement program aiming to remove or reduce those deficits.

Keywords: project management, case study, software project management improvement

1. INTRODUCTION

Many software projects are faced with a common situation: They fail in developing the required functionality within their schedule and planned budget; the results often lack the required quality. Thus, during the last years several companies have started initiatives to improve their software development. These initiatives mostly focus on improving the software processes and the technology used during software development. One area often underestimated but crucial for every software development project is project management. Project management is one of the key factors influencing the project success or failure.

In this paper we present and discuss the findings of two case studies on project management in industrial software development and the conclusions drawn from it. These studies were motivated by the company's goal to improve software project management. Before an improvement program could be developed, we first had to identify the strengths and weaknesses mainly focusing on project management. We also wanted to get an overview of the problems project managers have to face in industrial software projects contributing to schedule and budget overruns as well as to software systems with poor quality. Thus, at the very beginning a project management assessment was performed using on-site interviews with software project managers. The results of this first case study have been summarized in an assessment report (Lichter et al., 1996). To verify the qualitative data collected during the assessment a second investigation was performed by measuring quantitative data of a software development project.

Based on the results of both studies we are currently implementing a process improvement program mainly focusing on project management aspects. The key elements of this program will be presented.

2. DESCRIPTION OF THE CASE STUDIES

2.1 The First Case Study: A Project Management Assessment

At first we conducted a study whose purpose was to assess and better understand the current practices and problems of software project management as well as how they impact software development projects. The investigation was performed by means of structured, on-site interviews with software project managers. The questionnaire used for these interviews covers more than 70 factors potentially influencing the process and outcomes of software projects (Drappa, 1993; Jones, 1986; Paulk et al., 1993). The questions mainly address human factors and organizational aspects. In case the interviewed project managers had managed more than one software project, they were asked to answer the questions for each of these software projects. Thus, the project managers could supply all the information they had acquired. But it became apparent that the same project manager answered most of the questions identically for every project he had managed.

The data in the following sections describe the kinds of information we collected. The complete definition and planning of this study can be found in Lichter et al. (1996).

Project Profile

This aspect comprises characteristic attributes of software development projects. This includes attributes such as project kind, project nature, project scope, project class, and project type. Furthermore we recorded items such as the organization structures used on the project, project duration, and the application domain.

Project Environment

For this aspect it had to be analysed in which way the project environment had affected the course of the project and in particular project management activities. In our investigation we considered the following environmental factors: senior management, users, customers, and subcontractors. Some of the topics we explored concerning environmental issues include:

- Goals or constraints levied against the software project by senior management, customer or user demands, e.g. schedule and/or budget constraints, staffing constraints
- User experience and participation during the project
- Prototyping method
- Customer involvement during the project
- Stability of requirements
- Agreement or disagreement on project goals
- Influence of customers, users, and senior management on the progress of the software development project
- Use of external personnel on the project
- Organization of distributed software development

Project Management

To assess the strengths and weaknesses of project management we measured the following variables:

- Project management experience
- Time and effort spent by the project manager for management activities
- Methods and tools used for project planning

- Methods applied for project control
- Responsibilities for project management activities
- Quality assurance function and staffing

Project Staff

Since project staff skill mainly influences the software development project, we collected information on the qualification of team members, experience in the application area, experience with the programming language, experience with methods and tools, and experience with the development hardware used in the project. Furthermore we recorded data on personnel management capabilities such as staff availability, training of team members, effort spent for communication, and team continuity.

2.2 The Second Case Study: Measuring a Software Project

Based on the results of the first assessment we performed a second investigation by measuring an already completed software development project in considerable detail. Its goal was to verify whether the qualitative and subjective information collected during the first study could be validated and melded with the quantitative and qualitative data of a measured software project. Measuring a software project requires multiple sources of information. Hence, the empirical investigation was implemented as follows: At the beginning, as in our first study, we performed structured interviews with the technical and the project manager of the examined software project. In addition we executed written questionings with the project staff. Some information concerning defect and change management data was provided by a tracking system used in the software project. Data on the project key deliverables such as specification, source code, and user documentation were collected by applying several code, size, and complexity metrics.

The following section gives an overview of the kinds of information we recorded for the examined software development project in addition to the data already described in chapter 2.1 (Jones, 1997).

Project Totals

To get an overall view of the project we reported summary information of the overall project effort, schedules, cost, and the total number of defects found. To get an overview of the performance of the investigated software project we compared the planned and actual items for overall effort, schedule, and cost.

Project Tasks

For each development task that had been performed we recorded detailed information on resources, effort, schedule, and cost. Furthermore we were interested in the documents that had been produced in each task as well as in the sizes of the deliverables in terms of page counts.

Project Environment

To analyse the influence of the participation of users and customers as well as the involvement of external staff on the software development project we measured quantitative factors such as effort, schedule, and cost of user, customer, and external personnel by tasks and for the total project. To explore the influence of external staff we also identified the components of the delivered system that had been produced by subcontractors.

Project Management

In our second study the variables we analysed for the aspect of project management were considered in more detail. Hence, we collected information such as

- Project management training (training possibilities offered to project managers)
- Methods and tools used for progress tracking
- Methods and tools used for cost tracking
- Methods and tools applied for controlling quality and productivity
- Risk management (methods used for the analysis and monitoring of risk factors)

We were also interested in measurement factors like what kinds of metrics had been recorded during project development, whether these metrics had been adapted to the project characteristics, and what kinds of quantitative data on productivity and software quality had been collected.

Project Staff

Some of the items we recorded on the project team include:

- Staff availability including quantitative data on the percentage of time each team member had been assigned to the project
- Project team roles and responsibilities
- Size and structuring of project team and project management by month
- Changes in project climate during the project
- Motivation of project staff
- Unpaid effort/month – total amount of unpaid effort
- Participation in project tasks
- Staff turnover and attrition
- Participation in training courses

Methods and Tools

For each method and tool used in the software project we recorded information on the schedule of application, the impact on productivity and on quality, the costs caused for purchasing the tool and for training as well as the usefulness of the method or tool. Furthermore we were interested in the general tool availability. In case of defect removal methods, some more information was required. To explore the effectiveness of quality assurance activities applied in the project we collected information on the number of defects found and the effort spent for the quality assurance activity. For every defect we recorded the origin, severity, the amount of effort required to fix it, and the quality assurance activity that had detected the defect.

Project Outcomes

For each document produced during the project we were interested in the nature of the document (e.g. specification, design document, user documentation) and the number of pages in total. For source code we measured the size in lines of code per baseline. We distinguished different code classes such as delivered code, generated code, prototyping code, test code, and reused code. In addition we measured the complexity of the source code per baseline.

To measure productivity we were also interested in the number of documentation pages and in the number of lines of code produced per calendar month, and the number of personnel involved.

Project Effects

To evaluate the project success or failure it was important to explore value factors of the project such as the enhanced capabilities, the enhanced moral, the satisfaction of users, staff, customer, senior management, and project manager, the value of competitive advantage and market share, and the value of direct and indirect revenues from the project (e.g. new commissions).

3. RESULTS OF BOTH CASE STUDIES

In this section we present the main findings of both studies. We concentrate on the weaknesses we identified by means of our investigations to motivate the improvement activities described in more detail in chapter 4. We mainly focus on problems concerning software project management.

Inadequate Amount of Time Invested for Project Management Activities

The first investigation shows that the effort project managers invested in leading a project was in the range of 5 to 100% of their overall working time. Although even those project managers who spent less time for project management activities (5-35%) regarded this time as adequate and sufficient, analysis of the data indicates that less time for project managing and controlling often led to enormous overruns in cost and schedule. As a consequence, 75% of the investigated projects were not completed in the planned schedule – the deviations from schedule were up to 150%. Another interesting aspect in this context is that most project managers regarded bad education of team members, moving requirements, or unstable developing environments as the main reasons for those overruns in schedule. If more time was spent in managing the project and controlling its progress the deviations from schedule were minor (from 20-70%). Here project managers regarded too optimistic planning as the main reason for the overruns in schedule.

The second investigation verifies these results. It shows that the project manager could only spend a maximum of 50% of his overall working time for leading the project, because at the same time he participated in up to four different software projects. The project, too, was completed with enormous cost and schedule overruns.

Often Project Managers did not have the Right Skill

To effectively manage a project the person who leads the project must have managing skills. We have encountered that often people were selected as project managers based on their software development capabilities. This seems to be counter-productive. Project managers who had lot of experience in developing software and in solving technical problems often did not have good managing skills. If these people had to manage projects it is obvious that they tried to reduce management activities to a minimum. From the perspective of the organization we saw three main pitfalls if project managers did not have good managing capabilities. First, the projects were insufficiently managed leading to a lot of problems. Second, the development skills of those people were not used. Third, people that had to perform activities they did not like were often dissatisfied. This resulted in a reduction of their overall motivation.

Inadequate Training for Project Management / Inadequate Usage of Project Management Techniques

Both investigations show that project managers never had participated in training programs on project management techniques. Thus, the only method that was applied by every project

manager was planning of schedule and milestones by means of Gantt charts because this was supported by tools (e.g. MS Project). But other essential techniques like systematic risk assessment or cost estimation were neither known nor applied. Risk assessment was often done only at project start; the estimation of effort and time was always done based on their experience. In most projects there was a lack of regular and explicit updates of the project plan. We identified a clear deficit of project tracking and control.

Our conclusion is that organizations and project managers do not know how important and useful the application of basic management techniques is in order to support the project manager in effectively leading the project. As a consequence, organizations often do not spend time and money in educating project managers. While training on new or unknown software development techniques (like programming languages) is considered as necessary or mandatory, training on project management techniques is often neglected.

Important Quantitative Data were not Available for Project Management

We have encountered that organizations did not systematically collect data on projects in order to use this information to estimate new projects and to control quality and progress of the current project. In our opinion this is a crucial point. Since data was not collected and evaluated, the only source of information for planning new projects was the experience of project managers. We do not underestimate this, but we believe that an organization will get benefits from systematically collecting and evaluating projects. Looking at the answers we got from project managers it is obvious that there were often unrealistic estimations concerning the productivity of team members or the effort necessary for inter-project communication. The second investigation for example shows a dramatic underestimation of the effort needed for producing the user documentation. Because of this, all testing activities were cancelled in order to complete the user documents. If there are no metrics defined to measure quality and progress, project managers cannot realistically plan for further quality assurance tasks or effort necessary to complete the software product. In addition measuring a software development project is the basis for any kind of software improvement.

Often Quality Assurance was Neglected

Another aspect we like to mention here concerns quality assurance. Our investigations show that the value of quality assurance activities was underestimated by project managers as well as by the development organization. Some project managers stated that their projects had no quality assurance function at all. Sometimes the project manager or the developers were responsible for quality assurance. In case of the project investigated in the second study, it was planned to perform quality assurance activities such as code reviews and design reviews. But although these activities were mentioned in the project plan no effort was assigned for these tasks. As a consequence, during the project these activities had never been performed. Both investigations show that in case the project ran into schedule problems, the most common way to short-cut it was by skipping quality assurance activities. That means that the schedule constraints always had priority over the quality goals. This may be reasonable in some cases, but it should not be the standard case.

Inadequate Support from Senior Management

All project managers stated that better support from senior management would help to improve their job. The main problems resulted from poor communication between project managers and the senior management. Often the only communication media were written status or quality reports. In case of the fully measured software project, the senior management took over planning and control as a consequence of the considerable delays. One

significant reason for these delays had been major requirements changes after the requirements phase. But although the project manager mentioned the risk of highly unstable requirements in every status report, there was no support from senior management until the project seemed to be out of control.

Distributed Software Development Requires Better Project Management

If software is developed geographically distributed at different sites, project management becomes harder. Some project managers regarded this as the main reason that the overall effort for communication was considerably higher than estimated (about 30% more). In two cases the main problem was the loss of information because the interfaces between all organizations involved in the project were not exactly defined. Other problems that were reported were conflicts between the organizations and negative impacts on the project based on co-ordination deficits.

4. IMPROVEMENT PROGRAM

After we had identified the deficits of software project management as well as the main problems project managers had to face in the examined projects, we proposed a project management improvement program to remove or at least to reduce those deficits. When analysing the investigations' results it became obvious that most of the problems were concerned with managerial, sociological, environmental, and organizational issues. But despite of this, the company's activities for improving the situation had concentrated on technological issues.

In the following we describe the main activities the company should implement for improving the current situation of project management.

Set Up a Training Program for Project Managers / Use Adequate Project Management Techniques

Both investigations show that project managers often did not have the right capabilities for performing their management tasks. Hence, the company must set up training programs on project management to improve the qualification of project managers. A project manager must not start leading a software project before having completed the training program. The training program should conciliate knowledge on topics such as project planning (project sizing and estimation, project scheduling), progress tracking and reporting, quality control, risk analysis and management, adequate project management tools, personnel management (team building and leadership), communication skills, and quality assurance techniques. To enable effective project management project managers need to learn what information must be included in project plans, status and quality reports. In our studies it became obvious that the available project management tools were inadequately used. Thus, project managers need better training on the tool facilities (e.g. progress tracking facilities of MS project). It is also important to emphasize the need for regular updates of the key documents such as the project plan, the requirements and the functional specification.

Nevertheless, organizations have to ensure that people selected for managing software projects must have strong managing capabilities. Furthermore organizations have to find ways for people with strong development capabilities to get ahead concerning their occupational career.

Establish a Permanent Software Measurement Program

The establishment of a permanent measurement program is a significant step of the whole

improvement program, because the collection and evaluation of substantial quantitative data is an important basis for effectively planning and controlling software projects.

First of all we had to define what information should be measured to improve estimation of future projects. To get accurate schedule information it is important to measure software documentation variables quantifying for example the volume of different document types produced in a software project. The sizes of each deliverable can be measured in terms of page counts in case of paperwork and in lines of code in case of source code. Furthermore they need to measure productivity when producing these deliverables to be able to estimate the effort for paperwork and source code.

Second, software measurement can be used for progress tracking. Having accurate data problems can be noticed earlier in the life cycle and probably be solved without causing too much damage. For example, if the project uses an adequate change management system, project progress can be monitored by the number of completed change requests.

Third, software metrics must be applied to control software quality, because finding and fixing defects is one of the most expensive tasks in all projects. For each defect the following data has to be recorded: origin, severity, and amount of effort required to find and fix them. These data should also cover the quality assurance activity used to find the errors. For measuring product quality it is highly recommended to use an adequate defect tracking system.

Our studies show that most of the project data was already available by means of e.g. defect and cost tracking tools and project reports. What lacks is a procedure to systematically collect and evaluate these data.

Plan for Adequate Quality Assurance Methods

Both studies show the considerable deficit of quality assurance in all examined projects. We conclude that often a “mature development culture” is missing in software organizations. In a mature development culture quality assurance activities are planned and performed during the project. Quality assurance must be an integral part of all development tasks. Hence, to improve this situation top management decisions are necessary as well as good educated project managers and developers concerning quality assurance know-how.

Project managers should at least plan for reviews of the specification documents, design reviews, and code-inspections. The test series of the source code should include unit testing, regression testing, function testing, system testing, and acceptance testing. It is important to emphasize that the effort planned for quality assurance activities must be expended even in case the project runs into schedule problems to avoid an even greater disaster.

Initiate Policy and Organizational Changes

As mentioned above, project managers often invested only an inadequate amount of time for project management activities. Our conclusion from this finding is that on the one side project managers have to concentrate on managing the project. Courses on project management must stress on the importance and complexity of project management tasks. On the other side organizations have to ensure that project managers are able to effectively manage their project, i.e. project managers must have sufficient time to perform the necessary management activities. As a consequence, the policy that often causes project managers to divide their time among several projects needs to be changed. Especially in case project managers have to lead a large project of several hundred manweeks effort it should be the only project they are assigned to. The same holds for distributed projects. We conclude that distributed development

projects need much more management effort than local projects. This must be taken into account when selecting the project manager as well as during project planning.

Based on our results we strongly recommend another change in the company's policy. We conclude that continuous communication between project managers and senior management is necessary to early identify and solve problems, to prepare decisions, and to avoid conflicts. It is important to create an open door policy. It must not only be possible for project managers to mention risks as soon as they are noticed, but be self-evident for senior management to react to these notifications immediately. Hence, senior management must support project managers not only when the project seems to be out of control and it is obvious that it runs into unacceptable schedule and cost overruns, but during the entire software project. They must support project leaders by setting up boundary conditions that help to successfully manage the project.

5. CONCLUSIONS AND OUTLOOK

In this paper we have presented the findings of two investigations on software project management in industry showing that there are strong deficits. We have proposed some improvement activities focusing on managerial, technical, organizational, and sociological aspects. Of course, improving software project management in a large company is a difficult and complicated task. We are at the moment implementing parts of this improvement program, e.g. collecting essential project data and training basic management techniques. Although so far we do not have quantitative data to validate the implemented activities, the feedback coming from project managers as well as from senior management is prevailing positive.

As we know from former improvement initiatives, the benefits will not come immediately. Hence, the implemented activities need support from senior management for years to be as successful as possible. It is important to annually perform investigations on a number of projects to explore whether the suggested activities overcome the noted deficits. Because of the established measurement program this can be supported on the basis of the data collected during software development.

References

Basili, V.; Selby, R.; Hutchens, D. (1986): Experimentation in Software Engineering. **IEEE Transactions on Software Engineering, 12** (1986), No. 7, pp. 733-743.

Deininger, M. (1995): **Quantitative Erfassung der Software und ihres Entstehungsprozesses**. Verlag Dr. Kovac.

Drappa, A. (1993): **Erhebung von Metriken in industriellen Softwareprojekten**. Diplomarbeit, Universität Stuttgart.

Elzer, P.F. (1989): Management von Softwareprojekten. **Informatik-Spektrum, 12** (1989), pp. 181-196.

Frühauf, K.; Ludewig, J.; Sandmayr, H. (1991): **Software-Projektmanagement und Qualitätssicherung**. vdf.

Jones, C. (1986): **Programming Productivity**. McGraw-Hill.

Jones, C. (1997): **Applied Software Measurement**. McGraw-Hill.

Lichter, H.; Mandl-Striegnitz, P. (1996): **Software-Projektmanagement in der Industrie – Erfahrungen und Analysen**. Bericht SL-2/96, Universität Stuttgart.

Paulk, M.; Curtis, B.; Chrissies, M.; Weber, C. (1993): **Capability Maturity Model for**

Software, Version 1.1. Carnegie Mellon University.