

Assignment 5 / Management of a Software Project

Kenan Esau

February 2001

Organization: TAE/Esslingen
Course: M.Sc Distributed Systems Engineering
Lecturer: Prof. J Rodney Turner

Contents

1	Introduction	3
2	Pre-Conditions of the Project	3
3	Course of the Project	4
3.1	Decisions taken	4
3.2	Decisions concerning Personnel	4
4	How it should be	5
4.1	Personnel-based decisions	6
4.2	The Milestone Plan	8
4.3	Responsibilities	10
5	Conclusions	11

1 Introduction

This assignment describes a software project whose objective it was to create a software which is capable of managing a whole little company. The software should be able to manage customers, the suppliers and the store. It has to be able to create commissions and reckonings. This was a very ambitious project of a little company which already had some products – running on MS-Dos – which were capable of doing the things described above. The first approach was to port this product which was written in an very old language called clipper to Windows NT/95. The second possibility was to do a complete rewrite with some new kind of tool/programming language.

In the next sections I will describe the opportunities which were possible at the start of the project and the approach taken. After this I will describe how it could be done to prevent a complete failure of the project.

2 Pre-Conditions of the Project

The situation of the project was like already described in section 1 that a solution for MS-Dos already existed. But since MS-Dos was very old and all customers demanded a version with a graphical user interface the company which maintained the MS-Dos product decided to port it to Windows. The old MS-Dos program was written in the language clipper. A main part of it was bought from another company and the functionality was extended from time to time to satisfy customer needs. Documentation for the source-code was practically not existent and the structure of the database was not documented too.

The source-code was in a very bad shape. File-names for example looked like “a0024532.clp” to satisfy the old requirement of MS-Dos for file names which said that there are maximum 8 characters allowed followed by a point and a file-extension with the maximum of three characters. The whole program consisted of a few hundred files with very unreadable file names like that shown above. Since the very cryptic structure of the file names and the fact of non-existent documentation no one really knew where what kind of functionality was implemented. The functionality and the size of the product increased more and more over the time and finally – since the total lack of documentation and project management – became completely unmaintainable.

The situation with the staff of the company was not ideal too. It was a very small company offering complete solutions for industrial partners. This means they sell hardware to the customers do the cabling for the network and provide the service needed by those customers concerning hard- and software. At the time of the start of the project there where

two technicians, two people doing the work in the office, the boss himself and one person who was capable of doing some programming. The boss of the company was also able to do programming and the management of the project. So there were two people capable of doing programming and the design of the database. These two people also had a little knowledge of the old program since they were the people who extended its functionality more and more.

3 Course of the Project

The following subsections describe the course of the project like it happened. Section 3.2 describes the decisions taken concerning personnel. Section 3.1 describes the decisions concerning the course of the project – which programming language was chosen what schedule was selected and so on.

3.1 Decisions taken

The first decision was to port the old MS-Dos application to windows. The first intention was to reuse as much as possible of the source code. But after analysing the old source code this decision was revised since it was found that it was almost impossible to reuse the mess of code which was present. The lack of documentation and personnel who really knew the program made a reuse impossible. Since the project had grown and grown by the time there was no real software design anymore. The old software seemed just to consist of a lot of patches. Thus it was finally decided to write a completely new application. Although the project leader still wanted to use bits of the old product to speed up the development.

Analysis of the present database showed that it was not usable anymore too. The column of some tables were named in a very cryptic manner and no one was able to tell what their sense was. So the first intention to use the database further had to be discarded too. The first step after this decision was to analyse the old database in more detail to be able to convert the old database to a new format which had to be defined at that point of time. This step was inevitable since the old customers had to be able to use their old data. There was no point in losing the old data. The conversion had to be complete and correct.

3.2 Decisions concerning Personnel

Since the company was not able to provide enough personnel to complete the project in reasonable time two students who were doing their industrial internships at the company were instructed to do the programming work and the analysis of the database. The option to commission some external programmers was not used since the cost would have been

to high for the company. Nevertheless the boss of the company meant it would be possible to commission some external programmers if that would be necessary later if the project runs late.

4 How it should be

The decisions described in the previous sections were mainly based on an ad hoc approach rather than on analysis or project management. This explains why it was not possible to produce a working product under the circumstances which were presented in the preceding sections. The first step should be to analyse the present situation very carefully and then decide what to do. The decision not to use the old source-code was right but it could have been achieved a lot earlier if the project leader would have had a closer look at the old source and the state in which they were. After this decision no one should even waste a thought about using a single line of code from the old program to try to speed up the development of the new one. If the old one was bad designed and the circumstances are as described in the previous sections all you can do is to start all over again.

Before the start of the project the objectives and the overall strategy should be clear. Since this project is a project with very well defined objectives and its goal is to develop a product, a milestone based planning approach would be appropriate. The whole product is easy to divide into sub-components:

1. Customer Management

This software module concludes organization of the customer data and the correspondence with the customer. The correspondence consists of offers, commissions, delivery notes.

2. Supplier Management

This part is very similar to the Customer Management. The address data can be handled the same way as the address data for the customers – it has only to be saved in a separate table. The part which is responsible for the correspondence with the supplier is also very similar to that of the customer. Thus it may be possible to reuse a lot of the code of the customer-component.

3. Store Management

The store-component is very closely linked to the correspondence parts of the suppliers and the customers. If a customer orders a part it has to be taken from the store. If the number of present parts of a certain type drops below a certain value automatic ordering of that part might be needed.

4.1 Personnel-based decisions

Now it should be easy to create a milestone-plan for each of those sub-components.

One of the key success factors is that all of the stake holders agree with the objectives of the project. In a normal case it is very unlikely to get the agreement of all stake holders but in this case it was possible. The stake holders for this project are:

1. Company's boss
2. Customers
3. Company's programmers
4. Users

All those stake holder would benefit from a successful outcome of the project. The boss would make a lot of money with the new product. The present customers would never think of changing their vendor and if the product meets their requirements it would be easy to get new customers to sell the product to.

The customers themselves would like to increase their productivity further. They would benefit from a new product in many ways. Besides this they are able to take some influence on the development.

The programmers get an interesting work and are able to increase their influence within the company. Maybe it would also be possible to earn more money if the product is a success.

The Users are the only ones who could be afraid of the new product. But since they already worked with a similar solution no one of them is made obsolete by the new system. Thus they should be informed about the new software and they could actively take part in the development process eg to increase the usability of the new system.

So here we have got the very rare case that all stake holders agree with a project and its objectives. Thus it should be possible to successfully complete this project.

4.1 Personnel-based decisions

One problem still remains. There is too less qualified personnel who is able to do software design implementation and test. To give this work exclusively to some students who are doing their industrial internships at this company might be too dangerous. These students can do a lot things but their work should still be watched carefully by experienced

4.1 Personnel-based decisions

designers and programmers to guide them to a good solution.

To believe that it is possible to add programmers at a later state of development to speed up a late project is completely wrong (see [1]). Adding programmers to a late project slows it down since these new programmers have to be trained by the experienced staff. Additionally communication gets more and more complicated if more people are involved.

A good way would be to find one or two experienced designers who are able to do the software design and a part of the project management at the beginning of the project. The implementation tasks could be done by the students. Even the software design could be done together with the students so they are able to learn from the experienced designers.

Assuming that it is possible to get two experienced software designers and two students doing their industrial internship the following resources are available.

- 2 experienced software designers
- The company's boss
- One programmer
- Two students

According to those resources the following teams can be formed. Some of the people are in more than one team but I think that is quite realistic.

1. System Design Team

This team will be responsible for the design of the software and the database. **This team consists of:**

- (a) Experienced designer 1
- (b) Experienced designer 2
- (c) Student 1
- (d) Student 2

2. Implementation Team 1

The two implementation teams will be concerned with implementing and testing. If one team implements one component the other team has to test it. Testing should be performed throughout the whole development process but there is also a phase of heavy testing which should not be performed by the developer of the component.

This team consists of:

- (a) Experienced designer 1
- (b) Student 1

4.2 The Milestone Plan

3. Implementation Team 2

This team consists of:

- (a) Experienced designer 2
- (b) Student 2

4. Project Management Team

The task of this team is to manage the progress of the project and guide the other teams through the project. Risk analysis is also done by this team. **This team consists of:**

- (a) Experienced designer 1
- (b) Experienced designer 2
- (c) The company's boss

5. System Integration Team

The System Integration Team has to perform all tasks which are needed to integrate the different software modules and to integrate the prototype(s) at the test-customer's system. The "native" programmer of the company is only member of this team since he has a lot of other tasks in the company. **This team consists of:**

- (a) Programmer
- (b) Student 1

4.2 The Milestone Plan

This section contains a more detailed description for the milestones in the chronological order. This is intended to be an example for the module customer management. Similar plans have to be created for each module of the whole project and these plans have to be incorporated in one big project plan.

Figure 1 shows the different milestones in their chronological order. The abbreviations/names of the milestones and the actions needed to accomplish a milestone are described below.

As you can see below an approach with several prototypes is used. Thus an iterative software process model – like the spiral model (see [1]) – would be good. The spiral model, originally proposed by Boehm, is an evolutionary software process model which provides the potential for rapid development of incremental versions of the software [1].

Milestone **PR0**:

This milestone defines the start of the project.

4.2 The Milestone Plan

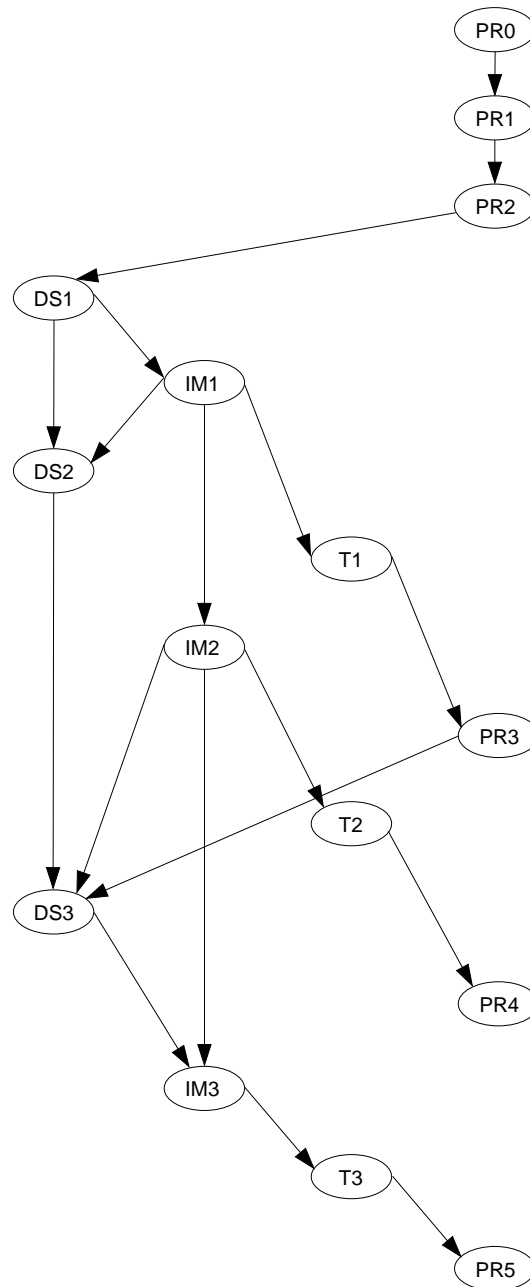


Figure 1: *Milestone Plan for the customer module*

Milestone **PR1**:

To accomplish this milestone the requirements have to be identified and written down in a specification book.

Milestone **PR2**:

All use cases have to be derived from the milestones. These use cases have to be written down in a design specification.

Milestone **DS1**:

Design of first prototype finished.

4.3 Responsibilities

Milestone **IM1**:

Implementation of the first prototype finished.

Milestone **DS2**:

Design of second prototype finished.

Milestone **T1**:

Testing of the first prototype and error correction have to be finished.

Milestone **IM2**:

Implementation of the second prototype finished.

Milestone **PR3**:

Presentation of the first prototype.

Milestone **T2**:

Testing of the second prototype and error correction have to be finished.

Milestone **DS3**:

Design of the final version finished.

Milestone **PR4**:

Presentation of the second prototype.

Milestone **IM3**:

Implementation of the final version finished.

Milestone **T3**:

Testing of the final version and error correction have to be finished.

Milestone **PR5**:

Presentation of the final version and integration of the final version of the customer module with other parts of the project (eg. module for the suppliers or module for the store).

4.3 Responsibilities

The responsibility chart shows which team has to perform which tasks to complete a certain milestones. The schedule of the tasks and the actions which have to be taken are described ins section 4.2.

5 Conclusions

Project Responsibility Chart						
X – executes the work D – takes decision solely d – takes decision jointly P – manages progress T – provides tuition on the job C – must be consulted I – must be informed A – available to advice	<i>Boss</i>	<i>Project Management Team</i>	<i>System Design Team</i>	<i>Implementation Team 1</i>	<i>Implementation Team 2</i>	<i>System Integration Team</i>
Milestone Name						
Project Start	D	d	I	I	I	
Requirements	d	XP	X	I	I	
Use Cases	d	XP	X	I	I	
Design (1st prototype)	I	P	dX	X	X	
Implementation (1st prototype)	I	P	IA	X		
Design (2nd prototype)	I	P	X		X	
Test (1st prototype)	I	P		I	X	
Presentation (1st prototype)	I	C		A	A	X
Implementation (2nd prototype)	I	P	IA	X		
Test (2nd prototype)	I	P		I	X	
Design (final version)	I	P	X		X	
Presentation (2nd prototype)	I	C		A	A	X
Implementation (final version)	I	P	IA	X		
Test (final)	I	P		I	X	
Integration with other Modules	I	Pd	I			dX

5 Conclusions

The project described in this assignment could have been a success since the preconditions were very good. But the lack of planning and management made it a complete failure. The failure of the project might have also been prevented if there would have been enough trained personnel. Some wrong decisions would not have been taken if the trained personnel would have been available.

In middle to big sized projects a certain amount of management is needed. If there would be no management at all no project can succeed.

References

- [1] Roger Pressman, Software Engineering – a practitioner’s approach