40

GUIDELINES FOR SOFTWARE PROJECT CONTROL

Working Group 01 (Control Centres)
of Study Committee 35
(Power System Communication and Telecontrol)

cigré

CIGRÉ – Study Committee 35

GUIDELINES FOR

SOFTWARE PROJECT CONTROL

Working Group 01

GUIDELINES FOR SOFTWARE PROJECT CONTROL

CONTENTS **PAGE**

GUIDELINES FOR SOFTWARE PROJECT CONTROL

1 INTRODUCTION

The purpose of this report is to provide guidelines for customer and supplier project control to ensure quality in developing software for telecontrol system projects. The report is focussing on project management from a **software** point of view, and is therefore not intended as a general project management manual. The guidelines cover all stages of a typical software project, from specification to final commissioning.

The functionality of control systems and equipment is today increasingly implemented with computers and software. Also conventional hardware units, such as process interfaces, communication controllers and protection relays, are now equipped with microprocessors – and software. This means that in the future software will be more challenging than hardware. It is also essential that the software development takes into account the software portability to make the software independent of the hardware platform.

The programmability of systems gives both the user and the supplier maximum possibility to implement advanced functions in the systems. This possibility, however, makes it extremely difficult to check, test and verify the functions in all possible situations and combinations. The new technology demands strict and systematic quality assurance in combination with project control.

The goal of this document is to give practical advice to the project groups handling the implementation of software projects, primarily for **Telecontrol (SCADA)** and **Energy Management Systems (EMS)** applications. The report will set up guidelines for the project management and project team. The ideas will serve both for delivery projects and for a company's internal development projects. In both cases there are the **supplier** part (developing organization or manufacturer) and the **customer** part (user organization or power company). The utilization of consultants is discussed also.

First, the importance of detailed specifications and project targets is discussed. Without the knowledge of the final user's needs and environment, the functions may not be implemented correctly. It is also important that all project team members are familiar with the project targets and limitations, not only functionality and time-schedule but also financial goals and resource availability.

The fundamental part of a project is the project plan. This plan defines in detail the structure of the project, including organization and management, project phases, specifications, budget, time-schedules, tests and verifications, training, installation and commissioning, and maintenance arrangements. In order to manage technical and other changes during the project it is essential that certain rules are established on how to deal with modification requests without causing disturbances and problems to the project.

The final result of a project is usually due to the capability of the management to control, advise and motivate the resources in the correct direction. An essential role in this aspect is the project team organization and its functionality. It is important that the relations between all parties is agreed upon in advance, in order to minimize difficulties for the project. Relations should be positively maintained both within the project and with the customer/supplier and external experts, such as consultants and subcontractors.

The tasks should be divided into smaller entities to be able to supervise, follow-up and validate the progress of the project. They can then also be validated more easily. The follow-up also covers the supervision of costs, working hours and resources.

The validation of the final product starts in an early project stage. The testing philosophy is part of the project plan. Detailed specifications should also be made for unit testing, system integration, process simulation and test reporting.

International standards, such as the ISO 9000 -series, are available for the development environment and organization. There are also software tools available for software development and version management. The use of standardized methods makes it easier to evaluate the quality level of the product and facilitate future maintenance.

It is a well-known fact that quality cannot be added into a product later, it has to be built in.

This report will give recommendations about what should be done and what precautions should be taken at the various stages of the project to guarantee a qualitative final result. As a help for the reader and project manager, check lists are included in the report.

KEYWORDS

Software - Project - Management - Quality - Guideline - Specification - Requirement - Telecontrol - SCADA - EMS.

2 PROJECT GOALS AND REQUIREMENTS

2.1 A Few Questions As An Introduction

The purpose of this report is defined as *"to provide guidelines for customer and supplier project control to ensure quality in developing software for telecontrol system projects"*. The following basic fact cannot be emphasized enough :

The quality of software is extremely important for the correct operation of telecontrol and EMS systems.

It is well-known that many difficulties still occur during systems development and operation, that resultant functions often do not exactly correspond to the user's expectations and that the period following the system delivery is a very hardworking one, which may last a long time before users are satisfied.

In fact, the more technical advances, standard functions and programs that are offered, the more we have to make choices and to accept limitations in defining our needs.

The objective of this chapter is to underline the importance of the user requirements definition, the role and the obligations of the customer, and what is still relevant, sometimes more relevant than a few years ago, in this field of telecontrol systems.

2.2 Definition Of End User Requirements

2.2.1 Focusing On The User

Because of lack of experience and technical limitations, the first generation of telecontrol systems was usually specified by telecontrol and EMS experts. It was very often the same for the second generation, although the technical concepts were more familiar to the users, because of the emergence of many possibilities to be experimented on and the inability to greatly improve the functions offered to the users.

The result is that the end users had, and sometimes still have, to adapt themselves and their needs in terms of technical solutions, instead of describing the characteristics of the energy system and its associated problems without imagining new processing possibilities.

Today many basic functions are well-known. Technical solutions have been stabilized and fully tested, so that it is no longer necessary to alter them. In addition, many standard modules are available, both in the software and the hardware field.

The definition of a new system consists more and more of choosing which traditional functions are necessary, how to combine them with the existing operational environment, to identify the dimensional parameters and to deal with data base design and configuration.

Finally, the definition will be based on a global functional architecture and the technical specification will first have to identify the way to integrate different modules in this architecture, in accordance with a set of constraints to be respected.

The primary quality condition is to separate formally two complementary points of view:

* the one of the user, who "*knows what*", and
* the one of the Telecontrol/EMS experts, who will employ their "*know how*" to help the user.

The user must be at the centre of the development!

2.2.2 Customer/Supplier Relationship

Different development organizations, in internal or external project teams, are possible regarding the political choices and the resources of the utility. Nevertheless, it has to be underlined that the "*know how*" becomes more and more a question of specialists and that it becomes more difficult - and expensive - to maintain such a competence internally. It is better for a company to develop its knowledge in the field of the energy system dynamics and to entrust the software development to a supplier who becomes a kind of partner.

Moreover, the user staff has neither time enough available, nor the possibility to consider the project from an external point of view or to control and follow the development from the beginning to the end. This role can be very effectively filled by a consultant, who will assist the end user in describing his needs, by his contribution as an expert in answering the supplier questions and by verifying that the rules of the art are respected. The consultant team is very often composed of internal staff and external assistants, coming for example from another utility which has already acquired this competence.

This distribution between three roles - user, consultant and supplier - is the key of a good control throughout the project.

In every case, internal or external consultant, internal or external supplier, it is essential:

* to give the leadership of the project to the user's point of view, for example through a **Project Direction Committee**
* to provide the user the possibility to intervene in the project at any time, and to validate the completed work at the end of essential phases of the development
* to establish a formal customer/supplier relationship between the parties, that clearly specifies the rights and obligations of each partner.

During operation of the system, the maintenance and the follow-up may be provided by either the user or the supplier, depending on the political choices and the resources of the utility.

2.2.3 Successive Implementation Phases

The responsibility for the phases of the project changes during the project. Typical detailed project phases are:

a. Pre-specification

Examination of the state-of-the-art, studies and demonstrations of existing systems or systems under development.

Users and consultants are involved with this phase for between 6 and 12 months.

b. User Requirements Specification

The User Requirements Specification forms the user's enquiry for the software system delivery. This specification can be divided into three parts:

- **The Functional Requirements** consist of results from the analysis of the existing system limitations and the difficulties that occured during the operation, interviews and demonstrations with the end users, fundamental choices and limitations for the next system, and in accordance with time and financial constraints.

  Users and consultants are occupied with these requirements for between 6 and 12 months.

- **The Technical Requirements** include additional specifications such as general architecture, environmental requirements, use of standards, dimensional parameters etc., so that the complete specification allows a tender.

  The technical requirements occupies the user and consultants for between 12 and 18 months, including tender and choice of an eventual supplier.

- **Commercial terms and conditions** include the user's commercial requests for the supplier's proposal.

c. Supplier's Proposal And Table of Conformance

This represents the supplier's response to the user's request for a proposal and indicates how the supplier would meet the user's requirements. The Table of Conformance (also called Statement of Compliance) indicates whether the supplier will meet the user requirement, take exception or provide an alternative.

d. **The contract**

This can take many different forms but usually includes the User Requirement Specification, the Supplier Proposal and Table of Conformance as well as letters and memos that clarify any differences between the above two documents. It can also provide for any new requirements mutually agreed to during the evaluation and negotiation phases.

e. **Functional Specification**

This is a series of documents that outlines what and how each item in the List of Deliverables is to perform. This includes all software and man-machine functions, as well as hardware. System integration performance is also included.

There may be several tens of individual Functional Specifications for a large system. They must be agreed to by both the user and the supplier. The system design is frozen at this point, unless there is a major reason for change.

The Design Specification and Test Plans can now be developed from the Functional Specification.

This phase usually takes between 3 and 9 months.

f. **Test Plan**

A Test Plan is developed to test each of the functions in the Functional Specification. It contains enough detail so that when the Test Plan is followed and completed, the user will accept the system and the supplier will be paid. The Test Plans are used in all testing. The user and supplier agree to the Test Plan.

g. **Design Specification**

The Design Specification is developed from the Functional Specification and tells how each of the functions will be fulfilled.

The user usually reviews the Design Specification but does not have approval rights. This phase lasts between 6 and 12 months.

h. **Development And System Integration**

Development (coding) of each software module can now begin once its design specification is complete. Each software module is tested individually and as it is integrated with the others.

It is not easy to involve the user in this part of the project. However, it is useful to provide for additional and successive demonstrations and to deliver different documents such as acceptance tests specification and user and installation documentation as soon as possible. However, it is absolutely necessary to refer to the user everytime a difficulty appears from the functional point of view, such as the need for an additional specification or discovery of any incoherence.

Usually, the user is not involved in this phase unless some of their employees are on loan to the supplier for training. In this case, the user's employees are treated as though they are the supplier's employees and are involved as workers in software module development.

i. **Acceptance Tests**

There are several tests involved in the Acceptance Tests.

- **Pre-Factory Acceptance Tests (Pre-FAT)** are conducted on each individual module as it is developed and again as it is integrated into the whole. The supplier usually conducts a test prior to beginning the formal Factory Acceptance Tests (FAT). The user is not involved in these tests.

- **Factory Acceptance Tests (FAT)** are conducted to certify that the system meets all of the contract system performance criteria except for availability. Tests are done by the users according to the Test Plan with the supplier witnessing the tests. Corrections are made to the variances and the system is retested. The system is shipped when all variances agreed to have been corrected.

  Please note that it is much easier and more cost effective to make the changes in the factory than in the field. It also does not interfere with operations and have the problems of getting permission to take the system out to make the change, or have the risk of making things worse while the system is in service.

- **Site Acceptance Test (SAT)** is a brief test conducted after installation to verify that the system works as it was tested in the factory. Approximate duration is 2...8 weeks.

- **Field Test** tests the system in the field as it is connected to all pieces in the system including the data base, RTU indication, control and telemetering points, communication systems, and other computer, control, SCADA and EMS systems. Approximate duration is 6...12 weeks.

- **Commissioning Test** – a combination of SAT and Field Tests.

- **Availability Tests** are conducted in the field to ascertain that the individual functions and the system will meet the availability criteria in the contract. Duration usually 3...6 months.

In this planning, the approximate times given above are guidelines and depend mainly on the size of the project. Nevertheless, it is clear that the time spent at the beginning of the project is a wise investment for the success of the project.

For example, it is essential to define from the beginning the objectives of quality, the organization of the acceptance tests, the commissioning and the general specifications of staff training, maintenance, hardware and software. On the other hand, the hardware should be chosen as late as possible.

Many conflicts can be avoided by requiring the vendors to include the commercial conditions with the tender document.

2.2.4 End User Requirements And Fundamental Choices

The end user requirements definition is an essential part of the project:

- The system must meet the needs of the user.

- An error or an omission at this point of the project may have heavy consequences for the continuation of the development.

Typically, the customer has an existing system in operation, so one of the major concerns is the renewal of this existing system.

Another report from CIGRÉ SC35-WG01 gives guidelines for these concerns. Key points from this report are:

- Examine the limitations and the operational problems met with the existing system.

- Identify required new functions, regarding the improvement of performance, end user comfort, secondary analysis and automatization of the operation, etc.

- Analyze the changeover phase, with all contraints which have to be considered, especially to ensure the continuation of the operation without disturbing the user.

Two important parameters have to be taken into account before the final choice is made: financial considerations and the time necessary to get the new system operating. It may be possible to wait longer before carrying out certain new functions and enhancing the system, provided that there is flexibility in the development.

Fundamental choices must also be made on quality.

> **It is essential to understand that quality has a price and that many actions and procedures during the development will depend on the specified level of quality.**

It may be difficult for a customer to accept the idea that the quality of a development may not be total and, in consequence, to specify a kind of "no quality". In fact, the comparison is possible, for example, with the availability point of view. Everybody hopes for a 100 % availibilty rate, but it is well-known that:

- a 100 % availibility rate is impossible
- a 99 % availibility rate is easy to reach without demanding expensive devices
- the over-costs to improve the availibility rate become rapidly exponential.

For the same reason, it is not easy to give a reference scale so that the user may choose the required level of quality.

One possibility is as follows: classify each function by the consequences of a default occuring in the system or in a subsystem. For example:

- possible consequences for the security of people (example of the control system in a nuclear plant)
- severe financial consequences in case of material destruction (example of the remote control of a substation)
- limited financial consequences (example of statistics subsystem).

It is clear that in the first case, the quality inspections and the testing procedures must be systematic and redundant. The third case is an example where it may be possible to accept residual errors.

In conclusion, it is necessary that the user requirements specification includes all the parameters that will have important consequences for project implementation. It is essential that all these parameters are the choice of the user, but that they also come about as a result of technical discussions and commercial negotiations with the consultant and the supplier.

2.3 User Environment

The system to be developed is always a part of another system. To be complete, the specification must include the environmental specifications.

2.3.1 The Energy System

The power system, such as the end user's system, is by definition the system to be developed. Every part of the project must deal with this system.

It is essential that the supplier staff, even a programmer, has a good comprehension of the energy system. If not, it is possible that the supplier, for example in the case of omission in the functional specification, will make choices without comprehending their real consequences for the end user and for the energy system management.

It is therefore a good investment to organize technical visits and presentations for both the supplier staff and internal staff at the beginning of the development and to maintain this knowledge throughout the project.

However, the description of the energy system is fundamental for two reasons:

- the size of the system and its expansion rate (number of substations, teleindications etc.), has a direct impact on the capacity and the performance of the system to be developed,

- The application functions to be developed depend directly on its operational characteristics as well as its standard ones.

2.3.2 The Telecontrol System

The telecontrol system represents the technical environment in which the project has to be integrated and gives a set of constraints to be respected.

The telecontrol system is itself divided into different subsystems:

- the control centres
- the different Remote Terminal Units (RTUs)
- the data transmission subsystem
- the data base management subsystem
- off–line and on–line functions
- the follow–up management subsystem.

Different organizations may contribute to the whole telecontrol system, very often for historical reasons and also because of the interactions of various departments in the user's organization with the present system.

The principal requirements to be respected are the following:

- data exchange interfaces with the Energy Management System (EMS) functions

- data transmission interfaces, especially regarding protocols, data formats and transmission speeds and flows

- general rules for the Man–Machine Interface (MMI)

- data base configuration rules, especially regarding the consistency of the data. For example the telecontrol system may be based on a unique data base, so that each subsystem has to respect data files formats and protocol rules for the extraction of the utilized datas.

This list is of course not all inclusive, and many principles, guidelines and interfaces may have to be respected. For example, automatic data downloading, testing procedures to set a new substation in operation, hardware and software autocontrols, defaults supervision performance follow–up etc.

Finally, another very important field of requirements, that has to be defined is all the physical environmental conditions such as electric power, electromagnetic constraints and temperature conditions.

This last specification area may have heavy consequence, for example the obligation to add a metallic protection structure for each graphic screen to be installed in a substation.

2.3.3 The Personnel Organization

The third general area to be taken into account in the user environment is, of course, the personnel organization.

Sometimes, the arrival of a new system is the occasion to change the personnel organization, and that brings more factors into the development.

But this case is very unlikely. In most cases, it is absolutely necessary to respect the human environment and the staff abilities, to give the best chance for the project to succeed.

For example, it is necessary to examine the methods used for collecting the different required data when setting in operation a new device (circuit-breaker) in a substation or a new power generation unit. The same goes for the different operational procedures engaged in before this new part of the energy system is placed in operation.

Another example in defining the staff training means, is to examine the real possibility to detach a certain time from the project implementation for educating the staff.

On the other hand, it is clear that the new system has more chance to be well accepted, because:

- it does not disturb the operations during its installation
- it brings additional convenance to the end user, and also to each person involved in the installation, operation or maintenance of the system (this one is also a kind of end user).

With all the technical advances, this last point will be more and more the real key of the success of every project.

2.4 Definition Of Project Targets And Limitations

The object of this chapter is to obtain the formal result and the requirements list while taking into account the recommendations of the previous chapters.

2.4.1 Functional Requirements

The point to underline in summary is that

- this specification must be complete from the end user point of view
- it has to be limited to "*What*" and "*Why*", without specifying "*How*".

2.4.2 Technical Requirements

This specification has to describe all the environmental requirements and interfaces to be respected.

It is also the occasion to add requirements regarding the "*how*" if the customer wants to impose some technical and politic choices, for example regarding the use of standards such as the ISO model, graphic interfaces (e.g. GKS, X-OPEN, MOTIF), data base management systems, the use of open system products (UNIX, TCP/IP etc.) or the use of a particular range of computer.

2.4.3 Quality Requirements

In addition to the general specification of the required quality level, the customer has to define his political and his technical choices regarding the quality organization.

Two points have to be underlined here:

- The object of the project is a system, so even if the major preoccupation is on the software side (which particularly is developed in this report), the quality organization and control has to deal with the hardware side (for example the hardware acceptance tests specification) and the integrated system side (good software on good hardware does not necessarily give a good resultant system). In conclusion, the project has to be based on three complementary quality plans: software, hardware and system.
- The quality requirements belong to different fields and points of view. For example, the MacCall classification defines 12 different quality factors. Such as to obtain the general quality level, it is not possible to obtain a total from the quality for each factor. In addition, there are some conflicts between different factors, so that the user has to make choices and to clearly define what quality factors are important.

2.4.4 Dimensional Parameters

This part of the specification is also very important, as the choice of the general architecture and the range of computers (size, performance etc.) will depend on it.

It is essential that all dimensional parameters are well identified and that the effect of the choice of their values has been evaluated.

These parameters are notably:

- number of interfaces to be respected, number of MMI etc.
- data transmission durations, dialog answer durations, response times
- availabilty rates
- data base sizes
- number of functions to be active simultaneously
- restarting and changeover times
- physical environment requirements.

It is also absolutely necessary to identify the vital functions of the system, and what functions of the system may be suspended in case of partial system failure.

2.4.5 Time-schedule

The user has to define requirements regarding the commissioning date of the system or regarding different phases if a part of the functions must be in operation shortly.

However, it is clear that the time-schedule cannot be definitive without taking into account the supplier point of view.

2.4.6 Financial Point Of View And Available Resources

As for the time-schedule, the financial point of view will have direct consequences on the content of the project and on the required quality level. The evaluation of the available resources is also essential for the quality control: it is useless to establish certain particular procedures and visitations if nobody has sufficiant time in the user staff to participate in them.

For these financial and resource points of view it is difficult to give general recommendations. It is essentially a question of negotiation between the involved parties: user, consultant and supplier.

2.5 Check List For Project Goals And Requirements

* End User Requirements
  - What has been done to ensure that the user's points of view have been respected?
  - Are the availability requirements reasonable?

* User Environment
  - Has the supplier enough knowledge of the end user's environment?
  - Are there any description available?
  - Are technical visits made or planned?
  - Are all principal requirements specified and agreed to?
  - How will the customer's organization react to the new system?
  - Measures to make a smooth start-up and positive reactions?

* Project Targets And Limitations
  - Do the Functional Requirements tell "what" and why", and NOT "how"?
  - Are all environmental requirements and interfaces to be respected described in the Technical Requirements?
  - Are the target quality level and applicable standards specified in the Quality Requirements?
  - Where are the project limitations specified?
  - Which are the financial goals and limitations?
  - Which resources are available, and when?

3 THE PROJECT PLAN

3.1 General

A systematic approach is central to successful project planning. The planning is a difficult task, which never will be completed by itself. Therefore, the first thing to do on starting the realization phase is to make the decision to draw up a **project plan**.

The project plan forms the basic document for every project and should be drawn up in a very early stage of the project – if possible even before the decision has been taken to sign the contract.

The main objectives of good project planning are:

* to identify the critical work tasks
* to remember every task
* to perform the tasks in correct order
* to get the resources when needed
* to achieve steady personnel load
* to be up–to–date with the project situation
* to handle changes efficiently
* to actively identify and prevent potential problems.

The project plan is made in coordination between the project manager, the key resources and the responsible organization managers.

When the final project plan is ready, it should be presented to and approved by all relevant parties to applicable extent. This means supplier and customer, line and matrix organization, project team resources, and the responsible management.

3.2 Project Plan Contents List

The project plan is a simulation of the project implementation. If it does not work on paper, it will not be successful in reality.

The main items in the project plan are the same in almost all projects, although the technical solutions might differ. The following list gives an example of the typical contents in a project plan:

* Project Definitions
  - Background and general information
  - Targets
  - Limitations and interfaces
  - Risks and potential problems

* Organization
  - Project team
  - Project supervisors
  - Contact persons
  - Task lists

* Implementation plan
  - Time-schedules
  - Plans for each project stage

* Financial plan
  - Cost estimates
  - Project budget
  - Payment schedule
  - Cost supervision

* Control Plan
  - Meeting plan
  - Information plan
  - Reporting plan
  - Test plan
  - Documentation plan.

3.3 Project Definitions

This part gives background information and describes the starting point and the reasons, why the project is started. Investigations made earlier and the profit analysis may also be included here. The interfaces to the surrounding "world" and limitations are described. It also outlines the external factors and organizations that are connected to the project. It is also sometimes important to define which parts are excluded from the project.

The starting point is formed by the general targets, set by the management or the project leading group. The targets can be divided into three groups:

* technical and performance targets
* time-schedule targets
* financial targets.

The **technical targets** will be more detailed when the project proceeds. It is, however, important to define the main technical aspects as precise as possible already in the project plan. If several technical alternatives are known, they can be specified including all known corresponding consequences they will cause. This serves as a useful help when the final solution will be chosen. The technical targets also defines the requested quality aspects, wich are further defined in a **Quality Plan** (see Chapter 7.3).

The functional specifications should be as precise as possible regarding the targets, overall functions, quality and performance requirements, and limitations. It is, however, important to leave space for the designers to find efficient and creative solutions. The specification should focus on what the user wants to get out of the system, and not too much on technical details.

The **time targets** should be defined very exactly. In case of an external delivery project, the main milestones are normally included in the contract. Milestones in internal projects are usually defined by the company management.

It is essential that the time targets are realistic. Before the targets are set you have to pay attention to the way of implementation, the available resources etc. The time targets should not be allowed to influence the estimation of required amount of work or durations. Firstly, a probable time-schedule is drawn up. If the completion target is earlier than the estimated, the possibilities of shortening the time-schedule should be checked. Every decision of changing the schedule should be based on realistic considerations.

The **financial targets** are based on the project budget, which in turn is based upon time-schedules, work estimates, hardware procurements and cost level development estimates.

All project implementations are more or less subject to **risks** of different kinds. It is important to identify and take into account as many potential problems as possible. The risk areas and concrete difficult tasks should be discussed – or at least listed – in the project plan. This enables the project teams to prepare the design and implementation in a way to minimize the risks and eliminate the problems.

3.4 Project Organization

This part of the project plan should describe the project team structure, the persons and the main task division and responsibilities.

It is always important to the project and the project manager, that there is a high-level **Project Management Group** controlling and backing up the project implementation. The supervisors represent the top management and should hold regular meetings, where the project manager is present, reporting and discussing the status of the project.

Alternative project team structures are discussed in Chapter 4.3.

To make the project work as efficient as possible is it useful to list the contact persons on customer and supplier sides, specified separately for each functional and responsibility area. Typical items are:

* Technical matters
  - general software
  - data base
  - data processing
  - presentation and reporting functions
  - general hardware
  - computers
  - man-machine interface
  - remote terminal units

* Contractual matters
  - technical changes
  - time-schedule changes

* Project implementation matters
  - training
  - documentation
  - installation
  - maintenance
  - quality.

The contact person list should include all necessary information, such as names, addresses, telephone and telefax numbers etc.

Formal procedures may require that all communications be directed to the user and supplier project managers for appropriate project control.

3.5 Implementation Plan

The implementation section of the plan deals with the implementation of the project, based on the background facts and the targets. All phases of the project should be described to enable the detailed, practical arrangements in due time. An important part of the implementation plan is to define the **progress identification milestones** (see Chapter 5.3.5).

The work tasks are described as detailed as possible, including responsibility areas and limitations.

The resource planning and the time planning go hand in hand. Regardless how detailed the plan is, the time-schedule is worthless unless there are resources to perform the tasks. Nevertheless, you often find situations where the project planning is limited to checking the time-schedule, without any guarantees that the resources will be found and are available according to the needs. The result will be rush, overtime work and delays.

The importance of the resource planning will be emphasized when the main part of the personnel is formed by key resources and experts, who may not be available on short notice.

The targets of the resource planning are:

* to secure the availability of the resources in the time-schedule and hereby the total timing

* to optimize the use of the key resources, i.e. continuous and stable work load

* to reduce resource costs.

Resource management is discussed in detail in Chapter 4.

3.5.1 Tasks And Responsibilities

It is essential to define the work and cost division between the supplier and the customer in detail already in the delivery contract. The project plan should present the contractual items in a form, serving the needs of the designers. Clearly defined tasks, responsibilities and dead-lines eliminate misunderstandings during the implementation and guarantees that tasks and needed information are available at the correct time.

A typical task and responsibility list could have the following layout (CUS = customer, SUP = supplier):

POS.	*T A S K*	*RESPONSIBLE*	*DATE*
1	*Final list of data base size for power system information and calculations*	*CUS*	*94-09-10*
2	*Final list of power system information in each substation*	*CUS*	*94-10-01*
3	*System specification draft*	*SUP*	*94-12-01*
4	*Comments on system specification draft*	*CUS*	*95-01-15*
5	*Final system specification*	*SUP*	*95-02-10*
6	*Approval of system specification*	*CUS+SUP*	*95-02-20*

3.5.2 Training Plan

The project implementation and the system operation always require training, both for the supplier and the customer. To be able to draw up the training plan the following questions should be discussed:

* what kind of training is required?
* which and how large are the target groups?
* what are the trained persons expected to learn?
* when is the knowledge required?
* where are the training courses arranged?
* who is arranging them?
* which is the cost division?

3.5.3 Installation Plan

The installation plan defines information about how the system should be installed on site. The plan describes installation of both software and hardware, taking into account possible existing systems in use.

The plan also specifies the responsibilities of both parties during the installation.

3.5.4 Commissioning Plan

The commissioning plan describes how the new system will be taken into operational use. Important issues are e.g. how to arrange a flexible change from an old system to the new without cutouts.

3.5.5 Maintenance Plan

The normal lifetime of a telecontrol system is 10...15 years. It is expected that the the power system will change and expand continuously during the lifetime. To eliminate or at least minimize operational breaks during modifications all maintenance aspects should be defined in coordination between the supplier and the customer.

Items to be discussed in the maintenance plan are e.g.

* fault situations
  - maintenance contract (software backup)
  - overall responsibility
  - hardware/software responsibility
  - response times for correcting activities

* software modifications and upgrades
  - version management rules
  - implementation of new releases
  - location of source software

* maintenance tools and routines
  - maintenance workplace
  - software tools
  - testing equipment needed for verification of functions
  - remote maintenance and fault analysis using modem

* regular maintenance
  - software backups
  - statistic error logs.

Parts of the maintenance plan can be drawn up during the project implementation, but it is important that the main parts are defined in the beginning of the project, thus enabling the final user to arrange the maintenance organization and facilities, and train the personnel.

3.6 Financial Plan

Financial planning secures the profitableness of the project concerning investment costs, operational costs and profits. By analyzing the connections between task timing and derived costs the time-schedule can be optimized and thereby be made more profitable. The project budget forms the basis for the cost supervision.

3.7 Control Plan

The control plan specifies procedures to be used in the management and implementation of the project.

3.7.1 Meeting Plan

Efficient handling of a project usually requires regular review meetings between the parties. To enable long term planning of activities preliminary dates should be listed in the time-schedule.

Internal meetings should be arranged both regularly and when specific reasons occur, both for technical discussions and work progress follow-up.

All meetings should produce a short **report** with the main items discussed and all decisions taken. The distribution should cover all participants, the project management group, associated line management and persons mentioned in the report. The reports form part of the project's quality records.

3.7.2 Information Plan

To enable an effective implementation of the project, the recognition of possible problems and risks, and to keep up a proper motivation level it is important that all team members are up-to-date on the status of the project. This is achieved by providing regular information to the project team and management, both as published reports and with meetings. **The software team should also get information about hardware implementation and changes, because they may be of vital importance for the software implementation.**

A well informed team member is better equipped and motivated to contribute to the mutual target – a successful project completion.

3.7.3 Reporting Plan

The supervison of the project progress is mainly performed by regular discussions with the team members. However, to eliminate misunderstandings and to create data for post-delivery analysis, the work progress should be documented by every team member in form of internal progress reports.

To avoid long novels with diffuse messages the progress reports should be drawn up in a standardized and precise form, defined in the project plan.

Progress reporting between the customer and the supplier is based upon the results from the internal project reports. The supplier should prepare the report at least one week before the project follow-up meetings with the customer.

Progress follow-up is discussed in Chapter 5.

3.7.4 Control Of Changes

The first months of the project are normally reserved for completing the functional specification of all hardware and software. During this time it is relatively easy to adjust the extent and contents of the originally planned project. But when the system specification has been written and approved by both parties the contents should be fixed when the implementation of the software starts.

After this moment is it essential that the project team is protected from all unnecessary changes and disturbances. This can be achieved by using a system of handling changes, called the **Engineering Change Control Procedure (ECCP).** The general idea of this procedure is to enable flexible changes in the project without causing disturbances and risks for the project implementation.

The **Engineering Change Control Procedure** follows the following rules:

* Both the customer and the supplier have the right to request a change. The change needed is specified in a document called **Engineering Change Request (ECR).**

* The supplier studies the possibilities of including the change in the project. A suggested implementation is documented in the form of an **Engineering Change Proposal (ECP)**, which describes all consequences concerning performance, time-schedules, costs and other contractual items.

* When the ECP has been studied, the customer writes an **Engineering Change Notice (ECN)** to verify the acceptance (order) or refusal of the change.

If the requested ECP should require extensive preparation work, the supplier firstly should offer the ECP work to the customer. This is important to prevent frequent ECRs without any orders from occupying the supplier unnecessarily.

The stages of the ECR–ECP–ECN method are the same as the normal enquiry–proposal–order procedure, but the implementation is designed to avoid any unnecessary bureaucracy and paper work. The length of an ECR and an ECP should normally be only one page. Additionally, the ECP can be equipped with a line for the customer's approval or refusal, thus serving also as an ECN (see example in **Figure 3.7.4**).

It is useful to define in the delivery contract how changes should be handled during the implementation.

Figure 3.7.4: Combined ECP And ECN Form

ENGINEERING CHANGE PROPOSAL No ECP-123	**Issued by** James R. Bug
Distribution United Voltage & Current Inc. SCADA Software Consulting plc.	**Date** 20.03.1994

Description New Energy Display Function

The present SCADA system software will be completed with a new function for the Man-Machine Interface subsystem.

The function will include presentation on screen and printer. The display is activated using existing menu functions.

The layout will be as follows:

Time	Station 1	Station 2	Station 3	Station 4	Energy Sum
07	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxxx
08	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxxx
09	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxxx
10	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxxx
	(etc.)				

The delivery includes software development, installation on site, standard documentation, according to main contract XYZZ.

Proposal | **Valid thru** 20.04.1994

Delivery time: 3 months from order

Costs: ECU 12.345 excl. VAT

Payment: 100 % on delivery

Affects

- time-schedule ( )
- contract ( )
- paragraph ( )
- plans ( )
- costs (x)

Enclosures

Special conditions

ENGINEERING CHANGE APPROVAL

SCADA Software Consulting plc.

____________ Signature   20.03.94 Date

United Voltage & Current Inc.

( ) approved ( ) not approved   ____________ Signature   . .94 Date

3.7.5 Test Plan

The testing of complex software systems demands detailed plans of how and when the different tests should be performed. The test plan should test the functions listed in the functional specification. It should define:

* which entities should be tested
* to which extent the system should be tested
* where the tests should be arranged.

Additionally the test plan defines how and when the detailed test specifications should be written, which (internal or external) testing standard should be followed etc. Unless suitable standards are available should examples and test forms be included in the plan.

A correct timing of the testing phases is essential from several points of view:

* tests identify possible problems, that can be avoided easier in an early stage of the project
* without tests it is often impossible to estimate the progress of the software development.

A complete Factory Acceptance Test (FAT) is very important; all experts are available on short notice and in-house testing facilities can be used. The testing on site should be reduced to a minimum, because it tends to be extremely expensive, both to conduct and to make changes, if necessary. The Site Acceptance Tests (SAT) should be to duplicate the Factory Acceptance Tests to applicable parts.

Test and validation aspects are discussed in Chapter 6.

3.7.6 Documentation Plan

The project is normally documented in accordance with the supplier's internal standard, which - hopefully - is based upon international documentation standards. The delivery documentation is defined in the delivery contract or is specified in the beginning of the project.

To be able to keep the project documents in order it is essential to specify **which** documents should be prepared and **when, how** the distribution should be made and **where** the documentation should be kept and registered.

To simplify the distribution and the archivation, all documents should be given a unique identification code.

Other details to be specified in the documentation plan are:

* number of documents for each purpose
* detail level
* documentation media
* documentation language
* preparation tools (word processing or CAD type).

The documentation plan should also define the structure of the specifications to be written during the project.

3.8 Check List For The Project Plan

* Project Definitions
  - Are the project background and the implementation environment defined?
  - Which are the technical targets of the project?
  - Are the time-schedule targets defined for each milestone?
  - Which are the consequences if the project should be late?
  - Where are the financial targets specified? Are there any rewards if the project team should succeed better than expected?
  - What happens if the cost budget collapses?
  - Where are the project limitations described? What is included in the project and what is not?
  - Are the interfaces to the surrounding "world" described?
  - Any risks in sight? Or potential problems?

* Project Organization
  - Is the project team defined in detail?
  - Do the project supervisors have enough "power" to handle possible delicate problems the project team cannot handle?
  - Are contact persons defined for technical and contractual matters? Both for customer, supplier and consultant?

* Implementation plan
  - Are the time-schedules realistic?
  - Are there plans for each project stage?

* Work Breakdown And Responsibilities
  - Are the task lists detailed enough?
  - Which task has the longest duration? Should it be divided into separate subtasks?
  - Has every task a responsible person?
  - Where are the task deadlines defined?

* Training Plan
  - Which are the required training areas for the people involved?
  - How are the target persons or groups defined?
  - What are the expectations of the training result?
  - Where, how and when is the training arranged?
  - How is the training financed?

* Commissioning Plan
  - When should the commissioning time-schedule and arrangements finally be planned in detail to ensure a smooth start-up of the new software system?

* Maintenance Plan
  - Which are the responsibilities during the guarantee time, and after that?
  - How are fault situations handled? Are software, hardware and system contact persons defined?
  - Are there specific rules for software modifications and upgrades to ensure, that correct software versions are installed in all computer systems?
  - Are there maintenance tools and routines available?
  - Where are the regular maintenance routines described?
  - Where are software backups stored?
  - Are error logs saved and delivered to the supplier for necessary measures?

* Financial plan
  - Are the cost estimates based on adequate information?
  - Is the project budget realistic?
  - Is there a worst case budget, including all possible risks?
  - Is the payment schedule planned to minimize interest losses?
  - Are there routines and tools for supervision of costs?

* Control Plan
  - Does the meeting plan describe how often and where project review meetings should be held?
  - Will the meetings produce any documents?
  - How is information distributed to the project team members?
  - How are the individual project team members describing their working progress?
  - How often are progress reports written?
  - Are there any methods for handling changes in the project?
  - Who approves changes?
  - In which stages of the project are tests performed?
  - What are the expected results of the tests?
  - What happens if a test should be rejected?
  - Who is responsible for the test subproject?
  - Where are the documents, produced by the project, specified?
  - Are there dead-lines for each type of documents?
  - How are the documents coded and where are the original documents kept?

4 PROJECT AND RESOURCE MANAGEMENT

4.1 Introduction

Prior to developing some guidelines for the staffing and management of a software project, let us briefly put the objectives of what we are trying to achieve in perspective. A project in general is defined as:

> *A set of time related, usually non-repetitive activities needed to meet a defined, time limited and cost limited objective.*

All phases of the project should be defined to be result oriented so that the progress and the performance can be measured. Project milestones should be based on key deliverables rather than on the completion of some activities.

The software project should provide for:

* The delivery of the required functionality on time and on budget.
* The delivery of user training and user acceptance.
* The delivery of effective facilities and structure for operational, maintenance and support of the function.
* The delivery of the facilities and methodology for the future evolution of the system.

A narrower view of a software product will greatly shorten the life of the software package and provide poor return on investment.

The organization of the project teams is influenced by many factors. Amongst the many important factors are the following:

a. **Technical factors:**

The type of work to be performed: the expertise, the equipment and the supporting structure available in-house.

b. **Organizational factors:**

The in-house staff resource available, the development plan for the project, the organizational structure, the implementations philosophy for the project, etc..

c. **Financial factors:**

Budget for the project, cost of additional equipment and support facilities required.

The percentage of software project failures is very high, and the most major cause, apart from poor planning and project definition is the inappropriate project resourcing and poor project management. The most common staffing error is to nominate the top system analysis or programmer as the project leader. These are technologists who are enthusiastic and energetic but who inevitably get involved in the technical details, forgetting the big picture, consequently leading the project into grossly time and cost over run.

There are no golden rules for resourcing, structuring and managing the project team. What can be used, however, are some useful guidelines for the methodical analysis of the work at hand and the appropriate selection of resources for the various tasks within the technical, organizational and financial constraints that will provide a better chance to deliver the required product of the project.

4.2 Work Definition and Work Breakdown

All project work can be broadly classified into two categories: management and technical work.

4.2.1 Management Work

Management in general involves:

a. **Planning activity:**

* Breakdown of the project with definition of the deliverable of each task.
* Structure of the project team and definition of the areas of responsibility.
* Definition of the interface requirement between the team and the environment.
* Project Plan with milestones and deliverables.
* Staff loading plan.
* Cumulative cost estimate for each phase of the project with cost breakdown.
* Marketing plan for the project.

b. **Organizing activity:**

* Obtain from the organization the formal commitment of the required staff and their time commitment.

* Definition of channels of contact and the communications channels with the interacting groups.

* Organizing progress reports, project meetings and other facilities as required during the course of the project.

* Generally organizing the work of the team and of external personnel.

c. **Leading activity:**

* Marketing the project to team members and to the organization.

* Providing staff counselling and motivation.

d. **Controlling activity:**

* Establish regular reviews of progress to facilitate the early decision of issues and problems.

* Control of activities and deliverables against the Project Plan.

* Applied structured methodology to the control of changes (design freeze dates, change cost estimation method, change review and approval criteria, etc.).

4.2.2 The Technical Work

The technical work can be further subdivided into non-software tasks and software tasks.

4.2.2.1 Non-software Tasks

a. Commercial and contractual arrangement and liaison with vendors, contractors and consultants.

b. Specification or arrangement for the supply of the appropriate computing hardware and communication facilities for the project.

c. Specification or arrangement for the preparation of the target site, user design participation, operational, maintenance and support training.

d. Quality Control for the project.

Some of these tasks may or may not be done normally by other sections or persons within the organization. In any case the resources required whether jointly used with other sections or projects within the organization should be clearly identified and allocated to this project.

4.2.2.2 The Software Work

The software work includes all tasks related to the design, documenting, coding, testing and integrating of the software. The work can be divided into the following:

a. System Design

System Design includes the design of both the software and the accommodating hardware, communications platform. The design process generally results in a number of design specification documents which are to be produced and/or revised. Some or all of the following documents may be required:

* System functional design document: this is the synthesis of the user requirements and the basis of the specification if the software is to be purchased off-the-shelves or to be produced by an outside organization.

* System design overview document: this document is the product of the structured analysis of the functional requirements.

* Detailed design document: this document provides the breakdown of the generic design into submodules with clear definition of the interfaces between the modules.

* Translation of the software detailed design into some form of program description language.

* Translation of the Hardware/Communications detailed design into connection diagrams and equipment lists.

b. System development

The system development includes:

* Software submodules coding and testing; hardware and communications facilities installation and testing.

* System integration on test floor and at site.

c. **System testing**

Production of test procedures which includes test hardware set-up; perform testing, following written procedures for the testing and the recording of results.

d. **System commissioning**

Production of commissioning procedures and the performance of the commissioning function, including the resolution of any faults or difficulties at site.

e. **On-going support**

Procedures are required for the on-going support of the system.

The breakdown of tasks described in this subsection assists in identifying the various expertise and skills required and helps to make a rough estimate of the man-hours required for each task. This rough estimate will need to be refined when the team is in place in order to provide a more accurate resource plan for the project.

4.3 The Project Team Structure

The process of task breakdown, defines the type of expertise and the resource in terms of man-hours required for each type of work. The natural follow-up action would be to select the personnel for these tasks. The selection of these personnel depends very much on the structure we intend to adopt for the project team.

The team structure will depend to a large extent on the organizational structure of the corporation, the accepted method of forming project teams and the subsequent direction of development intended for the team members.

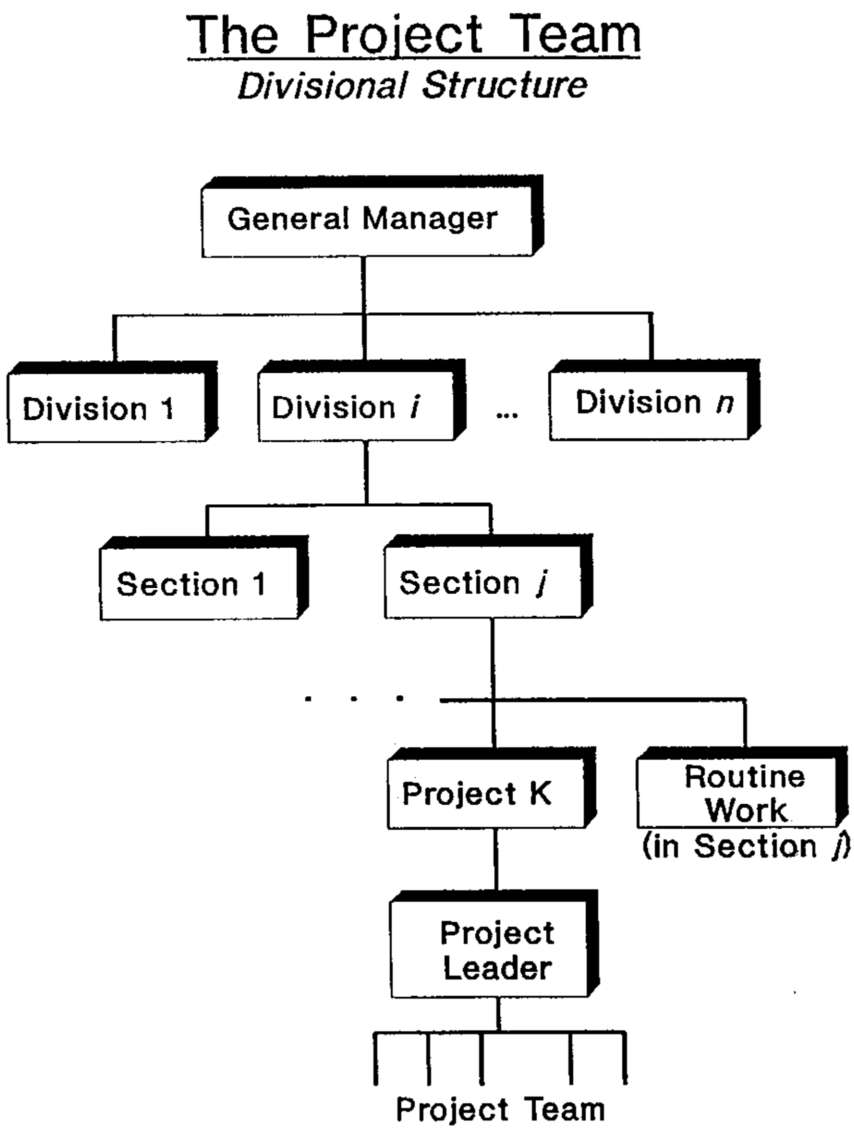
The two basic structures from which a number of variations can be adopted are: Divisional and Matrix structures.

4.3.1 The Divisional Structure

This structure stems from a corporate structure that is divided into divisions according to technical disciplines.

A typical project team within a divisional organization structure is shown in **Figure 4.3.1**.

Figure 4.3.1: The Project Team / Divisional Structure



4.3.2 The Matrix Structure

To enable this structure, apart from the core project members, the other members of the project are normally not on the project full-time. These are experts in their disciplines who may be handling a number of different projects simultaneously.

A typical project team within a matrix organization structure is shown in **Figure 4.3.2.**

The advantages and disadvantages of these structures are largely explained in most of the organization manuals.

4.3.3 Towards a Preferred Structure

Each of the two structures considered above has advantages and draw-backs. A suitable structure for a software project depends on the culture of the organization and lies somewhere in between these two extremes.

A preferred structure for the implementation of power system control software is to select a group of core staff with strong application domain knowledge to be in charge of the key responsibility of the work. This core staff should come from the same section or division to avoid the issue of divided loyalty. This core staff should assume the top line management and design role. They are essential to translate accurately the users requirements into the system functional and system module design. They should form the necessary bridge between the (power) system information and control needs and the enabling digital technology.

This core staff should consist of the project manager and a deputy manager to assume all the project management tasks. The other skills required are planning, organizing, leadership and personnel motivation skills.

It is not essential that the project manager is an expert in the technology employed in the project, rather a clear understanding of and a commitment to user requirements is necessary. The deputy manager assists the project manager in the project management task. The deputy's skills and expertise should complement those of the project manager, especially in the technical fields.

For the remainder of the core team, there should be one key team member to coordinate as well as perform some of the non-software tasks. There should be two key team members to look after the top level system design and testing, down to the functional module level. There should be one key team member to look after the site coordination work. There should be part-time involvement of user and maintainer representatives on the team for the duration of the project.

A suggested structure for the project team is shown in **Figure 4.3.3.** The technical management from quality point of view is discussed in Chapter 7.6.2.

Figure 4.3.2: The Project Team / Matrix Structure

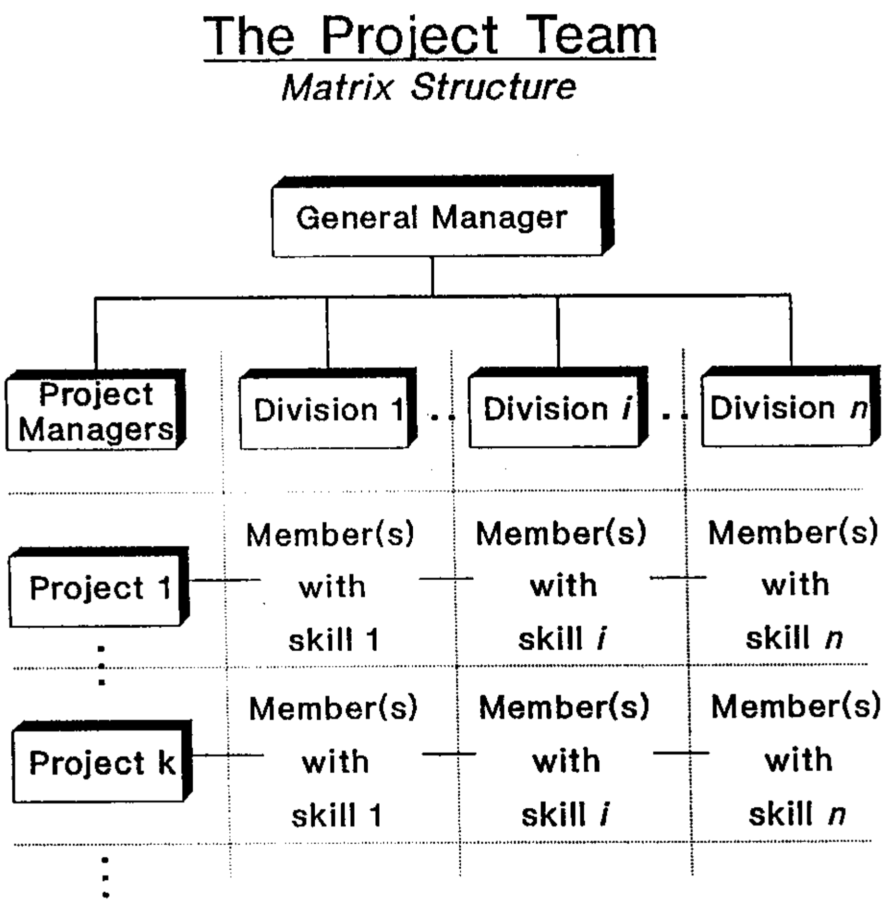
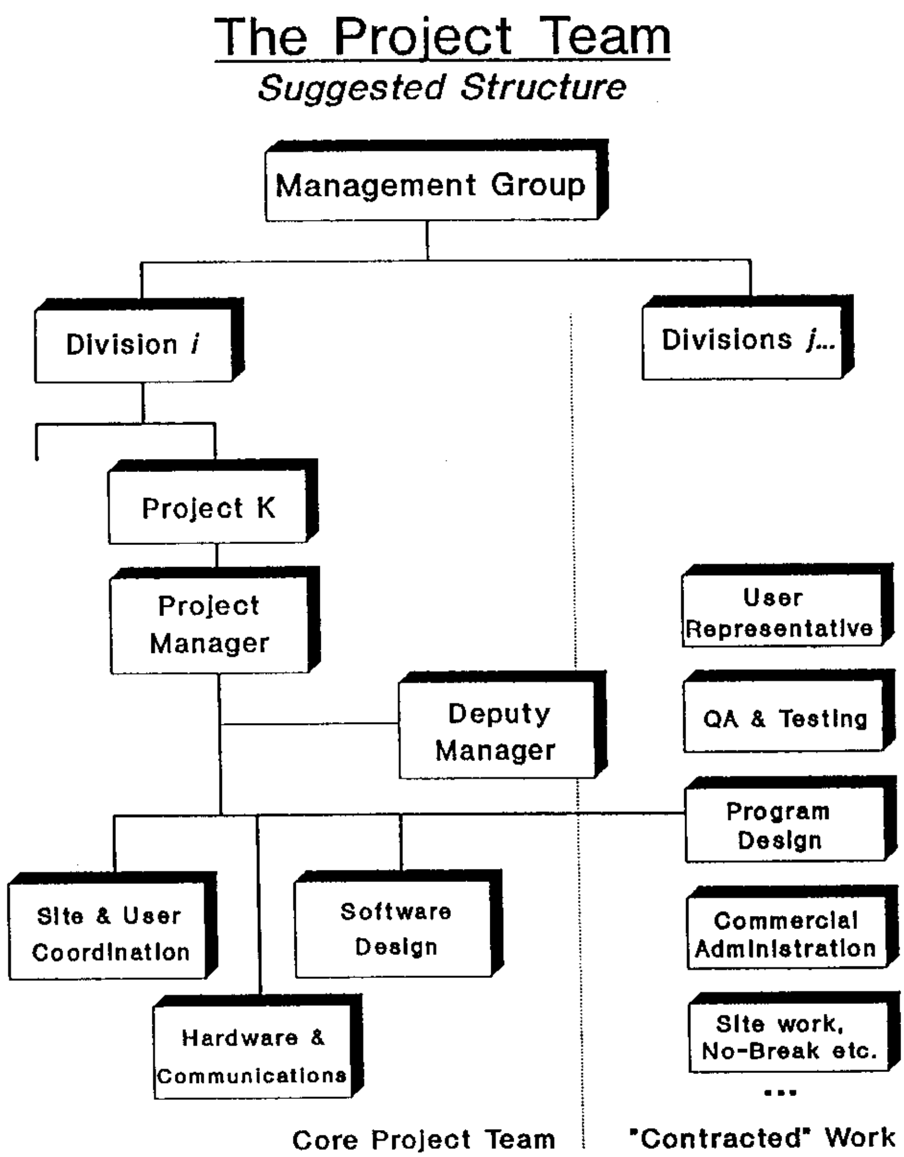


Figure 4.3.3: The Project Team / Suggested Structure



4.4 Customer–Supplier Relationship

Nowadays, with the availability of field proven and reasonably priced software packages for SCADA and EMS on the market, it is generally not efficient or cost effective to develop the software totally in–house. This way the task of the project team is either to specify and procure the software packages with in–house implementation or to specify a turn–key project with the supplier in charge of implementing the system.

In the first instance it is to be kept in mind that the system integration task is complex and the resources and time required are often underestimated in the implementation plan. Most vendor claims of compatability either with existing operating system, software and hardware/communications need to be verified and tested in detail. It is also to be kept in mind that suppliers up–date their products regularly and any changes to the software made by the customer may have to be modified to adapt to the new software version from the suppliers.

In the case of the turn–key contract, the bulk of the system design and implementation is to be performed by the supplier. The project manager is the sole point of contractual and formal contact between the customer and the supplier. On the technical level, there is generally a peer to peer relationship between the two sides. The core project team structure should be maintained with the reduction of the deputy project manager.

In cases where extensive software development is to be performed by the supplier, it is productive to concentrate on an effective progress monitoring and reporting protocol rather than to insist on a particular team structure from the supplier.

Depending on the need for staff development, arrangements can be made with the supplier to have the customer team members to join the supplier's team for some significant periods to perform specific design tasks. On the other hand, some suppliers may propose that one or more of their team members joint the customer's team to collect user requirements, system data, etc. These arrangements, if properly organized and coordinated could be beneficial to the implementation of the project and to the secondary issue of staff motivation/development.

4.5 Contractors And Consultants

When the resource required for the project can not be provided in–house and when it is not convenient to have the work done by the supplier, generally contractors/consultants are engaged to perform the work.

A consultant is engaged to supplement the expertise not available in–house.

4.6 Some Project Management Issues

The responsibility for the success of the project finally falls on the project manager. It is not enough to manage efficiently, for staff can be doing the wrong thing very well, very efficiently. It is to be kept in mind that the objective of the project is a long term product which includes not only the delivery of the required functionalities but also the long term survivability of the system in terms of user acceptance and system supportability. This subsection raises a number of project management issues that the project manager must pay specific attention to in order to enhance the chance of project success.

4.6.1 Time

Time, or the lack of time, is often the most major problem for project managers. This is because "time" is only the manifestation of a deeper problem, that of failure to understand the importance of the management process and its relation to technical work.

Most software project managers come from a technical background where solutions are found in a precise and systematic approach. This has led to the failure to grasp the nature of management work. It is easy to get embroiled in technical problems and solutions while neglecting management tasks.

It is therefore imperative for the smooth running of the project that the project manager possesses the ability to allocate the necessary effort in the management tasks while taking a step back from the day to day resolution of technical issues.

4.6.2 Project Definition

One of the direct consequences of the lack of planning is the resulting poor definition and breakdown of the project tasks, causing confusion amongst the team members and ineffective project performance.

Thus, the project stages need to be thought out carefully, and the tasks properly and comprehensively broken down to the required components. Only then will the project proceed with a more than even chance of success.

4.6.3 Lateral Thinking

The project managers, due to time constraints and other factors, often find themselves reacting to apparent problems rather than concentrating on the prevention of problems.

The objectives of a project, especially a long term project are never set in concrete. As the organization changes and evolves, as the technology advances, so will the objectives of a project change with time. The project manager must review the objectives, such as the user requirements, at predetermined stages, and use lateral thinking to make the correct trade-off for the overall success of the project.

4.6.4 User/Maintainer Involvement

The understanding of the user needs and the planning for future system support are crucial to the sucess of the project and it is the responsibility of the project manager to bridge the gap existing between the user/maintainer and his project team. The user/maintainer should be involved from the inception of the project. The project should be for the user and should be sponsored by the user.

The system once installed should be easy to maintain and support. A user representative should participate in relevant project activities, especially in the collection of user requirements and the definition of the Man–Machine Interface (MMI). A maintainer's representative should also be involved, especially in training, documentation review and system testing activities of the project. Only by the close involvement of the user/maintainer in various phases of the project can there be a guarantee on the success of the project outcome.

4.6.5 Staff Supervision and Motivation

The failure of many projects has been attributed to staff problems. The problems are often identified as lack of "motivation" or lack of "communications".

The project manager needs to be vigilant in order to identify early sign of problems within the project team in order to provide the appropriate inputs to remove/alleviate the problems.

a. The real cause of the problem needs to be identified.

b. Working conditions and terms of services are important and the manager needs to ensure that these are fair to all team members involved.

c. Once the organizational factors are under control, the project manager has within his authority, the opportunity to significantly affect the way the project team function and thereby provide motivation to the team members.

   All these activities require personal skills from the project manager in terms of communicating, establishing objectives, understanding relationships, knowing how to and when to delegate, etc. In general, the project manager must be sincere in his approach as the correct approach to each activity is a motivator in itself.

4.7 Conclusion

This section provides some guidance in project and resource management for software project implementations. The task of resourcing and structuring an appropriate project team is complex and there are no easy solutions. The components and structure of the team depend to a large extent on the structure and the culture of the organization.

Within these organizational constraints, the team manager must select his/her team, give it an effective structure and perform the normal planning, organising, leading, controlling and motivating activities to ensure the project is delivered on time and on budget.

An appropriate starting point is the project analysis and task breakdown to assess the required expertise and resources required to implement the project. With a check list of expertise extracted from the task breakdown and a rough estimate of the manhours required for each task, the project manager can start to select and negociate for the inclusion, co-opting, secondment or employment of the required team members. The structure of the team will be a hybrid between a divisional structure and a matrix structure. This selection of the team and the structure is made complicated by the added dimension of the possible involvement of an external supplier of software.

Finally, a selected number of project management issues are raised, including time management, planning requirements, user/maintainer involvement, lateral thinking and staff motivation. All these issues need to be addressed by the project manager to ensure the smooth and successful implementation of the project.

4.8 Check List For Project And Resource Management

* General Work Management
  - Where is it defined what is expected when a task is ready?
  - Which are the expected documents?
  - Where are the channels for contact and communication defined?
  - How is the project marketed internally – and externally?

5 PROGRESS IDENTIFICATION AND FOLLOW-UP

5.1 Introduction

Historically software development projects have had an exceptionally high probability of being late, often more than double the originally estimated time. The authors are aware of at least two such projects where the actual project time was more than four times the delivery period quoted by the supplier.

In order to be better able to identify true progress it is essential to have a good understanding of the causes of delays. This chapter first examines the major causes of delays on software development projects as determined by analyzing numerous such cases and then proposes means of identifying real progress and facilitating timeous delivery.

5.2 Causes Of Delay

5.2.1 The Art of Estimating

There is no scientific, deterministic way of reading through a user requirements specification defining user's requirements and determining the software development effort required to produce a system that will satisfy the specification.

The person responsible for estimating effort has to perform a top level software design, where the system to satisfy the functional specification is identified as comprising a number of intercommunicating subsystems, each of which performs defined functions.

Estimators then rely on two techniques. First, they make use of past experience in implementing a similar subsystem, and then use subjective judgement to estimate the effort to implement the subsystem under consideration.

Second, they refer a subsystem, which has not previously been implemented to a software designer for estimating. Historically one can say that for some reason the response provided by the software designer is inevitably optimistic. There appears to be four elements present in this optimism:

a. An intrinsic sense of optimism in the mind of the designer.

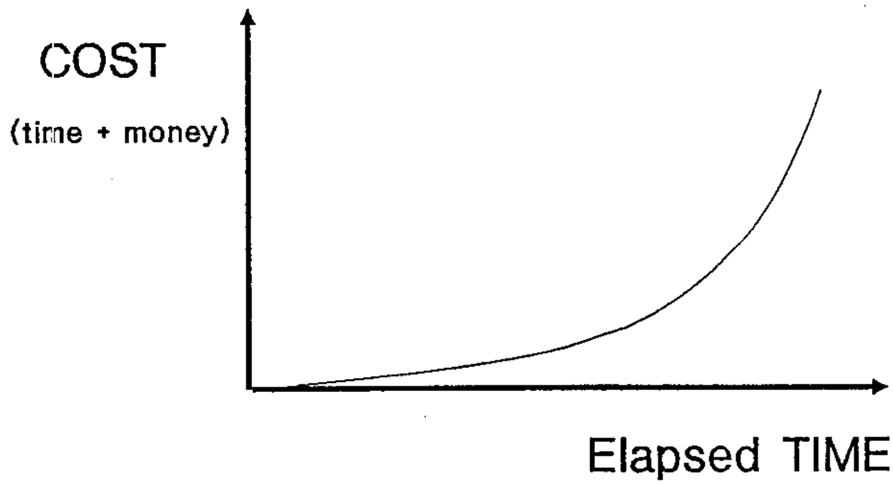
b. A total neglect and underestimation of the effort to take a working subsystem and integrate and test it in the total system.

c. The human need to give an answer, particularly to one's superior, which is acceptable. This means that it must be within the limits expected by the superior.

d. A misunderstanding of the function of the required subsystem.

5.2.2 An Inadequate User Requirements Specification

The ultimate purpose of the software system is to satisfy the user's needs. Through an often convoluted route, the user's needs are communicated to the system implementors by means of a User Requirements Specification. This specification must be recognized as an imperfect document, which will always have elements of ambiguities, errors and omissions. Experience has shown that the cost in terms of time and money of correcting a functional error increases exponentially as a function of the elapsed time on the project before the error is determined. This is shown graphically in **Figure 5.2.2** below.



The determination at a late stage of the project that the system does not meet user's expectations is not only a frequent cause of significant delays, but usually results in disputes between supplier and customer as to what exactly was defined in the contract.

5.2.3 Continual Technological Advances

Software is developed and runs on hardware platforms and software environments that are far from static. Major technological advances are being achieved each year. This means that developers are continually implementing systems on platforms, which are different to previous systems supplied.

The marketing of these platforms is often aggressive and misleading. The developer then has certain expectations in terms of the performance, capability and availability data of these platforms, which he uses in estimating the time and effort to produce his systems.

When his supplier then fails to deliver on time or the product does not live up to his expectations, a major delay is caused in the delivery of his system.

5.2.4 Critical Tasks

During the initial study of the User Requirements Specification particular attention is made to those functions, which will translate into time critical tasks and processor bound tasks. Certain assumptions are then made regarding ability of the proposed system design (hardware and software) to satisfy these critical requirements.

Where the development of these tasks is then spread over the full development time period, it often occurs that at a later stage in the project it is realized that the system design is inadequate to meet these stringent requirements. This typically necessitates a late change in system design with resulting delays on the project and disputes regarding responsibility.

5.2.5 Resource Limitations and Priority

Developers typically prepare estimates many months before a decision is taken as to whether or not they will be given the go–ahead to implement the system.

When the order comes to proceed with the work, they must then make optimum use of available resources. At the estimating stage for example, it could have been foreseen to make use of a particular experienced designer for the implementation of a critical task. By the time the order is received the particular designer has been assigned to another project and use must then be made of a less experienced junior person with resulting delays in the implementation.

Within a supplier's organization, which is usually handling many contracts, there is continual pressure to move the more experienced staff to the trouble spots on other projects. What this means is that those projects with a lower priority will tend to experience delays due to this migration of resources.

5.3 Contract Monitoring

Close monitoring of the project is essential in order to achieve three objectives:

a. **Functional correctness** – the ability of the system to satisfy the defined user's needs.

b. **Quality product** – particular attention is paid to software quality which is typically not ensured by traditional QA procedures.

c. **Project progress** – to ensure that the system is completed timeously or if delays do occur that these are minimized by early detection and corrective action.

In this chapter we are only concerned with the third objective, but will refer to the first two objectives where these impact on project progress.

A detailed process to identify project progress must be coupled to the development methodology used by the supplier. This chapter will identify essential elements of such a process in a generic way such that these elements can then be incorporated into a supplier/customer project implementation plan where cognisance is taken of the specific development methodology.

5.3.1 Customer–Supplier Relationship

A contract will be entered into between the customer and supplier for the provision of a system that satisfies the customer's needs. As such, normal contract and project management principles apply in the sense that there will be payment milestones for due performance and probably penalties for default or late delivery.

However, the successful completion of a software based project is critically dependent on the relationship between customer and supplier. This industry is often referred to as an immature high risk industry. This necessitates a joint commitment by both parties to ensure success.

Success to the customer means that he receives a quality product on time that satisfies his needs. Success to the supplier means that he delivers a system on time with an acceptable return on investment. Both parties need to be committed not only to ensuring their own success, but also the success of the other party as this is the best way to ensure their own success.

In this sense one needs to moderate traditional hard line project management principles and encourage the formation of partnership thinking without neglecting the formal contractual relationship.

5.3.2 The Proposal

When a proposal is received to perform a certain software project, the customer must ensure that he fully understands the basis of this proposal. Detailed discussions and cross examinations with the supplier is necessary in order to understand the full set of assumptions that lie behind the estimate. It is this understanding that will form the basis of progress monitoring during the project.

Some mature suppliers will be happy to share this information with the customer while others will consider this approach as an intrusion of their privacy. However, if the right customer–supplier relationship has been established and the mutual interdependencies for success have been recognized, then the supplier should realize the advantages to him of sharing this information with his customer.

5.3.3 The Functional Specification

As mentioned in Chapter 5.2.2 above, the incorrect definition of the user's requirements is certain to cause delay. The first phase of any project must then be the production of a mutually agreed to functional specification. This becomes the base document against which progress is measured and the system finally tested and accepted. A quality functional specification must satisfy the following requirements:

a. **Correct**: The user must ensure that there are no incorrect statements in the document. This is not a particularly difficult task.

b. **Complete**: The user must ensure that there are no omissions from the document. This is an extremely difficult task and requires concentration and cross checking by other team members.

c. **Unambiguous**: The user must test each statement to ensure that it is not open to misinterpretation.

d. **Understandable by User and Designer**: The document must be free of both user jargon which the designer does not understand, and designer jargon which the user does not understand.

e. **Maintainable**: While both parties should aim to avoid on-going changes to the functional specification, some changes may be unavoidable and where these are necessary it must be possible to update the document while retaining the unambiguous nature.

5.3.4 High Risk Areas

After approval of the functional specification the customer and supplier should sit together and identify the perceived high risk elements of the project. This will relate to new hardware platforms, software environments (see Chapter 5.2.3) and critical tasks (see Chapter 5.2.4).

5.3.5 Project Implementation Plan

The customer and supplier should agree on a project implementation plan, which specifically facilitates progress identification. While the customer clearly has the need to be re-assured of progress, the supplier should not see this process purely as a means of satisfying his customer. He should rather see the customer's involvement as an audit function, which will facilitate the success of the project for both parties.

On the project implementation plan, the proving of high risk areas should be identified as separate activities with early target dates. Both customer and supplier should work together and commit energy and resources to proving the feasibility of these high risk areas. This should include rapid prototyping of critical tasks, early delivery of new hardware platforms and software environments and close monitoring of work to be done by subcontractors.

In addition to focussing on high risk areas, the implementation plan must include a detailed development program for the rest of the system. This program should be detailed to the point where work units of 2...4 weeks duration are identified. This will help catch any variances between the reports and work actually completed. In addition, and often neglected, the interdependencies of work units and subsystems must be identified. Finally the implementation plan must include a resource allocation plan.

This project implementation plan together with the functional specification forms the reference documents for all project progress evaluation.

5.4 Progress Review

5.4.1 Frequency of Review Meetings

The frequency of review meetings should not necessarily be held constant throughout the project, but should be dictated by the state of the project with particular reference to the assessment of high risk areas.

5.4.2 Progress Identification

At each progress review meeting the supplier should report progress in terms of the implementation plan. This must include the following elements:

a. Development progress in terms of work units (i.e. 2...4 man weeks) for each subsystem.

b. Progress in terms of work units for each element associated with proving the high risk areas.

c. Confirmation of interdependencies.

d. Updating of resource allocation plan.

The customer must then verify this progress. It is extremely manpower intensive for the customer to verify all progress. It is therefore recommended that the customer verify the high risk areas and sample other areas in progress verification.

The exact means of verifying progress will depend on the development methodology being used. However, the high risk areas should have an early verification based on the actual running of the critical task and new hardware and software. The remaining of the development should be verified by structured walk–throughs during the early stage and the running of working threads at a later stage.

To make it easy to catch variances between the report and actual work, no software module should be more than four work weeks in length.

An example of a progress report is shown in **Figure 5.4.1.**

Figure 5.4.1: Software Progress Report

PROJECT: United Voltage & Current Ltd.

SW DESIGNER NAME: Mr James R. Bug

DEPARTMENT: EMS Systems

FOR UPDATING: FILL IN NEW ESTIMATES ON DOTTED LINE(S).

UPDATED: .... SIGNED BY: ....

SOFTWARE WORK TASKS:

POS.	SOFTWARE TASK	STATUS AS OF 940320 AND NEW ESTIMATES					
		Planned work [days]	Actual work [days]	Estimate to complete [days]	Start date Planned [yymmdd]	End date Planned [yymmdd]	Status
1	Man-Machine Interface / New Energy Display Function						
1.1	Specification	9.0	9.0	0.0	940227	940310	Completed
					940227	940310	
			.....	.....	......	......	.....
1.2	Report Layout	12.0	8.0	6.0	940311	940401	Started
					940311	940403	Late
			.....	.....	......	......	.....
1.3	Data Base Changes	2.0	1.0	1.0	940319	940320	Started
					940320	940323	Late
			.....	.....	......	......	.....
1.4	Module Testing	3.0	0.0	3.0	940406	940408	
					940406	940408	
			.....	.....	......	......	.....
1.5	System Implementation	2.0	0.0	2.0	940412	940413	
					940412	940413	
			.....	.....	......	......	.....
1.6	Documentation	1.5	0.0	1.5	940418	940420	
					940418	940420	
			.....	.....	......	......	.....

5.5 Follow Up

The significance of project slippage does not increase with time. A delay during the early stage must be taken seriously if the process of contract monitoring is to be effective in ensuring the timeous completion of the work. It is well accepted today that one cannot just add resources at a late stage in the project and expect to make-up time.

The important issue then is that as soon as slippage is identified, appropriate action must be taken. This must be a joint consideration between supplier and customer. The following courses of action are open to the parties:

a. Add additional resources. This is very effective during the early stage of the project.

b. Agree on a change to the Functional Specification. This may be necessary where it is identified that certain functionality is not feasible using the intended hardware and software platform or within the time scale of the project.

c. Agree on a phased implementation. This would be necessary where a subcontractor is unable to deliver the proposed hardware or software on time or where it is identified that certain functionality has been underestimated and will take more effort and elapsed time than originally estimated.

d. Agree on a change to the system design. This could mean using a different hardware platform and/or software environment. This would for example be necessary where the rapid prototyping of a time critical task indicates unacceptable performance.

e. Agree on an extension of the project completion date.

5.6 Summary

In conclusion, the critical success factors for the timeous completion of a software development contract are the following:

a. A partnership relationship between supplier and customer.

b. A good quality Functional Specification.

c. Close monitoring of project progress by both supplier and customer.

d. An implementation plan, which includes specific action to identify potential problem areas at an early stage.

e. Corrective action at an early stage as soon as slippage is identified.

Although a good partnership relation should be the objective of all parties involved, it is important to define the main management requirements also in the contract - just in case. The management requirements should include e.g. procedures concerning the management of special high risk areas, project implementation plan and content of the reviews to insure that the project will be controlled properly.

5.7 Check List For Progress Identification And Follow-up

* Risk Management
  - Are the high risk areas identified?
  - How can these risks be minimized?
  - How are the key resources secured?

* Contract Monitoring
  - Were detailed discussions held already during the proposal time?
  - Does both the user and the designer understand the meaning of the words in the Functional Specification?
  - How is a good relationship established and maintained between customer, supplier and consultant?

* Estimations And Reviews
  - How are project progress routines planned?
  - How is progress identified?
  - How can too optimistic reports be avoided?
  - Are the any standard report layouts specified?

6 TEST AND VALIDATION

6.1 General Remarks

The quality judgement on a software product, at the stage of its issue, has always been difficult to give since historically the software has always been considered as something of impalpable and indefinite, and therefore "not measurable".

The lack of a unique and clear "measure rule" has implied that the software evolution has been in the past not rigorous but subjective.

The quality of a software product is nowadays normally addressed under two points of view: the topological matching and the functional matching.

The topological matching consists in exercising at least once all the branches of the software code; the functional matching consists in exercising at least once all the parameters accessible to the user and in outputting at least once all the possible messages.

For the software projects of telecontrol systems both types of matching are to be taken into account, even though the functional matching is more important from the contractual point of view since the user specifications are essentially functional specifications rather than implementation specifications; the user specifies normally "what they want" and not "how to realize it".

The normal attitude for testing the software is to believe that the software will be wrong and then actively seek out the errors. A lot of errors found in software is an indication that the software is of poor quality and may contain many more errors.

Testing can only show the presence of bugs and never show their absence (Dijkstra). In fact it is impossible to be exhaustive in trying all the inputs combination of software programs.

This means it is impossible to have total confidence in a software program and testing may continue to find a reasonable level of confidence or level of quality.

A measure of the software quality could be the error-rate or the mean-time between failure if a randomly generated test case is being used.

An alternative method is the "**error-seeding**": First errors are added randomly in the program; then the test set runs to find the proportion of those errors that are detected. With that ratio and knowing the number of real errors found it is possible to predict the number of errors left.

The usual way to evaluate the quality of software is by executing the software, using test cases for which the correct results are known.

The examination and judgement of a system (or part of it) in terms of correspondence with the user/customer needs and requirements, quality of performance, degree of effectiveness and so on, are normally addressed as "system test and evaluation".

6.2 Test Planning

The system test should be an ongoing iterative process which begins during the conceptual design phase and extends through all the system phases (coding, integration, installation, operation).

Indeed although ideally it is desirable to wait until the system is fully operational before accomplishing an evaluation of system performance, it is not practical from the stand-point of allowing for possible corrective actions. In the event that the evaluation indicates noncompliance (i.e. the system as presently designed does not meet the operational requirements and fulfill its mission), corrective actions should be initiated as early as possible in the system life cycle. Accomplishing corrective actions after the system is fully produced can result in extensive modification programs, which are costly. Thus, it is feasible to establish an overall test program which allows for the evaluation of the system on an evolutionary basis.

During the earlier phases of the system production bench test models, engineering models, and service test models should be built with the intent of verifying certain performance and functional characteristics, as early as possible.

Very useful at this stage are the so-called *"stubs"*. The stubs simulate – in terms of CPU time, memory utilization and interfaces – software modules and programs already designed but not yet produced. As the coding phase proceeds the single stub will be replaced by the real module or program and more and more realistic performance and functional checks will be conducted. Problems introduced by each new module will tend to be more obvious and will be easier to fix.

Test software **should** be produced to the same standards as the deliverable software. Test software and data must therefore be documented and subjected to configuration control procedures. This allows monitoring of the testing process, and permits test software and data to be reused later, to verify that the functionality and performance of the software have not been impaired by modifications.

Throughout the various stages of system development, a number of more or less comprehensive tests may be specified; often there is the tendency to design a test for measuring one system characteristic, design another test for measuring a different parameter, and so on. In this way the amount of testing may become excessive and quite costly; an incremental approach should therefore be preferred in the testing activity during the system development.

On the basis of the above considerations the following test phases can be identified:

- module (or unit) level test
- software integration (program level) test
- software validation (system level) test
- hardware/software integration test
- Pre-Factory Acceptance Test (Pre-FAT) at the supplier's factory
- Factory Acceptance Test (FAT) at the supplier's factory
- System Acceptance Test (SAT) on site.

For each of the test phases, a test planning document is to be prepared. It should include the following items:

1. The strategy, scope and schedule of the test
2. The definition of organization, administration and control responsibilities
3. The definition of the general test conditions and logistic resources (test environment, facilities, support equipment, necessary simulations)
4. The description of the test specifications (see Chapter 6.3.)
5. The description of conditions and provisions for a retest phase (methods for conducting additional testing in case of a reject situation)
6. The identification of test documentation (test reporting requirements).

The test planning document should be used as a valuable reference throughout all the life phases of the system.

The scope of the testing activity normally changes with the level of integration of the components.

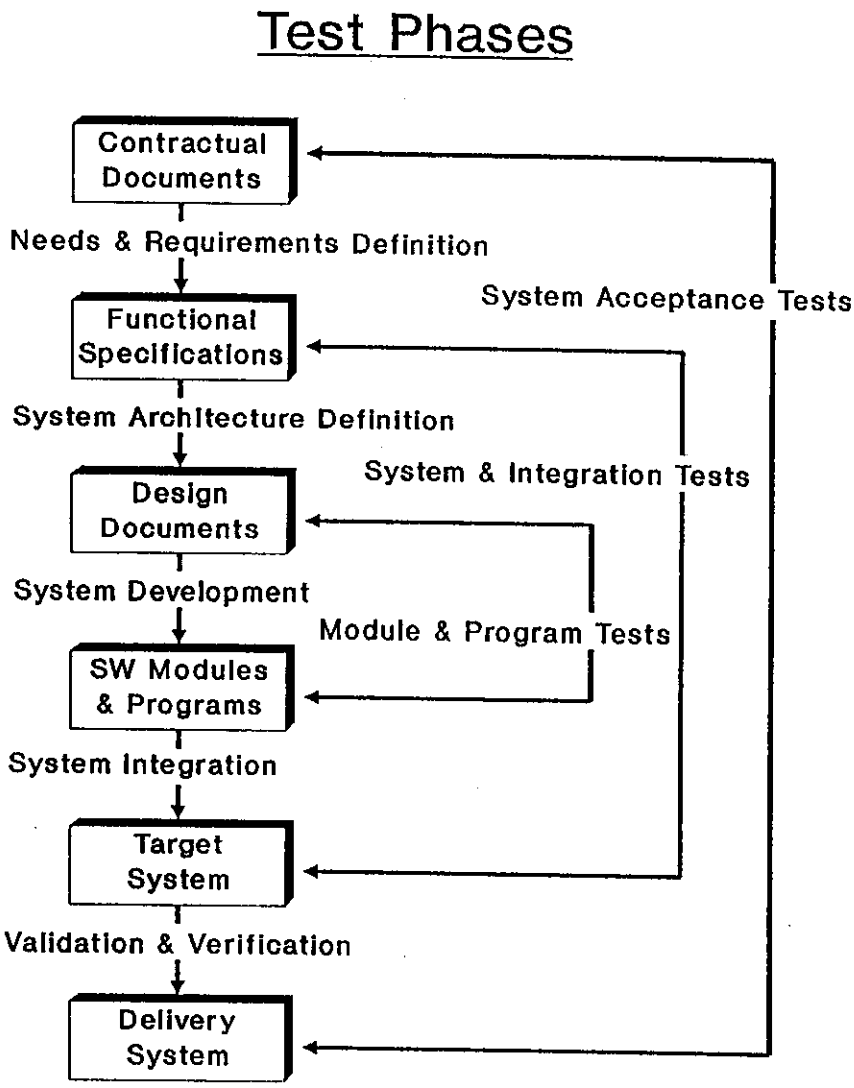
For the module and program level tests should be prevalent for the need of verifying the topological matching of the software; for the succeeding test phases the verification of the functional matching is predominant.

The module and program level tests are considered as informal testing activities and they normally involve the same programmers who produced the modules.

The software validation testing phase constitutes the beginning of the formal testing activity, that is the validation against the user requirements in terms of functions and performances. For this and succeeding testing phases it is necessary for support from independent software test organization since the test conductors should be different from the programmers. The system acceptance tests require of course the witnessing also of the user.

Figure 6.2.1 shows the relationship between the test phases and the phases of the system production.

Figure 6.2.1: Test Phases



6.3 Test Specifications

The Test Specifications should be written immediately after the Functional specifications and should test the functions outlined in the Functional Specification.

Particular attention should be paid to the preparation of the test specifications, also called test procedures.

The scope of the test procedures varies according to the test phase.

The module level tests should demonstrate the adequacy of the module design. Automatic standard tools are normally used for exercising all the branches of the code; the checks should include both correct and incorrect "stimulus". In this phase it should be also verified that the coding standards agreed upon during the system design phase have been correctly adopted by the programmers. The checks should be both visual and automatic.

The program level tests should demonstrate the right implementation of the interfaces among the modules; also in this phase automatic testing tools are normally utilized.

For the software validation, integration and system acceptance testing phases, the test procedures represent the means for verifying the correct implementation of the user requirements and constitute the basis for the system acceptance.

As shown in **Figure 6.3.1**, the contractual documents are the starting point for the test procedures preparation. The analysis of the functions required by the user brings to the definition of the functional check lists; they contain the list of each function to be tested and the reference - in terms of pages and/or paragraphs - to the contractual documents where the relevant functional requirement is indicated.

A careful examination of the functional check lists should bring to identify functional chains easy to test and containing more elementary functions.

The functional chains are then translated into test procedures.

A possible layout of a test procedure is indicated in **Figure 6.3.2.**

Figure 6.3.1: Testing

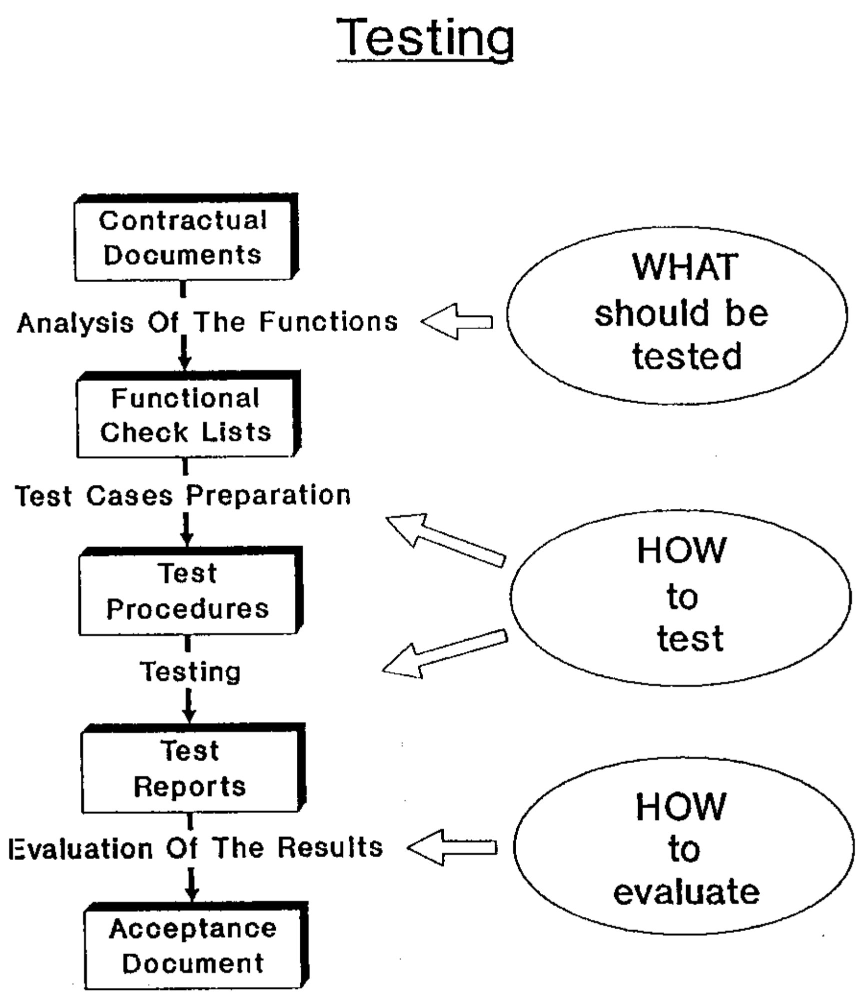


Figure 6.3.2: Outline Of A Test Procedure

XXX.NAME *XXX = Test procedure number* *NAME = Name (title) of the test procedure*
XXX.1 Test procedure purpose Description of the functions tested with the procedure
XXX.2 References Indication of the references to the contractual documents
XXX.3 Test procedure sequence requirements Number XXX of the test procedures to run succesfully before running this test procedure. This field is filled in only if necessary.
XXX.4 I/0 Media to perform the test Indication of the I/O media foreseen for the test. The input media may be simulators, console keyboards, data files, etc. The output media for the test results may be printouts, hardcopies, monitors, etc.
XXX.5 Assumptions Hypothesis and/or constraints (if any) to take into account for the test execution. In this section the test environment, the facilities, the test personnel, the support equipment, etc. are indicated.
XXX.6 Step-by-step instructions for conducting the test This section must be detailed and accurate as much as possible. The test conductor is normally a people different from the test procedure writer and there must be no doubt and uncertainty for "what" and "how" a function is to be tested. The operating instructions must contain enough information to enable two or more different people to execute the test procedure in the same way. Each step of the procedure must contain also the field OK/NOT OK that will be marked by the test conductor.
XXX.7 Expected results This section states the expected results of the test. On the basis of this section the test conductor will mark each step of the procedure with "OK" or "NOT OK".

It is very important to verify that all the functions of the functional check lists are covered at least once from the test procedures; the utilization of a coverage matrix should allow the above verification.

Some test specifications must deal with software and functions of Remote Terminal Units (RTUs) as well as data exchange with telecontrol centres. The utilization of one or more real RTUs (for direct links and star-shaped network) is suggested for a more effective testing of the data exchange. The simulation of a poor quality of the telecommunication links (by means of a noise generator) is useful to verify the recovery actions provided by the telecontrol system.

Tests must be conducted to verify software maintainability by simulating the insertion of a new application program into the system; this allows also to check the effectiveness of software tools (assembler, compiler, utility programs etc.) and operating system for developing and inserting new programs as well as the programs documentation.

The database updating procedure must be carefully tested also. The electrical network by its nature changes frequently, so that periodically there is the requirement to update the database describing the controlled network. The database updating procedure must be simple, short and safe especially for the installation and verification phases, since it is fundamental during the operating life of the telecontrol system to reduce to a minimum the changeover time from an old database to a new.

It is essential to specify a procedure, that guarantees that the final, tested system includes **the same software version** that was tested. This can be accomplished e.g. by generating the complete supplier-made software from source in the beginning of the factory tests. Furthermore, **no changes or corrections** should be allowed (without the permission of both parties) until all functions have been tested. From performance point of view it is also important, that **all specified software modules** are active (i.e. not passivated) during the tests.

6.4 Test Reports

After the definition of the test procedures it is possible to start with the test and demonstration of the system. Throughout the test process, data is to be collected for later analysis and therefore for giving the assessment on the system performance and effectiveness. The test procedures must indicate the data to acquire and for each test procedure a test report must be drawn up.

The test report should contain at least the following information:

a. Date of the test procedure execution

b. Name of the test conductor

c. Name of the user representative (applicable for the system acceptance tests)

d. Identification of the test procedure (e.g. XXX.NAME)

e. Test repetition number (used in case of retesting)

f. Test procedure notes, for any remarks concerning the test execution

g. Test output document list; this field lists the documents (e.g. printouts, files, hardcopies) produced during the test

h. Retesting considerations. This field contains advices in the event of test repetition (e.g. if there are other test procedures to run again for a non-regression verification)

i. Test revision team comments. This field contains the overall judgment on the test.

6.5 Post Delivery Follow-up

Even though the testing of a system is carried out with high accuracy and diligence, the true performance and effectiveness of the system can be established only with the system in the real operation environment.

It is therefore important to follow-up the behaviour of the system after its delivery to the user. Specific forms must be designed for a correct follow-up data collection.

The method for data collection may vary considerably, as there is no standard format for its carrying out and the information desired may be different for each system. However the following rules should apply:

1. The data collection forms should be simple to understand and complete (preferably on single sheets), as the task of recording the data may be accomplished under various environmental conditions by a variety of personnel skill levels. If the forms are difficult to understand, they will not be completed properly and the needed data will not be available.

2. The factors specified on each form must be clear and concise and not require a lot of interpretation and manipulation to obtain.

The people who must complete the forms must know the system and the purposes for which the data is being collected.

A means must be also provided for the retrieval, formatting, sorting, analysis and processing of the data by a designated centralized facility.

The results should be distributed to engineers and managers for decision making if the system can remain just as it is or if some modifications or improvements are necessary.

6.6 Check List For Test And Validation

* Test Phases Identification
  - Where are the different test phases described?
  - What is the strategy for the module level test?
  - Who is responsible for the program level test (software integration)?
  - Which documents will the system level test (software validation) produce?
  - Are the hardware requirements and time-schedule defined for the hardware/software integration test (system validation)?
  - Which are the test conditions and environment for the factory acceptance test?
  - Which are the conditions for the approval of the field acceptance test (system acceptance)?
  - How is a possible retest performed?

* Test Planning
  - Where is the scope of the test described?
  - Is the test environment defined and approved?
  - Is the test team identified for each test phase?
  - How and by whom are the test results evaluated and administrated?

* Test Documents
  - Who is responsible for the planning of the test procedures?
  - When will the functional check lists be ready?
  - Are there specified outlines for test reports?
  - Where is the contents of the final acceptance document specified?
  - How is the system functionality followed-up after delivery?

7 QUALITY ASSURANCE

7.1 Concept of Software Quality

The concept of **quality** can be understood from a wider or a narrower perspective. Quality is often connected to a company's final products, i.e. the word quality is used to measure a products' suitability for intended use. Hereby the quality mainly is defined by its user, and not by the supplier.

However, it is important to accept, that the quality of a product or a service is a result of a complete system, namely the company's **Quality Assurance System.** This wider perspective includes in fact all functions and activities, performed by the company.

In the standard ISO 8402 Quality is defined as

> *"the totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs".*

This is a definition of quality which emphasizes the user view - **fitness for purpose.**

Quality can be viewed differently, for example:

Product view	-	number of features
Manufacturing view	-	conformance to requirements
Value view	-	quality at acceptable cost
Transcendental view	-	felt rather than measured.

A first step to introducing quality must be to understand precisely what are the points of view involved.

To control quality we must be able to measure some objective assessment that can be quantified.

It is not possible to measure quality directly, but the concept of quality can be decomposed into a number of factors. These are are further decomposed into criteria, which are evaluated using appropriate metrics.

Examples of criteria are: device independence, self-containedness, accuracy, completeness, integrity, consistency, accountability, device efficiency, accessibility, communicativeness, self-descriptiveness, structuredness, conciseness, legibility, augmentability, etc.

Each individual factor has to have a clear definition based on the criteria adopted and measured.

As examples of factors we can mention:

Correctness - Extent to which a program satisfies its specification and fulfils the user's objectives

Reliability - Extent to which a system can be expected to perform its intended function with required precision

Testability - Effort required to test a system to ensure it performs its intended function

Flexibility - Effort required to perform modifications

Reusability - Extent to which an element of software can be used in others applications.

Although possible to measure the factors it is impossible to weight and combine them to find an absolute measure of quality.

7.2 Quality Assurance System

In a project, quality must be built in from the start. It cannot be added in at the end if it is found that the project has too little quality.

Hence from the beginning the values to be achieved by the quality factors have to be defined and the procedures to follow in the development have to be stated.

To achieve and control the quality of software, further to adopt the best engineering processes and management methods, a **Quality Assurance System** must be created.

A quality assurance system is a plan with systematic description of all actions necessary to provide adequate confidence that the item or product conforms to established requirements. It has to be based in a quality assurance staff or department, with independence from the line executive, together with a quality manual or a set of procedures and standards that represent current best industrial practice.

The aim is to say before the start of the project how it is intended to undertake the engineering development and afterwards to be possible to prove that it was done that way.

7.3 The Quality Plan

Each project must have a project plan, in which all the development, control and evaluation steps are identified as well as the different responsibilities.

As part of the project plan, or as an independent document, the project shall have a quality plan indicating the quality control practices that are going to be followed in the project.

Some important points of such plan are:

* Purpose of the plan
* List of all the documents referenced in the text
* Description of management organization
* Description of each organizational dependency of the elements responsibles
* List and sequence of tasks, specially those of software
* Quality assurance
* Identification of elements responsible for each task
* Definition of documentation requirements
* Identification of the standards, practices and conventions to be applied
* Presentation of reviews and audits to be conducted
* Explication of methods to identify software products and control the implementation of changes
* Procedures for reporting and resolving software problems
* Description of special tools techniques and methodologies employed in the quality assurance activities
* Methods to be used to control versions of identified software.

7.4 Quality Standards

The Quality Plan should be a very complete document. However, it is not necessary or even desirable to create completely new theories for the actual project, because there are suitable quality standards available to be adopted. Some of the most important in the field are:

ISO 9000	–	Guide to quality management and quality assurance standards.
ISO 9001	–	Quality systems: Assurance of Design/Development, production, installation and servicing capability.
ISO 9000-3	–	Guidelines for the application of ISO 9001 to the development, supply and maintenance of software.
ISO 9002	–	Quality systems: Assurance of production and installation capability.
ISO 9003	–	Quality systems: Assurance of final inspection and test capability.
ISO 9004	–	Guide to quality management elements.
ANSI/IEEE 983-1986	–	IEEE guide for software quality assurance planning.
IEEE-1058-1987	–	Software project management plans.
ISO/DP 9294	–	Guide for management of software documentation.
ANSI/IEEE 828-1983	–	IEEE standard for software configuration management plans.
IEEE 1042-1987	–	A guide for software configuration management.
IEEE 1063-1987	–	Software user documentation.
ANSI/IEEE 830-1983	–	IEEE guide to software requirements specifications.
IEEE 1012-1986	–	IEEE standard for software verification and validation plans.

7.5 Software Specifications

Before it is possible to measure the quality factors it is necessary to clearly define the development process.

A document mandatory for that, to be included in the quality plan, is the **Software Requirement Specification.** It shall specify all the relevant requirements related with system functionality as well as performance, reliability, security, etc.. It should be done by the customer and the supplier in close cooperation and be approved by both before the development stage. It has to be precise enough to allow test validation during the acceptance stage.

A good software requirement specification shall be:

Unambiguous – Every definition of requirement needs to have only one interpretation.

Verifiable – It has to be possible to check that the software meets every requirement (possibility to measure). Words as: good, well, many, usually, accurate, etc., are impossible to define in terms of metric. As an example the requirement: "the program output shall usually be given within 10 seconds" is not a verifiable requirement (there is no measure for usually). The definition should quantify the amount of times the outputs is given in 10 seconds.

Complete – This implies that all significant requirements are defined, including the design constraints and the external interfaces. The responses of software to all valid and invalid classes of input values should be specified.

Consistent – Each individual requirement definition should not conflict. The same object should not be defined in different terms along the document, neither should its characteristics be changed.

Modifiable – It should allow easy modifications. It needs to have an easy-to-use organization, the same requirements should not appear in more than one place, or then it should include cross reference tables so that when the document is altered it would not became inconsistent.

Another important document is the **Software Design Description.** The major components of software are described in it; phases, tasks and activities must be identified together with the output of each task. This way it is possible to identify the breakpoints or milestones at which certain predefined products are available and hence to define the appropriate process metrics.

The software specifications should strictly follow standardized forms, defined in a **Software Specification Manual**. Each software module and entity should have at least a minimum set of identification and functional parameters, such as

* module name
* module type
* file
* description
* software language
* calling programs
  - call parameters
  - exit parameters
* called programs
* data used
  - disk files
  - external areas
  - use of registers
* designer and date
* revisions
* other references.

The use of standardized software design and layouts should be a requirement for the software delivery. The implementation could e.g. be checked during quality audits at the supplier.

7.6 Internal Development Guidelines

7.6.1 Configuration Management

The generic standards exhort to keep records of all formal documentation of project and also working records as minutes of meetings, change forms, etc. All documents that are considered to be necessary to prove that the quality plan has been followed, and to be able to sustain the software during its operational life, must be kept.

To help this activity it is important to use a configuration management system to record the information about all items produced, and also its versions and variations.

Similarly, all changes in the software shall be carefully controlled. Changes are always very expensive, and uncontrolled changes are one way poor quality can enter into a system. Hence a change control system associated with the configuration management system is of great importance.

When a change is proposed an appropriate form shall be filled recording all the details. All change shall be approved by a "change control board" that will analyze the costs and benefits of the action and will decide to approve or cancel it. Change procedures are discussed in Chapter 3.7.4.

It is important to remember that when a program is operational, there are no "quick fix faults", because all the corresponding documentation has to be changed too.

Due to the costs involved it is recommended that the accepted changes are grouped to be done simultaneously so to spread the costs of documentation, tests, etc.

7.6.2 Internal Management Functions

In a software project of medium or large size it is possible to identify four technical management functions that must be done by separate persons, each having his own unambigiously defined responsibility area:

* Project management
* Software development
* Software control
* Software evaluation.

The **Project Manager** is directly responsible for the overall project as well as the software development and the software control. He is the single technical point of contact with the customer.

The **Software Development Manager** is responsible for the preparation of the software design description and others development plans. A more qualified member of the group is usually the leader and is responsible for the technical direction of the team assuring the overall coordination among development team numbers.

The **Software Control Manager** is responsible for establishing adequate control of the code and documentation, which involves setting up the configuration management control procedures at the beginning of the development and maintaining these procedures throughout the project. This requires high formalized procedures with rigorous implementation in order to maintain positive identification of software products during the project, as well as accurate record of changes of these identified products.

The **Software Evaluation Manager** has the responsibility to perform the technical review involving directly the code being developed, and the compliance review, which consists in determining if the items deliverables agree with the requirements specified.

The team has to work in parallel with the development team since the sooner an error is discovered the easier it is to correct.

The objective of this team is to find problems and not solutions and must report to a manager at a higher level than the project management.

7.7 Check List For Quality Assurance

* Quality Plan
  - Is there a quality assurance manual available for each project team member?
  - Where are the requirements for the documentation specified?
  - Which standards should be applied?
  - When and by whom are quality reviews and audits performed?
  - Which are the procedures for reporting and resolving software problems?
  - Are there documented methods to control software versions?

* Software Requirement Specification
  - Are the requirement definitions unambiguous?
  - Are all requirements verifiable, i.e. are specific measurable quantities defined?
  - Are all possible functional input and corresponding output alternatives defined?
  - Is the same object defined only once in the specification, or at least always in the same way?
  - Is the same parameter appearing only in one place in the software data base?

* Software Design Description
  - Are all major software components described?
  - Are the inputs and outputs of each task defined?
  - Are the project milestones identified?

* Software Specification Manual
  - Is there a specified standard layout for a software module?
  - How, when and by whom are the implemented software modules checked according to layout?

* Development Guidelines
  - Are there methods for configuration management, i.e. which software items are produced, which versions are valid, etc.
  - Are there different persons in charge of the management functions for
    . the project management
    . the software development
    . the software control
    . the software evaluation?
  - Are there documented guidelines for the systematic construction of test cases?

8 CONCLUSIONS

The problems of managing a software project are diversified. Project control can be considered equal to managing a company. You have to organize functions, motivate and control persons, write instructions, supervise the progress etc.

To plan the control system for a project is a comprehensive task. If you think of the control procedures only when the project has already started, you may find yourself in a situation, where the project control system is completed at the same time as the project. It is useful to think of the basic implementation facts already in the study phase or during the tendering period. This gives you time to prepare the project thus giving it a fresh start without start-up problems and delays.

Typical symptoms of a badly managed software project are that the persons involved are in a steady hurry, plans are often changed, task priorizations are altered every second day, decisions are not made etc. In a well organized project there are no dramatic cuts made and no fires put out every day. The operation is handled with routine. Problems are solved and plans are followed, and changed if needed.

The purpose of this report is to give some guidelines for software project control. Project management, however, is still more of an art than a science. A software project manager cannot be formed from a few simple lessons or advise. The only correct way is the long way - through extensive background training and exposure to project experience.

Effective project management does not happen by itself. It presumes from the company management a strict grip and a decision to develop the activities - and not least the quality system. The development should be given all support and resources needed. Once fulfilled, the development work will be returned in multiple in the form of a more profitable implementation of projects.

9 APPENDICES

9.1 Abbreviations List

ANSI	American National Standards Institute
CAD	Computer Aided Design
CASE	Computer Aided Software Engineering
CPU	Central Processing Unit
ECCP	Engineering Change Control Procedure
ECN	Engineering Change Notice
ECP	Engineering Change Proposal
ECR	Engineering Change Request
EMS	Energy Management System
FAT	Factory Acceptance Test
GKS	Graphical Kernel System (software for hardware independent graphical functions)
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
I/O	Input/Output
LAN	Local Area Network
MMI	Man-Machine Interface
MOTIF	Graphic User Interface (from OSF)
OSF	Open Software Foundation (computer supplier group for development of products in the UNIX environment)
POSIX	Portable Operating System Interface for computing environment
PC	Personal Computer
Pre-FAT	Pre-Factory Acceptance Test
QA	Quality Assurance
RTU	Remote Terminal Unit

SAT	Site Acceptance Test
SCADA	Supervisory Control And Data Acquisition (often synonym to Telecontrol System)
TCP/IP	Transmission Control Protocol Internet Protocol
UNIX	Standard Operating System for Open Systems
WAN	Wide Area Network (global network for long range communication)
X-OPEN	Computer supplier group, working with specifications for Open Systems

9.2 Report Team

This report has been written by the following team from the Working Group 01 of CIGRÉ Study Committee 35:

*	Christoffer Enckell	Finland (coordinator)
*	Erasmo Bitetti	Italy
*	Tom N Carolin	South Africa
*	António de Carvalho	Portugal
*	Quy Le Nguyen	Australia
*	John M Thorson Jr.	USA
*	Marc Virlogeux	France

with supporting comments from the other members.

Special thanks to Study Committee 35 members for their valuable comments.

≈ ≈ ≈ ≈ ≈

Publié par le CIGRÉ
21, rue d'Artois
FR-75 008 PARIS
Tél. : +33 1 53 89 12 90
Fax : +33 1 53 89 12 99

Published by CIGRE
21, rue d'Artois
FR-75 008 PARIS
Tel : +33 1 53 89 12 90
Fax : +33 1 53 89 12 99