Project Based and Methodological Design

A Practical Approach

Saxion University of Applied Sciences

J.W. van Dijk

Saxion University of Applied Sciences
J.W. van Dijk

Version 1.5.1
August 2013

Foreword

This document aims to give a practical approach of using the V-model as a methodological way to plan activities in a project.
The V-model was originally developed to structure software designs in a methodological way. Nowadays the V-model gets more and more important as a methodological approach in Systems Engineering. The V-model therefore is not only used for software design but also the design of mechanical, electrical and electronic systems or even any combination of them (i.e. mechatronic systems).

The *V*-model has now found widespread application in commercial as well as industrial projects throughout the project lifecycle. In Project Management it is a method which can be used in combination with PRINCE2, which can be used for the management of a project, and describes methods for project management as well as methods for system development.

Enschede, July 2013
Johan van Dijk

Notes

Contents

Abbreviations

A	Analysis – Method for verification
AD	Analogue to Digital
CE	Conformité Européen
RCRM	Requirements Cross Reference Matrix
D	Demonstration – Method for verification
DA	Digital to Analogue
DQ	Design Qualification
FAT	Factory Acceptance Test
FR	Functional Requirements
I	Inspection – Method for verification
IQ	Installation Qualification
MTBF	Mean Time Between Failure
MTTR	Mean Time To Repair
OQ	Operational Qualification
PCB	Printed Circuit Board
PQ	Performance Qualification
SAT	Site Acceptance Test
T	Test – Method for verification
TR	Technical Requirements
UR	User Requirements

List of figures

List of tables

1. Introduction

In projects four major phases can be distinguished:

- Initiation
- Planning
- Execution
- Closure

These four phases are called the Project Life Cycle.

Figure 1.1: Project Life Cycle

During the Project Life Cycle all kind of activities take place which have to be managed. Different types of activities are described in paragraph 1.1. Although is goes beyond the scope of this document in paragraph 1.2 an introduction is given in managing a project during its life cycle.

1.1 Activities during a project

During the Project Life Cycle all kinds of activities take place when a product or system is being designed. These activities can be characterized in three main ways:

- Improvised activities (see 1.1.1)
- Routine activities (see 1.1.2)
- Project-based activities (see 1.1.3)

Figure 1.2: Types of Activities

To show the special position that project-based activities occupy, all three types of activities will be described below.

1.1.1 Improvised activities

An improvised work approach to new activities is one option for designing a product or system. People usually improvise when something unforeseen occurs, which requires an immediate response. Their reaction will be an ad-hoc one: not according to a laid down plan, but decided on as events unfold.
Since there are no directions about what to do that can be given beforehand, it is hard to predict with any accuracy the outcomes of working in an ad hoc manner. The advantage of not having any directions is that new situations can be met in a flexible manner. However, while the large amount of freedom may be welcomed by the improviser, it carries the risk of chaos and could put the design process under a lot of pressure. Designers in a design process in which there is a lot of improvisation going on are likely to be working under a certain amount of strain. Having to constantly adjust to changing working conditions is, after all, quite stressful.

1.1.2 Routine activities

Routine activities are activities that are repeated frequently and are relatively predictable. The design of a product or system will be carried out according to a predetermined pattern. Since there is no precedent, it is not necessary to constantly think about what is to be done next. Note that this type of activities almost never applies for designing something new. However within organizations most of the activities carried out belong to this type. Production line activities, sales procedures, purchasing procedures and administrative activities are some examples.

1.1.3 Project-based activities

Project-based activities fall roughly midway between improvised and routine activities. They are non-recurring and have limited duration, but are reasonable predictable.
In order to increase this predictability, the work should be done according to a plan. Plans gradually illuminate each phase of the design process. Before the activities get underway, some time needs to be spent on working out what the aims are and how to achieve them.
For this to be effective, large (design) projects are often divided up into a number of phases. After each of these phases, the aims and procedures may be adjusted.
Projects often have an organization of their own, one created specially to deal with the needs of the project. People who do not normally work together may do so as part of the project group. Each will have their own specific tasks.

A project-based activity is not an objective in itself, of course. It is a way of structuring activities that are less predictable than others and that fall outside the scope of normal activities. These structured activities then become easier to manage and monitor.

1.1.4 Comparison of activities

The table below contains a summary of the properties of the various types of activities.

Table 1.1: Comparison of activities

	Improvised	**Project-based**	**Routine**
When?	Ad hoc (suddenly)	Predictable	Repetitive
Result?	Uncertain	Reasonably certain	Certain
Familiarity?	New, sudden	New, planned	Well-known
Freedom?	A lot of freedom	A suitable amount	Little freedom
Procedures?	Chaotic	Increasingly clear	Clear

1.2 Managing a project

Project-based activities should be managed. From the project management point of view, the Project Management Life Cycle, the four phases of the Project Life Cycle can be described as in the following paragraphs.

1.2.1 Initiation

Project Initiation is the first phase in the Project Life Cycle and essentially involves starting up the project. A project is initiated by defining its purpose and scope, the justification for initiating it and the solution to be implemented. It is also needed to recruit a suitably skilled project team, set up a Project Office and perform an end of Phase Review. The Project Initiation phase involves the following six key steps: starting up the project, by documenting a business case, feasibility study, terms of reference, appointing the team and setting up a Project Office.

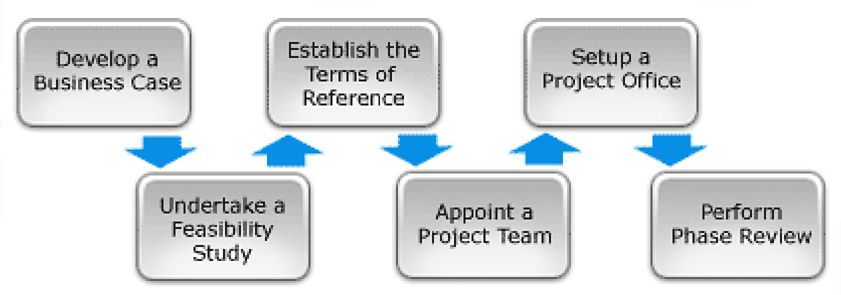


Figure 1.3: Six key steps of the project initiation phase

1.2.2 Planning

After defining the project and appointing the project team, the detailed Project Planning phase can be entered. This involves creating a suite of planning documents to help guide the team throughout the project delivery. The Planning Phase involves completing the following 10 key steps:
setting out the roadmap for the project by creating the several plans: project plan, resource plan, financial plan, quality plan, acceptance plan and communications plan.

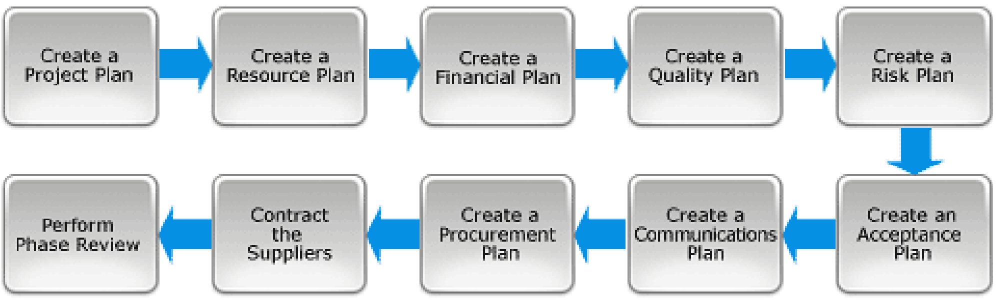


Figure 1.4: Ten steps of the planning phase

1.2.3 Execution

With a clear definition of the project and a suite of detailed project plans, the Execution phase of the project can be entered.
This is the phase in which the deliverables are physically built and presented to the customer for acceptance. While each deliverable is being constructed, a suite of management *processes* are undertaken to monitor and control the deliverables being output by the project.

These processes include managing time, cost, quality, change, risks, issues, suppliers, customers and communication. Once all the deliverables have been produced and the customer has accepted the final solution, the project is ready for closure.

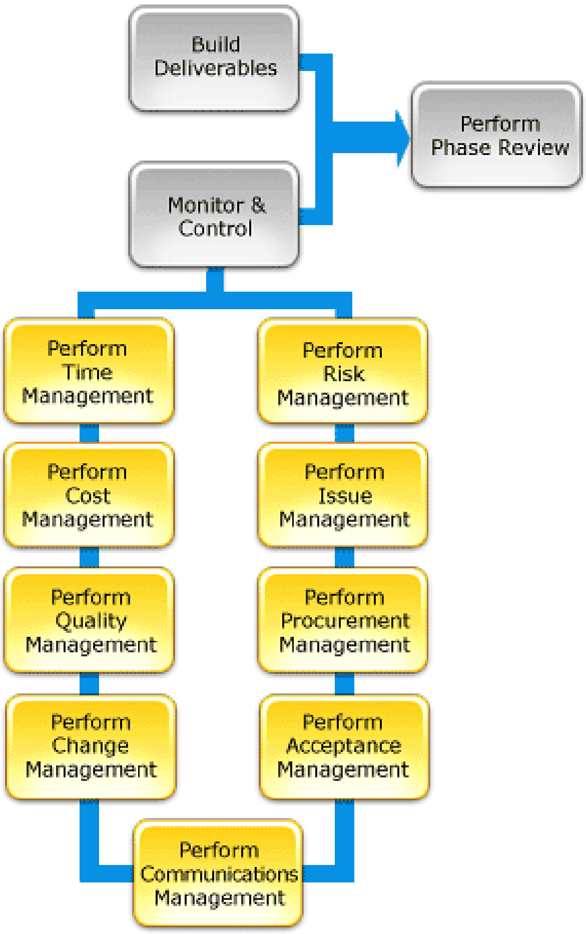


Figure 1.5: Steps of the execution phase

1.2.4 Closure

Project Closure involves releasing the final deliverables to the customer, handing over project documentation to the business, terminating supplier contracts, releasing project resources and communicating project closure to all stakeholders. The last remaining step is to undertake a Post Implementation Review to identify the level of project success and note any lessons learned for future projects.

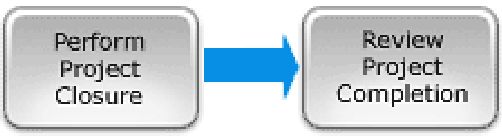


Figure 1.6: Closure of a project

2. Methodological Design

Designing a (new) system by project based activities requires a plan for the activities. When this plan is set up according to a standard method it is called methodological.
There are several methods for a more methodological approach of a design. These methods can be as well linear (i.e. Waterfall or *V*-model) as iterative (Scrum) or a combination of them. They are all based on: <u>think first, then act.</u>

The following objectives are intended to be achieved by a methodological project execution:

- **Minimization of Project Risks**: A methodological approach of design improves project transparency and project control by specifying standardized approaches and describing the corresponding results and responsible roles. It permits an early recognition of planning deviations and risks and improves process management, thus reducing the project risk.

- **Improvement of Quality**: A standardized process model aims to ensure that the results to be provided are complete and have the desired quality. Defined interim results can be checked at an early stage. Standardised project deliverables will improve readability, understandability and verifiability.

- **Reduction of Total Cost over the Entire Project and System Life Cycle**: During the Project Life Cycle the effort for the development, production, operation and maintenance of a system can be calculated, estimated and controlled in a transparent manner by applying a standardized process model. Also the costs during System Life Cycle (production, operation, maintenance and dismantling) can be reduced by taking these aspects into account during development.

- **Improvement of Communication between all Stakeholders**: The standardised description of all relevant (intermediate) deliverables is the basis for the mutual understanding between all stakeholders. Thus, the frictional loss between user, acquirer, supplier and developer is reduced.

When a new project idea arises, in a Methodological Design usually first of all a Feasibility Study (Preliminary Research) has to be undertaken. This Feasibility Study is part of the initiation phase of a project. The main goal of this study is to get an idea if the project is feasible. To get a good idea about the feasibility, activities during the Feasibility Study include among other things, coming up with problem areas, looking for similar products that already exist (market research), studying patents of similar products and evaluating the information found. When the project itself has a lot of uncertainties, and it is therefore not certain that the project will be success, the Feasibility Study can be a project itself in which the most essential uncertainties are investigated and possible solutions have to found.
A definite decision on whether or not to embark on the project can be made subject to the findings of the feasibility study: *the go/no-go decision*.

Figure 2.1: Methodological design starts with a Feasibility Study

2.1 *V*-model

One of the linear methods for methodological design is based on the *V*-model [ref. 1]. The *V*-model is a graphical representation of the systems development process and it simplifies the understanding of the complexity associated with developing systems. In systems engineering it is used to define a uniform procedure for product or project development.

The *V*-model represents the sequence of steps (activities) to be undertaken during the project life cycle and focuses mainly on the execution phase. It also describes the results that have to be produced during the execution of the project. The left side of the *V* represents the decomposition of requirements, and creation of system specifications. The right side of the *V* represents integration of parts and their verification. The *V*-model further demonstrates the relationships between each phase of the development process and its associated phase of testing. *V* stands also for "Verification and Validation".

The *V*-model deploys a well-structured method in which each phase can be implemented by the detailed documentation of the previous phase. Writing down tests for verification and validation in an early phase, well before hardware and software development, saves a huge amount of the project time.

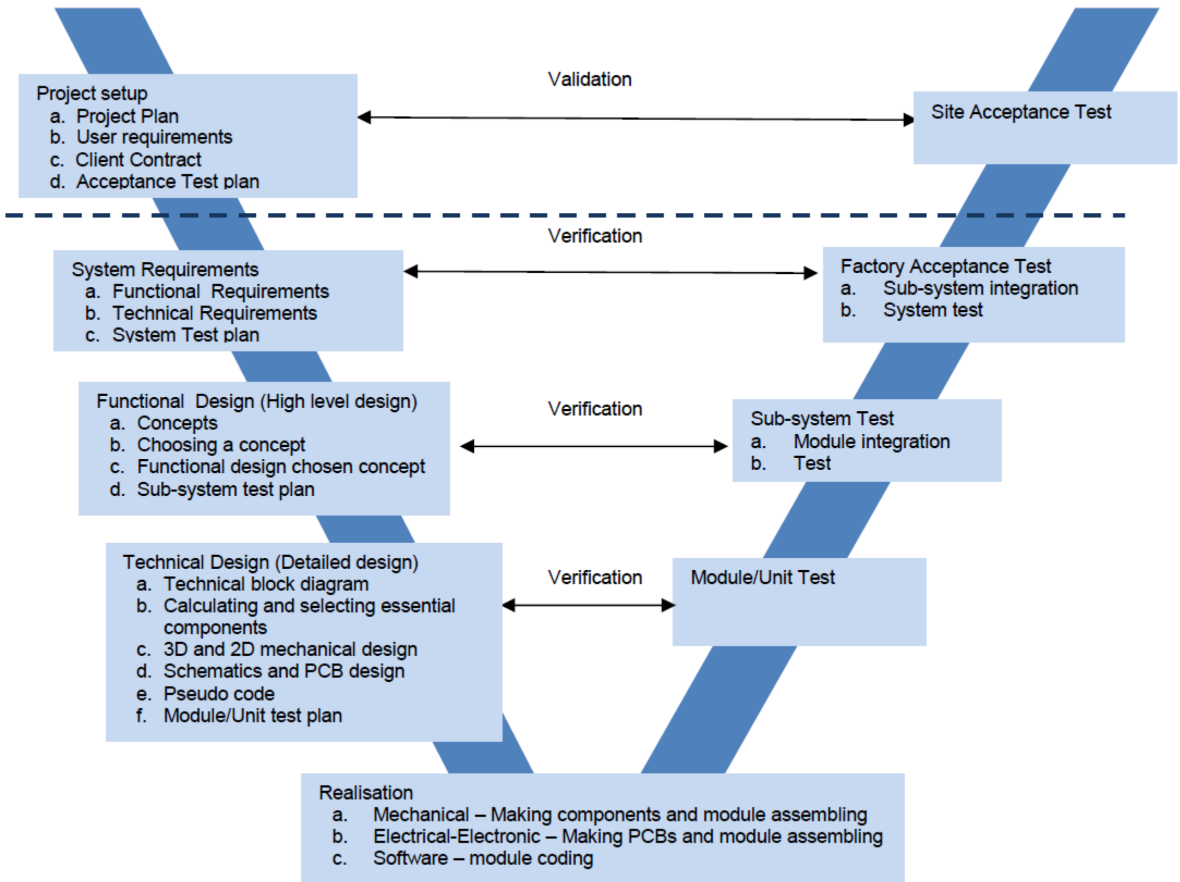


Figure 2.2: V-model

The concept of the *V*-Model was developed simultaneously, but independently, in Germany and in the United States in the late 1980s:

- The German *V*-Model was originally developed by IABG in Ottobrunn, near Munich, in cooperation with the Federal Office for Defense Technology and Procurement in Koblenz, for the Federal Ministry of Defense. It was taken over by the Federal Ministry of the Interior for the civilian public authorities domain in summer 1992.
- The US *V*-Model, as documented in the 1991 proceedings for the National Council on Systems Engineering (NCOSE; now INCOSE as of 1995), [ref. 1] was developed for satellite systems involving hardware, software, and human interaction.

Although originally designed for software development, it has been proven to be also very useful for hardware development or a combination of hardware and software. Its primary use is in Project Management throughout the project life cycle. The *V*-model summarizes the main steps to be taken in conjunction with the corresponding deliverables and it is very similar to the classic Waterfall model as it is quite rigid. On the other hand it also gives a lot of possibilities for iteration.

Advantages of the standardisation using the *V*-model during the project life cycle are:

- Improved communication among the persons involved in the project.
- Uniform procedure in public authorities and commissioned industry.
- Improved product quality.
- Productivity increase by the reduction of familiarization and training times.
- More precise calculation of new projects using standardized procedures.
- Less dependencies on persons and companies.

Advantages during the system life cycle using are a reduction of Maintenance and Change Problems:

- Decrease in maintenance cases, resulting from improved product quality.
- Decrease in the maintenance effort, resulting from the existence of adequate software documentation and an easier understanding because of the standardised structure.

2.2 The Phases of the *V*-model

The *V*-model consists of a number of phases. The Definition and Decomposition are on the left hand side of the *V*, the Realization and Implementation Phase (Electrical, Mechanical and Software) is at the bottom of the *V* and the Composition (integration) and Test Phases are on the right hand side of the *V*.

2.2.1 Definition and Decomposition

Project setup

If a decision to go ahead is made, as the result of the feasibility study, a number of things have to organized, such as:

- The objectives of the project, what functions the product or system must fulfil for the user, must be collected by analysing the needs of the client/user(s). This phase is concerned about establishing how the ideal system has to perform. However it does not determine how the software/hardware will be designed or built.
- The expertise needed (i.e. software, hardware, mechanics, etc) should be determined. As a result a team has to be put together and suitable people for that team have to be found. Permission for them to be on the team has to be requested. Some thought also has to be given on how the tasks should be divided.
- The time and effort needed for the project should be determined.
- The cost (hours, material, courses, consultants, equipment, etc) and benefits of the project must be analyzed.
- A project organisation has to be setup. Authorization to make decisions has to be arranged. What decisions should be made by the project group itself? What decisions should be authorized beforehand and by whom?

The result of this is a '**Project Plan**' (plan for the development). This Project Plan consists of: background information, the project result, project activities, project limits, the product, quality control, the project organization, schedule, development and product cost and benefits and a risk analysis of the project. In appendix B the basic outline for a Project Plan is given.

Also a document called the '**User Requirements**' document is generated. This document will typically describe the system as expected by the user:

- functional,
- physical,
- (human) interface,
- performance,
- security requirements,
- safety,
- environment,
- cost for the user,
- ….

The User Requirements document is one which the business analysts use to communicate their understanding of the system back to the users. The users carefully review this document as this document would serve as the guideline for the system designers in the system design phase.

During this phase also the Site Acceptance Test (SAT) with the client/user acceptance tests (validation) are specified and described in the acceptance test document. The SAT usually directly is related to User Requirements.
In some projects the User Requirements and the acceptance tests are combined in a contract with the client. In this contract also terms of delivery of (intermediate) products, costs, payment conditions, after sales support, etc are described.

System Requirements

The System Requirements phase is the phase where system engineers, who are part of the project group, analyze and understand the business of the proposed system by studying the user requirements document. They figure out possibilities and techniques by which the user requirements can be implemented. If any of the user requirements is not feasible, the user is informed of the issue. The system engineers also add and detail requirements necessary for further development. After informing and discussing the changes with the user, the user requirement document is edited accordingly and the **'System Requirements'** document, which serves as a blueprint for the further development, is generated. This document contains the general system organisation and a detailed specification of interfaces (i.e. inputs and outputs) of the system. It may also hold sample user interfaces, i.e. windows. The Factory Acceptance Test (FAT) document for system testing is prepared in this phase.

Functional Design (High-Level Design)

The Functional Design phase is sometimes also called Architecture Design.
This phase starts with analyzing the requirements, generating concept ideas and selecting a concept for what will be done mechanical, electrical or by software. This phase of the design of generating mechanical concepts, hardware architecture and software architecture can be referred to as high-level design. After one of the concepts has been selected the system is divided into sub-systems and these individual sub-systems are set up in a more detailed way: modules or units. Also the sub-system testing is described in this phase.

<u>Mechanical</u>
Basic concepts have to be generated and one of them has to be chosen.

<u>Electrical/electronic</u>
A basic electrical and electronic architecture has to be made and the sub-systems are split up into modules with their interfaces. The result should be the architecture of the hardware with a functional description of each module and its interfaces.

<u>Software</u>
The sub-systems are split up into modules with their interfaces. It should result in a functional design concept for the software which typically consists of the list of modules, brief functionality of each module, their interface relationships, dependencies, database tables, architecture diagrams, etc.

At the end of this phase a "Functional Design" document is generated. This document contains the different basic concept ideas and a detailed functional design of the chosen concept.

Technical Design (Low-Level Design)

The technical design phase can also be referred to as low-level design. In the previous phase the system to be engineered has been split up into up into smaller modules or units. For each of them essential mechanical and electrical/electronic components are selected. Drawings, schematics and pseudo code are generated and each of them is explained. In this phase simulations can be done to prove that the designs are according to the requirements. Also for each module test procedures has to be developed.

The low level design document or program specifications will contain:

- 2D and 3D models of mechanical components with exact dimensions
- detailed schematics for the hardware based on the selected essential components (i.e. chosen microcontroller)
- pseudo code for the software
- complete list with inputs and outputs for each module/unit
- test specification for each module/unit
- ….

The result of this phase should be that in the next phase special mechanical and electrical components (i.e. Printed Circuit Boards) can be made and modules can be assembled. For the programmer it should be possible to start coding directly.

2.2.2 Realisation

The result of the previous phase should be that in this phase the individual modules can be build.

<u>Mechanical</u>
Making of special components, purchasing standard components and assembling them to modules.

<u>Electrical/Electronic</u>
Purchasing components, making and assembling Printed Circuit Boards (PCBs) and assembling them to electrical modules.

<u>Software</u>
Writing code for modules, according to company standards.

2.2.3 Composition (integration) and testing

Module Test

In the *V*-model of development, unit testing implies the first stage of a stepwise testing process. A fault discovered and corrected in the module or unit testing phase is more than a hundred times cheaper than if it is done after delivery to the customer. Hardware is tested in modules (e.g. separate PCB's or mechanical units).

For software it involves analysis of written code with the intention of eliminating errors and it verifies that the code is efficient and is according to adopted coding standards. Testing is usually white box, having knowledge of the produced code. It is done using the Unit test design prepared during the module design phase. This may be carried out by software developers.

Sub-system Test

In Sub-system or integration testing the separate modules will be tested together to expose faults in the interfaces and in the interaction between integrated components. Testing is usually functional and therefore black box as the hardware and code is not directly checked for errors.

Factory Acceptance Test (System Test)

The individual sub-systems are built together to form a complete system.

The Factory Acceptance Test (FAT) or System Test now will compare the system requirements against the actual system. The system test design is derived from the system design documents and is used in this phase. Sometimes system testing is automated using testing tools. Once all the modules are integrated several errors may arise. The FAT is usually the phase to verify that all requirements are fulfilled and the system can be accepted by the user.

Site Acceptance Test

The Site Acceptance Test (SAT) is the test is the used to determine whether a system satisfies the requirements specified in the requirements analysis phase. The site acceptance test is derived from the user requirements document. The site acceptance test phase is the phase used by the customer to validate the system.

Notes

3. Project Setup phase

Projects are carried out in order to achieve a certain goal. This phase is used to describe the goals of a project and how they can be achieved; the project goals should be made SMART:

- S – Specific

  The project goals must be specific, i.e. described in detail.
- M –Measurable

  The goal must be measurable, i.e. it should be described how to determine whether the goal has been reached (or not) with regard to time, money, quality and quantity.
- A – Acceptable / Attainable

  A person or group must be responsible for achieving the goal.
- R – Realistic

  The goal must be feasible and realistic.
- T – Time-bound

  There is a specific deadline by which the goal must be achieved

During the preparation and execution of a project five aspects are essential [ref. 2]:

- T – Time
- M – Money
- Q – Quality
- I – Information
- O – Organisation

The idea is that by making the project SMART and thinking in advance about the TMQIO aspects of a project the outcome of the project will be more predictable, e.g. a success. For this the user requirements should be clear and a plan for the project should be set up.

3.1 User Requirements

Before a complete project organisation can be setup the user requirements should be made clear. This should be done by consulting the user about is requirements. The result is a User Requirements document. This document is one which the business analysts use to communicate their understanding of the system back to the users. The users carefully review this document as this document would serve as the guideline for the system designers in the system design phase.

The User Requirements (UR) document will typically describe the system as expected by the user:

- functional,
- physical,
- interface,
- performance,
- security requirements,
- safety,
- environment,
- ….

User requirements can be written down in tables, i.e. one table for each major field (functional, physical, ...).

Table 3.1: User Requirements

Number	Requirement
UR0101	Turn on/off possibility
UR0102	Mains-power connection
UR0103	Operation without mains-power possible
UR0104	

3.2 Project Plan

Analysing the User Requirements should result in information on what activities should be performed, required documentation, information, type of expertise what is necessary for the project, the phases of the project and its duration, the planning, the costs and benefits, the risks of the project.
After analysing the User Requirements a Project Plan [ref. 2] can be written, which contains the following essential information:

Background information

The (future) members of the project team, the organization in which the project is being carried out, and the outside world will need background information about the project. One way of doing this is including the information - the "project environment" - in the project plan. The information should be such that even the uninitiated can form a picture of both the project and the organization in which the project is being carried out.

The project result

A section of the project plan describes the ultimate project outcome based on the goals and commission. This section of the project plan explains why the project is being carried out. It also answers the following question: "What is the final result after the project is finished?"

Project activities

A project is likely to involve a number of activities. If you are carrying out a certain type of project for the first time, you are unlikely to be familiar with all of them. Do not hesitate to consult others who have carried out a similar project.

Project limits

What falls within the project's scope and what does not is frequently unclear. Sometimes it seems clear, but it later turns out the project group and the client each had their own interpretation. In order to prevent unclear situations from developing, boundaries of the project should be described.
Important boundaries are:
1. When does the project finish? (i.e. the "length" of the project).
2. What does the project include? (i.e. the "width" of the project).
Sometimes it is also good to describe what is excluded from the project.

Everything that is carried out as part of the project is called the "scope" of the project. The word "scope" literally means size, range and domain.

Products

A section which describes the product or "deliverables" of the project. The project group will carry out a number of activities. These will result in all sorts of products. All products (intermediate or otherwise) together give the 'project results' desired by the client. Designing a product or system and doing so to a deadline, the project's progress is measurable. Activities or a group of activities can be regarded as milestones in the schedule and have deliverables directly related to the milestones.

Quality control

A section on quality control in the project plan deals with the quality of the end product (and all intermediate products). How can you guarantee their quality?
Keep in mind that it is the client who ultimately decides whether the quality is sufficient. Quality can be guaranteed in the following ways:

1. Working/designing/developing in a methodological way (V-model) and reporting on a regular basis.
2. Describe the quality of the intermediate deliverables and end product. Also describe how the client will assess this quality after the project has been completed.
3. The client will have to be reassured early on – during the project – about the quality of the project and the end results. You can do this in advance in your project plan by determining how you plan to assess the quality of each of the intermediate products while carrying out the project.
4. Indicate what "checks" will be carried out to guarantee quality. These could include both tests and technical procedures.

Project Organisation

A section which gives information about the members, client and stakeholders involved in the project. Tasks and responsibilities should be assigned to each member.
See also: Appendix A.

Schedule

Once known what the project's activities are, what products are involved, what project's milestones will be, and who part of the project's team is, a schedule can made (i.e. a Gantt-chart). A schedule provides an overview of activities in relation to time. Eventually already members of the project team can be assigned to various activities. Note that in this phase of the project the activities are globally planned. When, during execution of the project, a certain activity has to be done a more detailed plan for this activity has to be set up.
It is also important to define the milestones of the project.

Costs and benefits

Carrying out a project always takes time (labour hours), and therefore costs money. It also involves the use of resources (i.e. software, computers, test equipment, hardware, evaluation boards, etc.) and has to produce something: the yield or benefit. In this section the total estimated cost of the project and the benefits are given.

Risk analysis

A project's success might come under threat from all directions. Some of the risks a project runs are listed below. They are split up into two types of risks: internal and external.
The internal risks include:
Project not feasible, too many areas of expertise involved, too little experience in project based work, insufficient knowledge or expertise, are unwilling or unable to work together, project members are not sufficiently motivated.

The external risks include:
Too much dependence on other projects, the project's scope has not been clearly defined, insufficient corporation by employees of the organisation.

The risk analysis ends with recommendations to reduce the risks.

4. System Requirements phase

The Systems Requirements phase is the phase where system engineers, who are part of the project group, analyze and understand the business of the proposed system by studying the user requirements document. They figure out possibilities and techniques by which the user requirements can be implemented. If any of the user requirements are not feasible, the user is informed of the issue. A resolution is found and the user requirements are detailed.

In this phase a distinction can be made between functional and technical requirements. Functional requirements are related to system functions, technical requirements are related to system design. The main goal of this distinction is to prevent the designer to start thinking directly in terms of known solutions, without first analyzing -solution independent- the required functions, and checking all possible solutions with the functional requirements. Besides for this reason an exact distinction between the two types is not important.
The user requirements from the previous paragraph are the basis for the functional and technical requirements. Because these are leading for the design process these must be exact measurable and complete.
The user requirements are to be converted into either functional or technical requirements, not in both.

At the end of this phase of the V-model the system that is to be developed is fully defined, both functionally and technically and the '**System Requirements'** document, which serves as a blueprint for the further development, is generated. This document contains the general system organization and a detailed specification of interfaces (i.e. inputs and outputs) of the system. It may also hold sample user interfaces, i.e. windows.

Chapter 4.1 covers the investigation on use scenarios and detailing of the user requirements by the engineers.

Chapter 4.2 covers the functional aspects:
- the functions of the system
- functional requirements to be fulfilled
- its external interfaces
- requirements imposed on its external interfaces

Chapter 4.3 covers the technical (or design) aspects:
- technical description of the system, as a block diagram of subsystems
- technical requirements on system and subsystems
- internal interfaces between the subsystems
- boundary conditions (dimensions. mass, forbidden materials, etc.)
- environmental conditions (vibrations, temperature. etc)
- reliability
- maintenance
- safety requirements (CE, medical, …)

Chapter 4.4 covers the Quality Assurance provisions and results in the Factory Acceptance Test (FAT) document for system testing, the verification that the system functionally and technically is accordant the requirements.

4.1 Functional description and requirements

Before functional and technical requirements can be written down use scenarios should be analyzed and a functional description of the system should be made. This usually will be based on interviews by users and other stakeholders. After interviewing, the interviews should be analyzed and result in a description of the way(s) in which the system is (to be) used.

4.1.1 Use scenarios

A time-line structure or flow diagram may assist writing down the procedure of a use scenario, e.g.:
Start conditions:

- Object in initial position

Sequence:

- Operator enters desired position
- Operator gives 'start' command

End conditions:

- Object in desired position

Different stakeholders may provide different use scenarios. Preferably the complete life cycle of the product is to be covered. Stakeholders may be:

- operator
- manufacturer
- seller
- transporter
- maintainer
- dismantler

Related use scenarios:

- normal operation (leading to requirements on capabilities, performance, ergonomics)
- faulty operation (leading to error handling requirements)
- production (leading to requirements on standard components and tools)
- sale (leading to requirements on features, expandability design, cost)
- transport (leading to requirements on mass, dimensions, handgrips, modularity)
- installation (modularity interfaces, (sub-)system tests)
- maintenance and repair
- dismantling

4.1.2 User requirements

Each use scenario will lead to one or more user requirements. A user requirement is defined as a property that the system will provide in the domain or business process of the user. These should be the basis for the functional requirements of the system. User requirements are described in terms of the user, functional requirements in terms of the designer: User requirements describe the use; functional requirements define the system functions.
User requirements may be formulated vaguely and inconsistently. User requirements are to be converted into either functional or technical requirements. User requirements are organized on basis of use scenarios; functional and technical requirements on basis of required functions or technologies.

An intermediate step may be to list the user requirements following from the definition of the use scenarios, and review these with the stakeholders.
Example:

- User requirement: transfer object from one position to another; with given distance and time.
- Functional requirements: provide drive system with range, speed, acceleration, accuracy, for object with mass m.

4.2 Functional requirements

For the functional requirements the system is described as a 'black-box' with its external interfaces.

Figure 4.1: System as black-box with its external interfaces

Functional requirements are:

- performance
- operation (setting functions = command input. providing feedback = data output)

Performance and operation are the external ('black-box') requirements related to the functional user requirements.

Also the requirements to the external mechanical, electrical and software interfaces should be defined. These are the requirements imposed by the system on its interconnections, to enable it to perform its intended functions within the limits set by the functional requirements.
The requirements on the external interfaces are grouped with the functional requirements, because they are defined black box' as well, but in most cases they do not prescribe a solution principle.
Categories can be:

- user (type of interface, type of user)
- electrical power (supply voltage and current limits, battery operation, earthing)
- electronic interfaces (analogue, digital)
- data-communication (network, wireless)
- pneumatic and/or hydraulic supply (How rate, pressure, connector types)
- mechanical interfaces (pedestal, footprint flange, floor stiffness and flatness)
- substances (gases, fluids or solids functionally interacting with the system)

All functional requirements should be documented and approved by the client. A good way to summarize all requirements is to write them down in a table, i.e. table 4.1. The table can be split up into sub-tables according to the type of the requirement, e.g. **performance** and **operation**.

Remark: Probably not all requirements are **essential** and some are just **nice to have**, then also the requirements-table should be split up.

Table 4.1: Functional Requirements

Number	Requirement	Relation	Value	Unit
FR0101	Turn on/off possibility	=	True	-
FR0102	Mains-power connection	=	True	-
FR0103	Operation without mains-power possible	=	True	-
FR0104	In case of mains-power loss automatic switch to battery operated	=	True	-
FR0105	Automatic battery charge from mains	=	True	-
FR0201	Open loop control mode	=	True	-
FR0202	Closed loop control mode; feedback based on actual speed	=	True	-
FR0203	Closed loop control mode; set-point derived from distance sensor(s)	=	Design goal	-
FR0204	No significant movement during (normal or unwanted) power up or power down	=	True	-
FR0205				

4.3 Technical requirements

Technical requirements are related to system design. Some can be derived from the user requirements, but most will be defined independently. Like the functional requirements they must be exact, measurable and complete. When trying to find all relevant requirements it is good to realize that there are technical requirements imposed by the system on its environment (e.g. supply voltage), and technical requirements imposed by the environment on the system (e.g. mass). The difference is whether the source of the requirement lies inside or outside the system.

The technical requirements can be subdivided in a number of categories, given below. Note that not all will be necessary for each project!

Cost

Examples:

- components, made or supplied
- labour, maximum allowed time against given hourly rate
- special production equipment, maximum allowed time against given hourly rate

Physical properties

Examples:

- maximum dimensions
- maximum mass
- transportability (hoisting provisions, divide into segments)
- styling (shape and color)

Environmental

Examples:

- min. and max. operational and storage temperature range
- vibration limits (operational and survival), usually defined as a frequency spectrum
- shock limits (operational and survival)
- pressure range (operational (vacuum!) and transport)
- humidity, (non)-condensing
- electromagnetic compliance (EMC), could be up to 4 requirements: both conducted and radiated - emission and susceptibility all with frequency range and A/m level.

General design requirements

Examples:

- growth potential/ expandability.

Mechanical design

This includes dynamical and thermal requirements.

Examples:

- operational forces, moments on the system
- required stiff nesses
- restricted materials, processes and parts, e.g. outgassing for vacuum or clean room / poisonous or corrosive materials / material compatibility
- use of standardized major or minor components, e.g. nuts and bolts.

Electrical and electronic design

Examples:

- requirements and restrictions on power, cabling, shielding
- use of standardized major or minor components, e.g. power supply, connectors.

Software design

Examples:

- software coding standards and languages
- implementation proposals, including resulting software requirements
- prescribed states and modes, e. g. startup, calibrate, standby etc.
- error handling, built in test (BIT)

Production and assembly

Examples:

- restrictions on production methods due to installed base of machinery
- use of standardized major or minor components, e.g. power supply, nuts and bolts, software.

Reliability

Example:

- Main Time Between Failure (MTBF) of the system

Maintainability

Example:

- Main Time To Repair (MTTR) of the system

Safety

Examples:

- legal restrictions, e.g. CE, safety, medical standards
- required safety features, e.g, covers, emergency button
- documentation, e.g. user manuals

All technical requirements should be documented and approved by the client. A good way to summarize all requirements is to write them down in a table, i.e. table 4.2. The table can be split up into sub-tables according to the type of the requirement, e.g. cost, safety, mechanical, electrical, ...

Table 4.2: Technical Requirements

Number	Requirement	Relation	Value	Unit
TR0101	Cost of components, based on series production	<	200,-	€
TR0201	Allowed environmental temp. during operation	><	5..40	°C
TR0202	Allowed environmental temp. while switched off	><	-25..60	°C
TR0203	Humidity	<	80	%
TR0401	Allowed environmental temp. during operation	><	5..40	°C
TR0402	Allowed environmental temp. while switched off	><	-25..60	°C
TR0403	Humidity	<	80	%
TR0501	Delay between set-point and control output	≤	1	sec
TR0701	Mains - voltage	><	200..240	VAC
TR0702	Mains - frequency	><	49..51	Hz
TR0703	Battery operated time	>	3	hrs
TR0704	Battery charging time	<	1	hrs
TR0705	Electrical insulation of all conductors	=	True	
TR0706	Fuse(s) on power line(s)	=	True	
TR0707	Shielding against external electromagnetic fields	=	True	
TR0708	Separation of power and data cables	=	True	
TR0709	Cable fracture may not create unsafe situations	=	True	
TR0802	Safe against crushing of fingers between moving parts	=	True	
TR1001	MTBF: 1% of failure probability with 'normal use',	>	5	years
TR1101	MTTR	<	2	hrs
TR1102				

Remark: Probably not all requirements are **essential** and some are just **nice to have**, then the requirements-table(s) should be split up.

4.3 Quality Assurance Provisions

At the end of the design and realization process a system or product is to be qualified. This is done by verification of the functional and technical requirements.

Verification definition

The verification of requirements has three properties:

- Qualification goal
- Verification method
- Verification setup type

Qualification goal:
Qualification of a system or product can have three different goals:

- Installation Qualification (IQ)
- Operational Qualification (OQ)
- Performance Qualification (PQ)

Verification method:
There are four methods for verification:

- Inspection (I): if the requirement is of the type that a certain visible provision must be implemented. e.g. transport handle.
- Demonstration (D): if the requirement is of the type that can be shown qualitatively to be functioning, e.g. hatch opens.

- Test (T): if the requirement is a quantitative parameter that can be proven by means of a well-defined test procedure, e.g. speed.
- Analysis (A); it the requirement is a quantitative parameter that cannot be proven by testing, either because it damages the system, or the test procedure is too complex, e.g. maximum shock level.

Verification setup type:
Requirements can usually be grouped into a limited number of types, which can be verified with one test or analysis setup per type.
Verification setup types are e.g., accuracy dynamics, thermal, magnetic, control.

The table below, called the Requirements Cross Reference Matrix (RCRM), gives method, goal and setup type for all requirements. For those requirements that are to be verified by means of or testing (T) it refers to the paragraphs in which a global description of the test setup and procedure are given. More detailed descriptions of these will be given in a Test and Analysis Report (TAR).

Table 4.3: Requirements Cross Reference Matrix

Number of requirement	Goal (IQ, OQ, PQ)	Method (I, D, T, A)	Setup type	TAR setup paragraph	TAR procedure paragraph

Verification by means of testing (T) or analysis (A) generally involves five steps:

- Definition of requirements that are to be verified
- Definition of test setup or analysis method
- Definition of test plan or analysis procedure
- Definition of results that are to be recorded
- Definition of method for processing and evaluation of results

Notes

5. Functional Design phase

Goal of the Functional Design phase is to generate, for the system or product to be designed, basic concept ideas, select the most promising concept and detail it into functional diagrams which directly can be used in the following phase in which simulations can be done, components will be selected and drawings will be made.

Generating concept ideas and selecting a concept is usually a quite chaotic process involving several iterations of creativity and analysis. Common steps that can be distinguished are:

- The main requirements of the System Requirements are analysed.
- Generation of possible concept principles for the main function(s) of the system, in the form of sketches, equations, block diagrams and/or mass-spring systems.
- Analysis to motivate compliance to the requirements of a concept principle, preferably quantitatively.
- Sometimes the concept principle for a function needs to be tested using some existing hardware.
- Evaluation and selection of concept principle, using e.g. a morphologic overview.
- Generation of one or several concept designs, in the form of sketches, block diagrams of servo control, electr(on)ical hardware and/or software.
- A global dimensioning of the most promising concepts is done.
- An analysis on essential aspects is performed, to motivate quantitative compliance to the main requirements.
- A choice of the most promising concept is made.

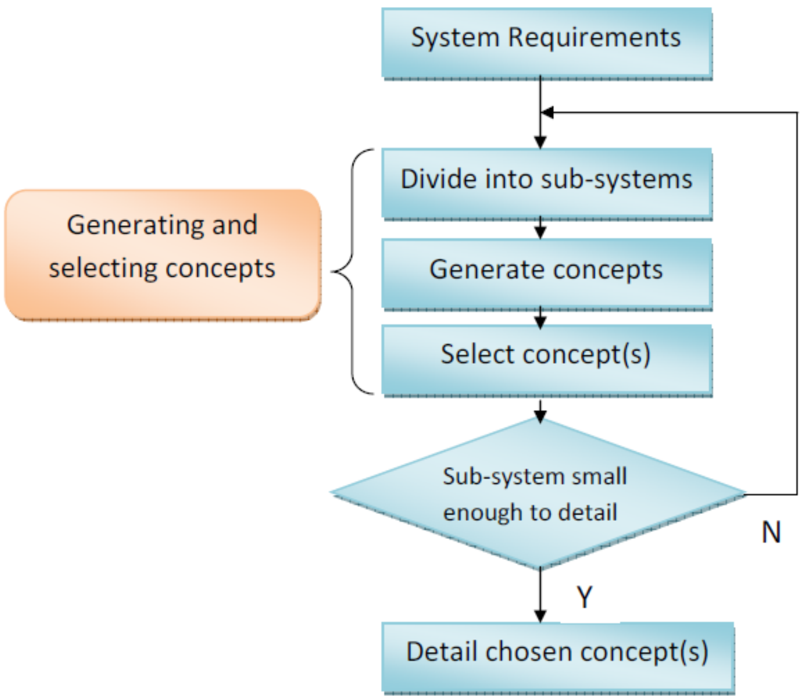


Figure 5.1: Functional Design phase – flow diagram

5.1 Analysis of requirements

This analysis must aim at proving the principal feasibility of the functional and technical requirements as formulated in the System Requirements phase. The analysis can be performed in several sub-sequent levels, i.e.:

- Geographical: analysis of how the system can be split up a system into sub-systems based on geographical locations
- Functional: analysis of how a system can be split up into sub-systems based on functionality, i.e. input, processing, output, sequence control, see figure below
- Technology: analysis of how a (sub-)system can be split up based on technology: i.e. mechanical ↔ electrical ↔ electronic ↔ software
- Technical: Analysis of what techniques can used

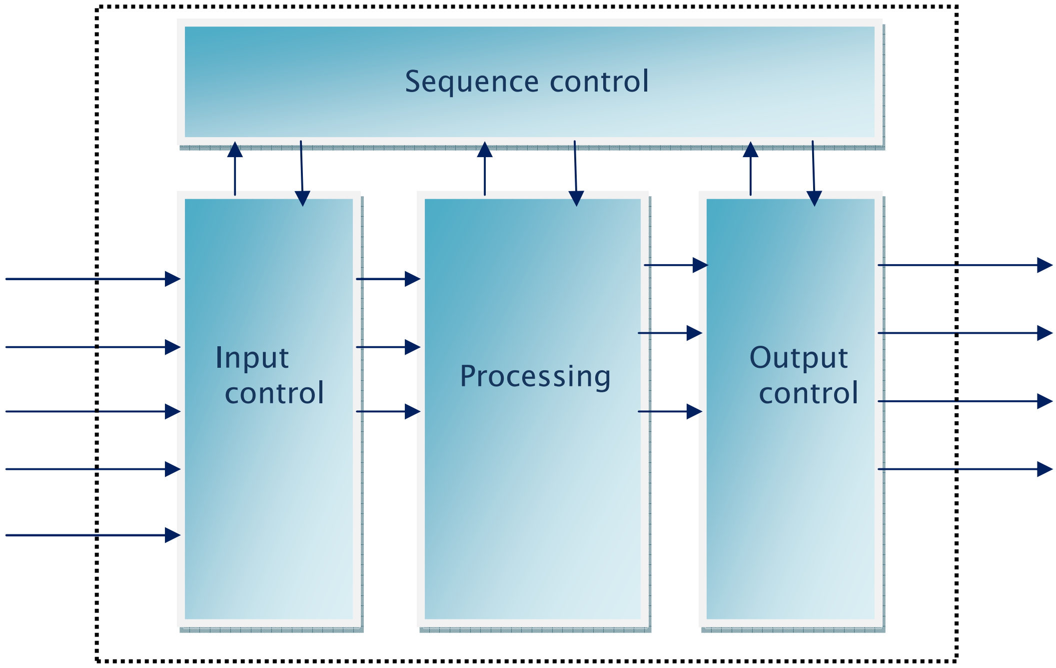


Figure 5.2: Example of splitting up a system in sub-systems

Each level of analysis starts with defining key-parameters for the system (or sub-system). For each key-parameter different concept-ideas should be generated.

Key parameters can be i.e. speed reduction, transportation, motor type, processing platform, topology. The figures below shows these key parameter with some concept-ideas. Usually sketches are made for each concept-idea.

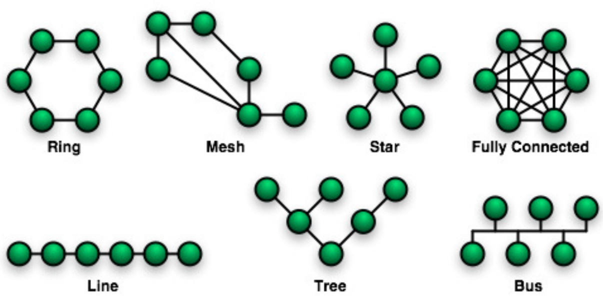


Figure 5.3: Topology (ring, mesh, star, fully connected, line, tree, bus)

Figure 5.4: Speed reduction (gear , chain, belt)

Figure 5.5: Transportation (wheel, omni-wheel, caterpillar, ball)

Figure 5.6: Proximity sensor (inductive, capacitive, ultrasonic/radar, infrared)

Figure 5.7: Motor type (linear, rotation DC, rotation AC, stepper, servo)

Figure 5.8: Processing platform (FPGA, microcontroller, PLC, PC)

5.2 Selecting a concept

After defining key parameters with concept-ideas a morphologic diagram [ref. 3] can be made.

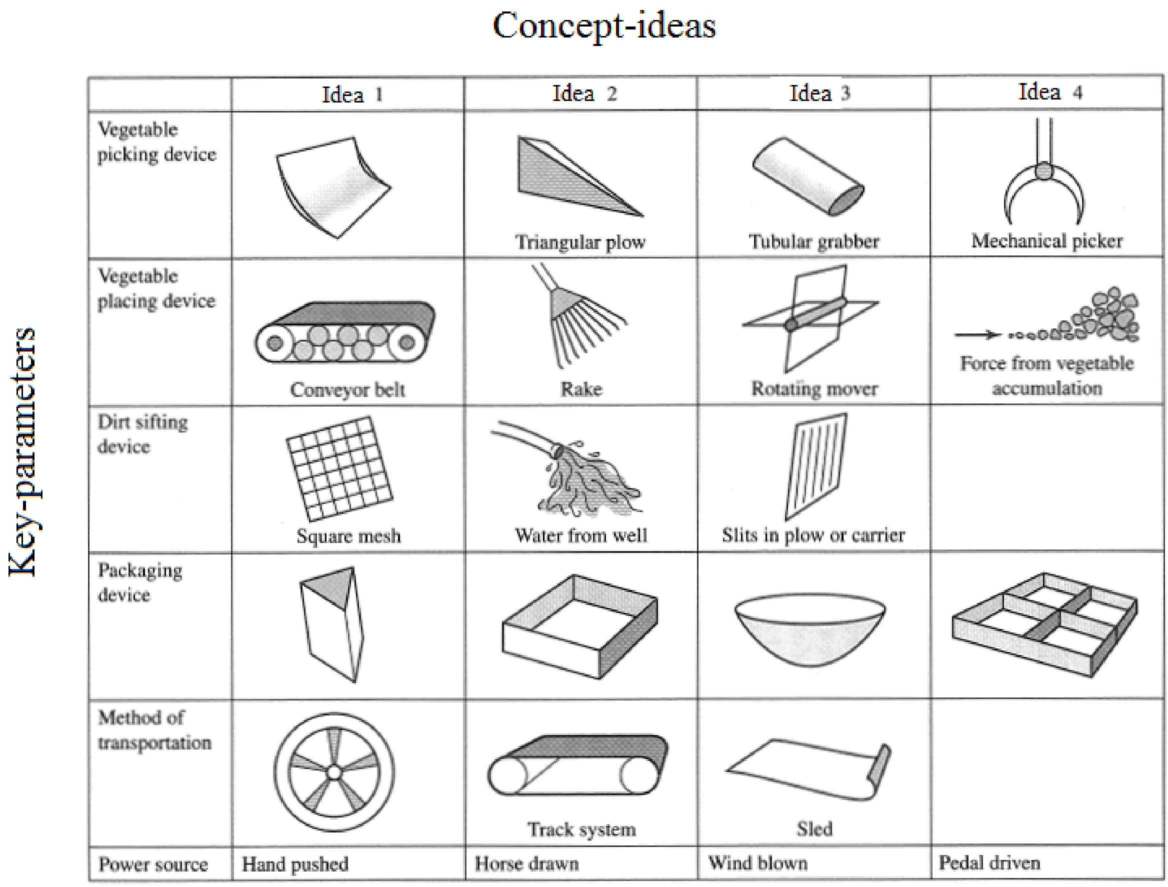


Figure 5.9: Morphologic diagram

Each key parameter with concept-ideas now has to be evaluated against functional and technical requirements. A good way to do this is to make for each key-parameter a table with concept-ideas and selection criteria. The most promising concept-idea now can be selected.

Key-parameter	criteria 1	criteria 2	criteria 3	criteria 4	criteria 5
	10%	20%	20%	30%	20%
concept-idea1	++	+	-	+	+
concept-idea 2	+	-	+	--	-
concept-idea 3	+	++	++	+	+

Figure 5.10: Weighed criteria

By indicating the most promising concept-idea for each key-parameter in the morphologic diagram a most promising concept for a (sub-)system can be selected.

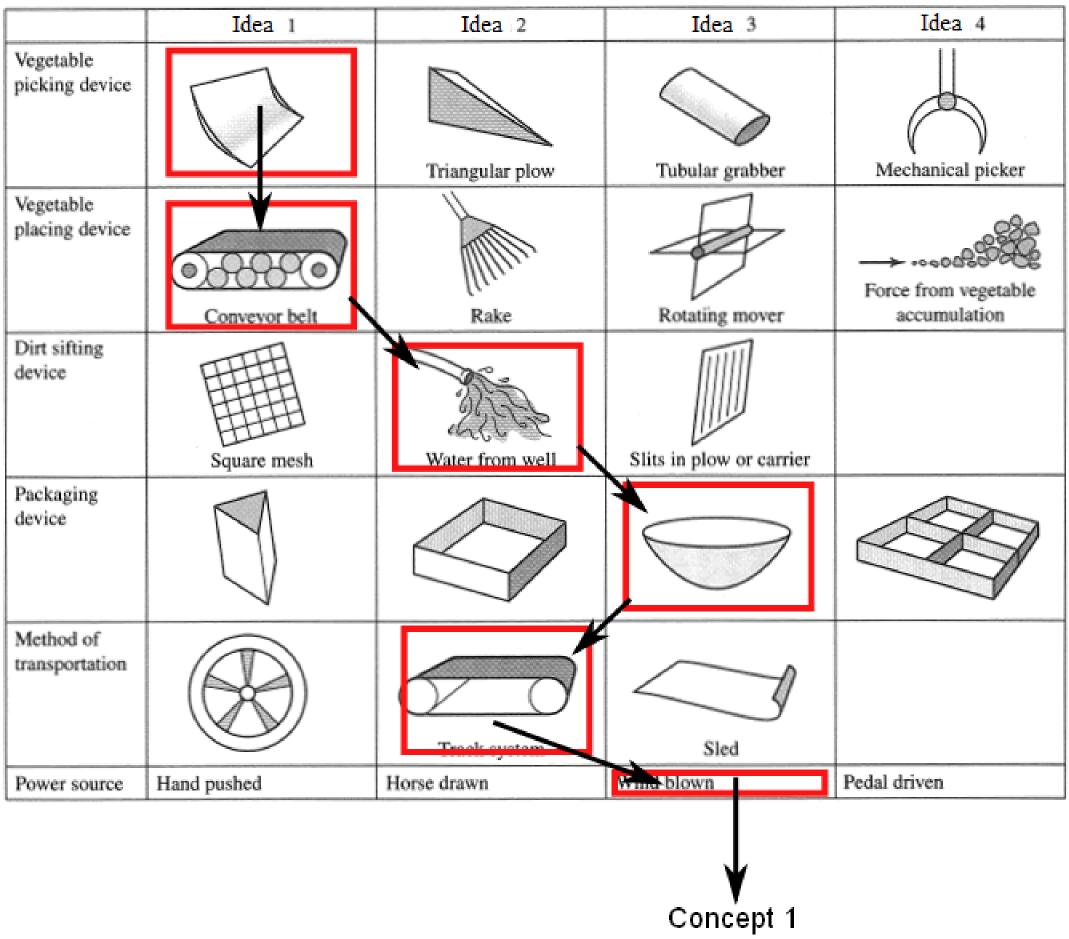


Figure 5.11: Choosing a concept

Probably there is no single best solution, without any disadvantages. So an argumentation to support your choice is required.

5.3 Elaboration of chosen concept into functional design

For the one or two most promising concept principles a functional design is to be made. This generally includes general overviews across the disciplines and elaborations of concepts and functional designs for each discipline:

- mechanical
- electrical
- electronic
- software

5.3.1 Overview of the functional design

Overviews:

- Technical block diagram, including all mechanical, electrical and electronic blocks.
- Safety philosophy, standards and measures.

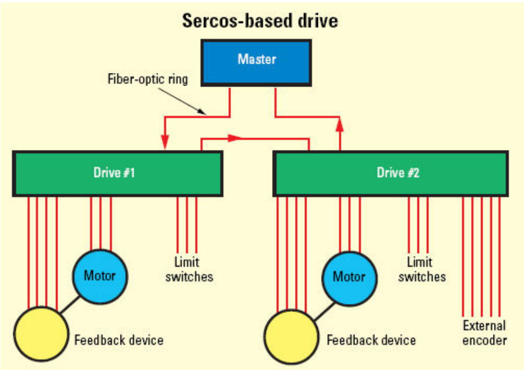


Figure 5.12: System architecture example-1

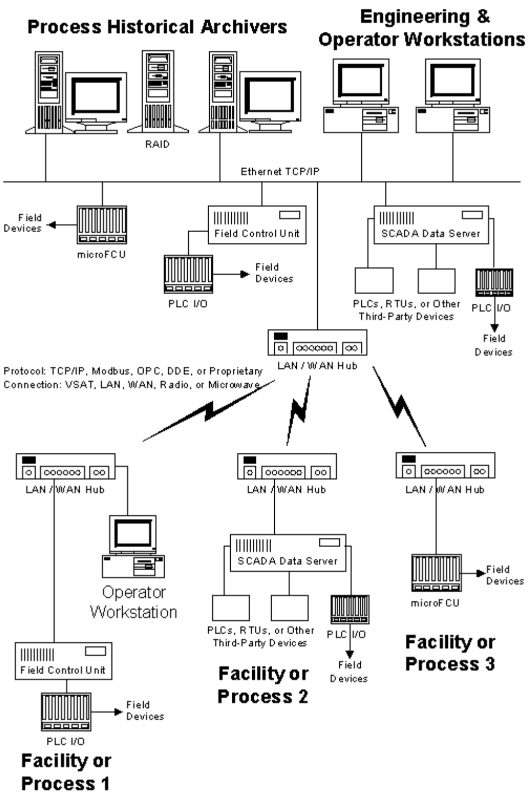


Figure 5.13: System architecture example-2

5.3.2 Mechanical concept

This starts by creating a mechanical layout; positioning and relations of major components. If necessary, dynamical and/or thermal calculations are to be done.

- Frames; mass, strength and stiffness, eigen-frequencies.
- Mechanisms, guidance's, etc.; stiffness's and disturbance forces.
- Drives; dimensioning, transmissions, flexibilities.

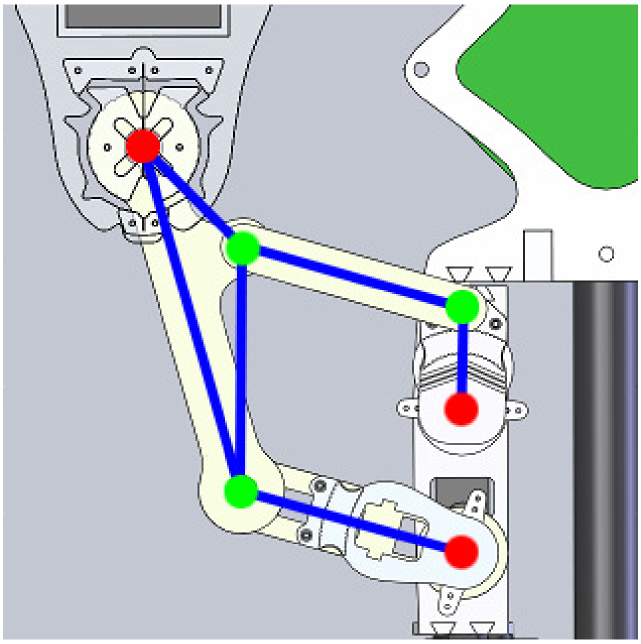


Figure 5.14: Mechanical concept example

5.3.3 Electrical functional diagram

This starts by creating an electrical layout; modules and interconnections on wiring level. The figure below gives an example for a medium-voltage architecture. Note that during this phase components are illustrated by its function (transformer, breaker, ...), no component types are selected during this phase.

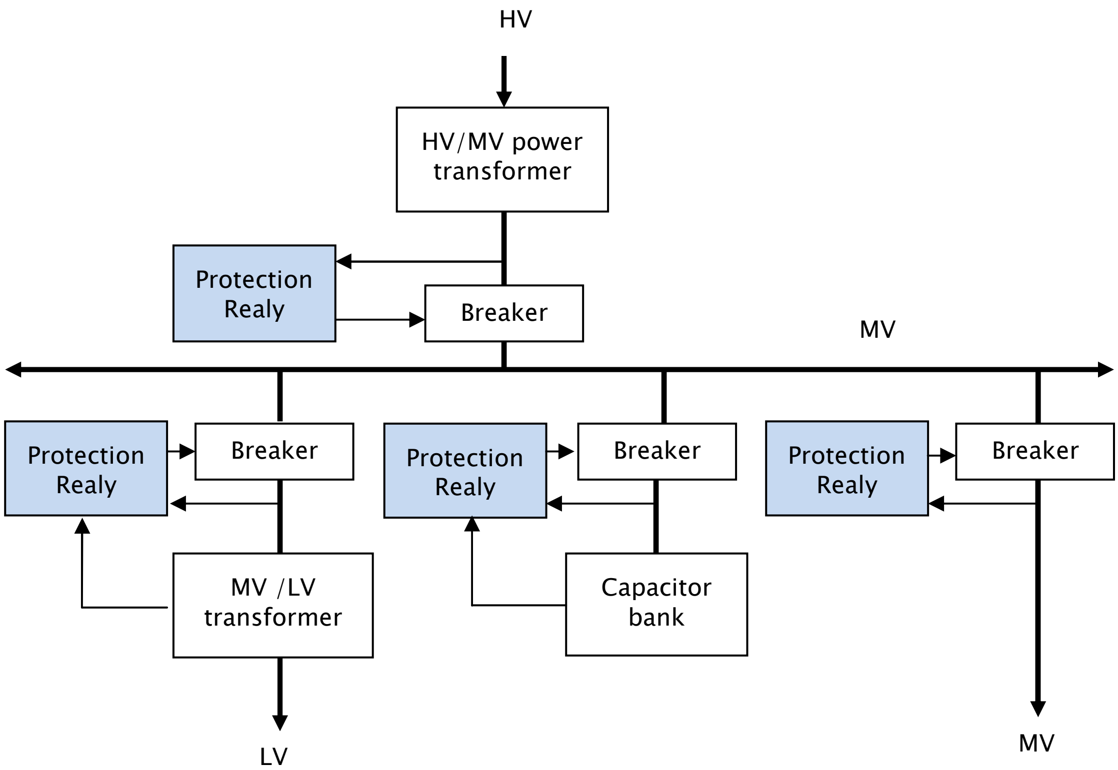


Figure 5.15: HV/MV/LV architecture example

Finally, for each Functional Block a short explanation should be given which covers the interfaces (inputs, outputs) and a functional description, i.e.:

Inputs

Name	Description	Relation	Value	Unit

Outputs

Name	Description	Relation	Value	Unit

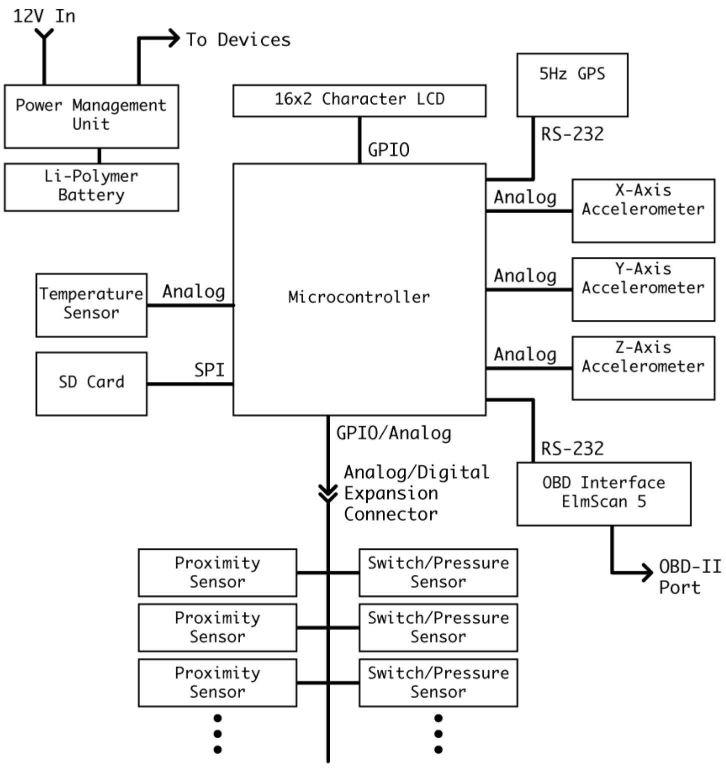
Functional Description

5.3.4 Electronic functional diagram

This starts by creating an electronic block-diagram. The figures below show some examples for electronic block diagrams. Note that in this phase components should not be selected and therefore not indicated in the diagrams!

Finally a detailed description for each block should be given, indicating:

- Inputs and outputs, AD/DA, resolution, sample rate.
- Measurement systems: location, resolution, accuracy.



-

Figure 5.16: Electronic architecture example-1

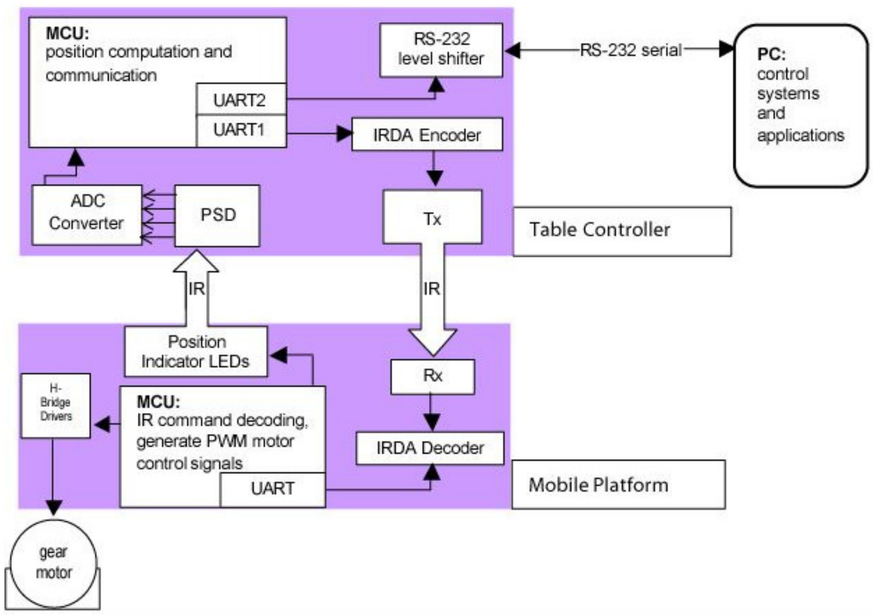


Figure 5.17: Electronic architecture example-2

5.3.5 Software functional diagram

This starts by creating a software architecture: software structure charts and block diagrams.

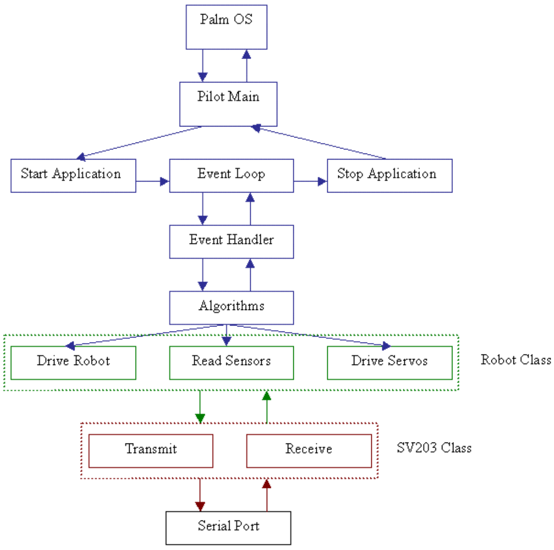


Figure 5.18: Software structure chart example

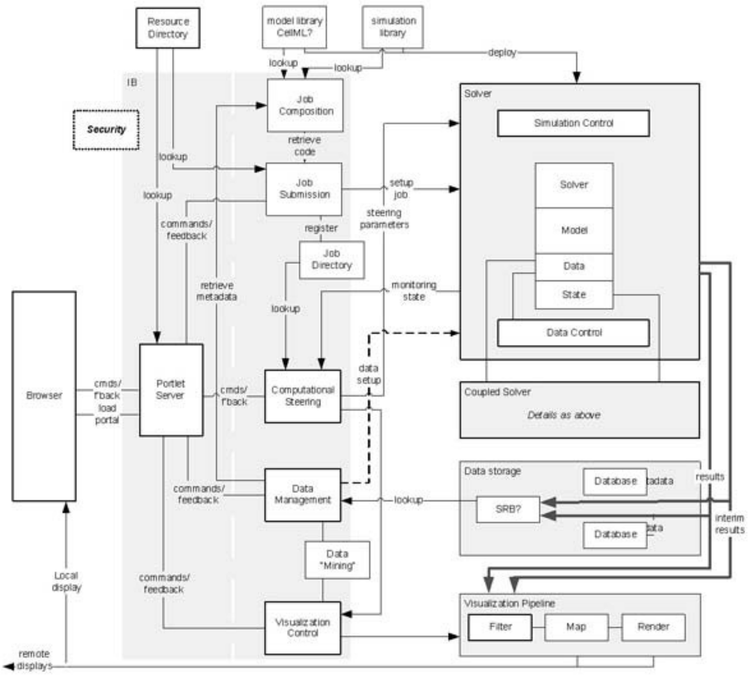


Figure 5.19: Software block diagram example

After that each block should be described individually:

- Algorithms for main function(s), e.g. controlled motion.
- Algorithms for sub-functions, e.g. start-up, self test.
- Architecture diagram with modules and interfaces.
- System states plus state transition diagram.
- Interrupts.
- Programming language and environment.

6. Technical Design phase

Goal of the Technical Design phase is to generate all schematics and drawings to make a direct realisation in the next phase possible. To get from the functional diagrams the final design the several steps can be performed (depending on the complexity of the unit), i.e.:

- Global schematics and drawings
- Simulation
- Calculation of essential components
- Selecting essential components
- Detailed schematics and drawings

The result should be that in the next phase special components can be made and units can be build. For the software programmers should be able to start programming directly.
In the following paragraphs some examples are given for:

- Mechanical
- Electrical
- Electronic
- Software

6.1 Mechanical

The result of this phase should be detailed 2D and 3D drawings of special components which has to be made and their assembly to units. An example is given in the figure below. Important are alse instructions for assembling the different components to a testable unit.

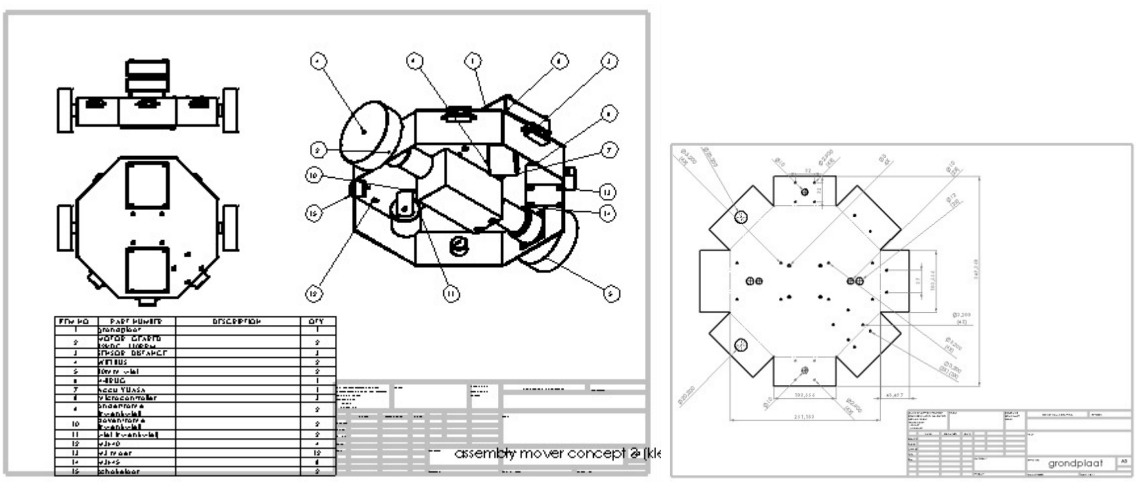


Figure 6.1: Detailed mechanical 2D and 3D drawings

6.2 Electrical

Detailed schematics are drawn and a component list with mounting instructions.

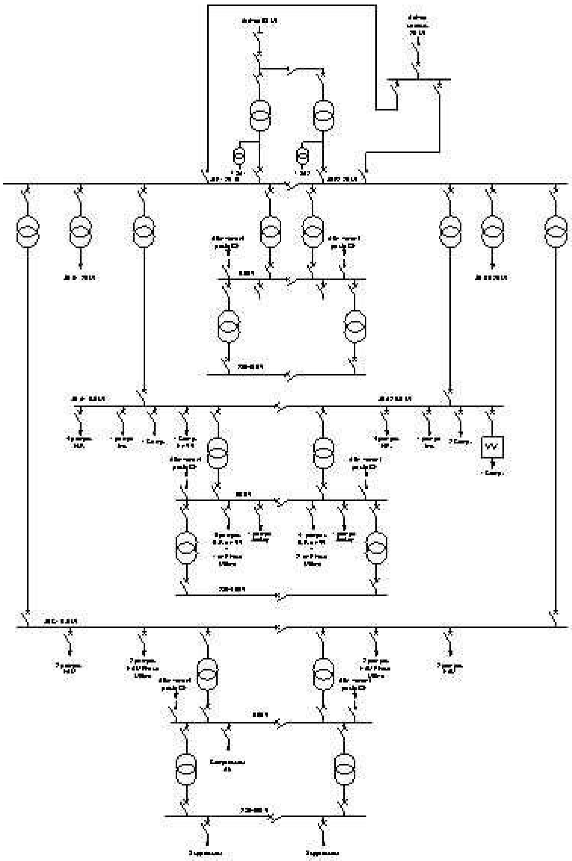


Figure 6.2: Electrical drawings

6.3 Electronic

The result of this phase should be:

- Schematics
- Drawings of special components, i.e. coils
- PCB designs with all files necessary for the manufacturing
- Component lists

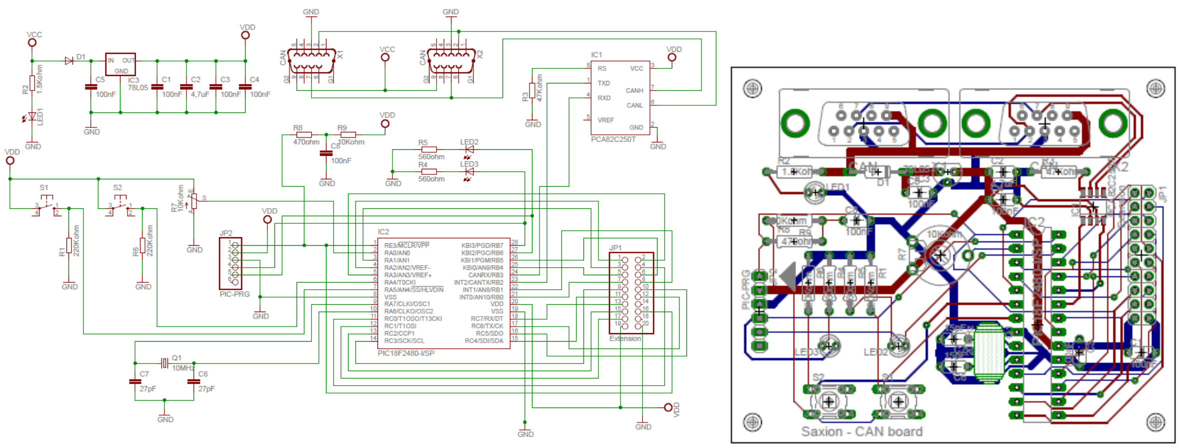
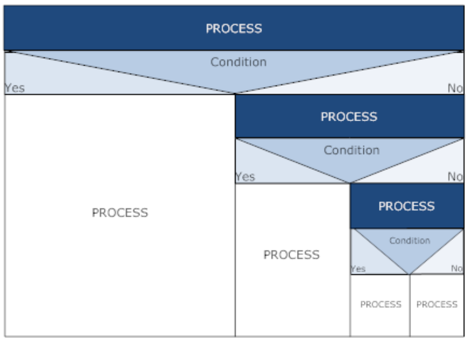


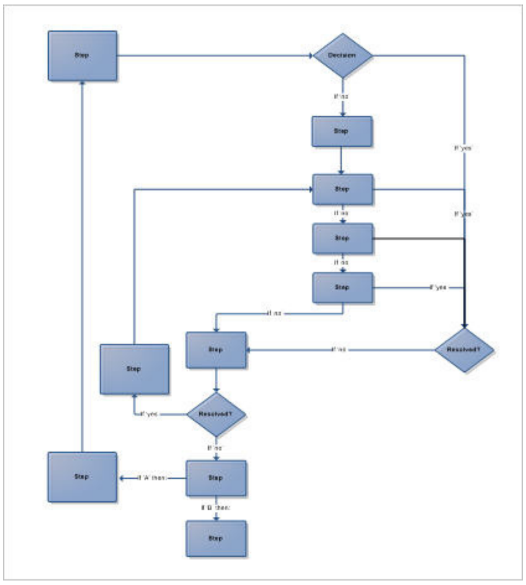
Figure 6.3: Schematic with PCB

6.4 Software

For the software the functionality of each block should be analysed and result in a description with which programmers directly can start coding. For this several techniques can be used, i.e.:

- Nassi-Shneidermann diagram
- Flowchart
- Pseudo code





Begin
  Until *each cell contains exactly one machine,* ***Do***
    Identify machines n1 and n2 such that $d_{n1,n2}$ *is the minimum.*
    Assign n1 and n2 to two different and empty cells.
    Discard machines n1 and n2 from the unassigned machines set.
    If *only one cell is remaining* ***then***
      Assign n1 to this cell
      Discard machine n1 from the unassigned machines set.
  End Until
  Until *unassigned machines set becomes empty,* ***Do***
    Identify machines n1 and n2 such that $d_{n1,n2}$ *is the maximum*
    Assign n1 and n2 to the same cell
  End Until
  Read *V (* interactively from the user *)*
  Add *V% dummy individual machines to each cell, such that the C cell sizes are equal*
End

Figure 6.4: Software description techniques

Notes

References

1 Forsberg, K. and Mooz, H., "The Relationship of Systems Engineering to the Project Cycle," First Annual Symposium of the National Council On Systems Engineering (NCOSE), October 1991

2 Roel Grit, “Project Management”,
Dutch edition: ISBN 978-90-01-79093-6, Noordhoff Uitgevers
English edition: ISBN 978-90-01-79092-9, Noordhoff Uitgevers

3 Prof.dr.ir. H.H. van den Kroonenberg and F.J. Siers, “Methodisch ontwerpen”,
ISBN 978-90-01-50901-9, Noordhoff Uitgevers

4 MPMM Project Management Lifecycle
http://www.mpmm.com/

Notes

A Project organisation

Theoretically, there is nothing particularly difficult about managing a project. Many projects, however, fail to attain their objectives. Since project work is very dependent on the people involved in it, those involved need to be aware of the fact that there is a difference between ordinary ways of approaching work and working in a project-based manner.

While working in project-groups it is also essential to plan and perform the work in a methodological way. Note that a methodological approach not only is essential when working in groups but also for projects that are done individually.

Within a project the members have clearly different tasks and responsibilities. We distinguish between main functions as those where the emphasis lies on the end responsibility, and on those where the tasks are actually carried out. It's important that these tasks are delegated early enough, so that the planned preparation work can be carried out by the appointed person. The main functions are explained below.

Besides the participation in the technical content of all the members of the team in the project, basically there are 4 assigned functions, namely:

- Project leader
- Documentation & Material Administrator
- Quality controller
- Designer

A number of tasks are linked to these functions, described in the following sections.

A.1 Project leader

The project leader has the final responsibility for the total result of the project group.

1. Setting up and monitoring of the planning. In this role it's important to make any necessary adjustments to the planning or undertake any action needed to ensure the project result.
2. Caring for optimal sharing of work and maintaining harmony among project members. This can be done by delegating appropriate tasks in consultation with the members and organizing group meetings.
3. Responsible for the work area and the issuing of tools.
4. Communication with the client
5. Preparation of the meetings.
6. Filling out the different forms in the project file.
7. Total direction with regard to the end result of the project.

A.2 Documentation and Material Administrator

In a project there are scores of documents and materials used and generated. The administrator must manage all the documents and materials that play a role in the project. Here the following are to be considered:

1. Be responsible for the consistent and functional layout of all documents (uniform style counts in the project) for internal and external communication.
2. Be responsible for all the documents being present in the file, including specifically the minutes, decision list and the time sheets.
3. Being responsible for all the documents.
4. Administration of the project file.
5. At least once a week update the hours spent on the project by each member of the group. Also noting specifically where the hours were spent. The different members are themselves responsible for handing in there hours and the way in which they were spent.
6. Handing in the documents for external communication on time.
7. Caring for the collecting and keeping of documentation
8. Keeping the component list up to date.

A.3 Quality Controller

The quality controller has as the main task to make sure that the products realized within the project fulfill the agreed criteria. Here the following are to be considered:

1. Setting up of agreed test procedures and the carrying out of tests of diverse specifications, designs and finished hardware and software products.
2. Being responsible for the quality of the material.
3. Monitoring of the content and quality of presentation of all documents, materials and presentations generated and used by the group for internal and external communication. Including: schedules, minutes, content and layout rundown, overhead sheets, etc.
4. Register and make a report on whether all the group members' tasks and commitments have been carried out accordingly.

A.4 Designer(s)

The designer is responsible for the technical quality for that part of the design for which he's responsible. Here the following are to be considered:

1. Considering possible solutions for the posed technical problems.
2. Formulating selection criteria.
3. Making of choices in technical areas.
4. Establishing content of the diverse specifications, such as the formal, functional and technical specifications.

It's essential that the designers are kept completely up to date with all the technical aspects of the design.

B Project Plan

Project title

Project plan

Version: 0.1
Date:

	Name	E-mail	phone
1			
2			
3			
4			
5			

Table of contents

Remark: This appendix is the basic document for a project plan. For each section, in italic text shortly is described what the content of the section should be.

The italic text of each section should be replaced by your own text.

1 Background information

The (future) members of the project team, the organization in which the project is being carried out, and the outside world will need background information about the project. One way of doing this is including the information – the "project environment" – in the project plan. The information should be such that even the uninitiated can form a picture of both the project and the organization in which the project is being carried out.

2 The project result

This section of the project plan describes the ultimate project outcome based on the goals and commission. This section of the project plan explains why the project is being carried out. It also answers the following question: "What is the final result after the project is finished?" So this chapter defines the project result.

3 Project activities

A project is likely to involve a number of activities. If you are carrying out a certain type of project for the first time, you are unlikely to be familiar with all of them. Do not hesitate to consult others who have carried out a similar project.

4 Project limits

What falls within the project's scope and what does not is frequently unclear. Sometimes it seems clear, but it later turns out the project group and the client each had their own interpretation. In order to prevent unclear situations from developing, you should describe the <u>*boundaries of the project*</u>*.*
Important boundaries are:
1. When does the project finish? (i.e. the "length" of the project).
2. What does the project include? (i.e. the "width" of the project).
Sometimes it is also good to describe what is excluded from the project.

Everything that is carried out as part of the project is called the "scope" of the project. The word "scope" literally means size, range and domain.

5 Products

This section describes the product or "deliverables" of the project. Your project group will carry out a number of activities. These will result in all sorts of products. All products (intermediate or otherwise) together give the 'project results' desired by the client. Since you are designing a product and doing so to a deadline, the project's progress is measurable. You can regard activities or a group of activities as milestones in the schedule and have deliverables directly related to the milestones.

6 Quality control

The section on quality control in the project plan deals with the quality of the end product (and all intermediate products) referred to in the previous section. How can you guarantee their quality?
Keep in mind that it is the client who ultimately decides whether the quality is sufficient. You can guarantee the quality in the following ways:

5. *Working/designing/developing in a methodological way and reporting on a regular basis.*
6. *Describe the quality of the intermediate deliverables and end product. Also describe how the client will assess this quality after the project has been completed.*
7. *The client will have to be reassured early on – during the project – about the quality of the project and the end results. You can do this in advance in your project plan by determining how you plan to assess the quality of each of the intermediate products while carrying out the project.*
8. *Indicate what "checks" will be carried out to guarantee quality. These could include both tests and technical procedures.*

7 Project Organisation

This section gives information about the members, clients and stakeholders involved in the project. Assign tasks and responsibilities to each member.

Member(s) in this project group

	Name	E-mail (Saxion)	phone
1			
2			
3			
4			
5			

<u>Member1</u>
Member1 is the project leader. His responsibilities are:
1
2

<u>Member2</u>
Member2 is the documentation manager. His responsibilities are:
1
2

<u>Member3</u>
Member3 is the quality manager. His responsibilities are
…

<u>Member4</u>
…

<u>Member5</u>
…

The client(s)/coach(es)

	Name	E-mail	
1			
2			

The responsibilities of the client(s)/coach(es) are:
1
2

Stakeholders

	Name	E-mail	
1			
2			

8 Schedule

Once you know what the project's activities are, what products are involved, what project's milestones will be, and who part of the project's team is, you can start drawing up the schedules. A schedule provides an overview of activities in relation to time. Eventually already members of the project team can be assigned to various activities. Note that in this phase of the project the activities are globally planned. When, during execution of the project, a certain activity has to be done a more detailed plan for this activity has to be set up.

Incorporate a global planning according to the phases mentioned in the V-model, following the chart below.

Activity	Duration (days)	Planning 9	10	11	12	13	14	15	16	17	18
1. Requirements Analysis and Project setup											
2. System Requirements											
3. Architecture Design											
4. Technical Design											
5. Realization											
6. Testing											
7. System Validation											

Milestones during the project are:

1. ...... dd/mm/yyyy
2. ...... dd/mm/yyyy
3. ......

.

.

x. Presentation dd/mm/yyyy

<u>Remark</u>

The planning has to be made with Microsoft Projects or GanttProject (freeware) and can be added as an appendix

9 Costs and benefits

Carrying out a project always takes time (labour hours), and therefore costs money. It also involves the use of resources (i.e. software, computers, test equipment, hardware, evaluation boards, etc.) and has to produce something: the yield or benefit. In this section the total estimated cost of the project and the benefits are given.

10 Risk analysis

A project's success might come under threat from all directions. Some of the risks a project runs are listed below. They are split up into two types of risks: internal and external.
The internal risks include:
Project not feasible, too many areas of expertise involved, too little experience in project based work, insufficient knowledge or expertise, are unwilling or unable to work together, project members are not sufficiently motivated.
The external risks include:
Too much dependence on other projects, the project's scope has not been clearly defined, insufficient corporation by employees of the organisation.

This section ends with recommendations to reduce the risks.

Notes