Part Four: General Project Management

6. Project Planing - Scope And Tools

6.1 Start Of A New Project - The Project Plan

You are assigned to be the new Project Leader. You've been introduce in the objectives of the new project, the time-frame, the budget you can spend. Management has shown you some of the most valuable team members, the office spaces and the IT systems currently in use for current projects (and - in parenthesis - you are expected to be knowledgeable in and to use too).

At that point, you are probably knowledgeable about your duties how to realise the *Project Management Cycle* (figure 19) for the actual project.

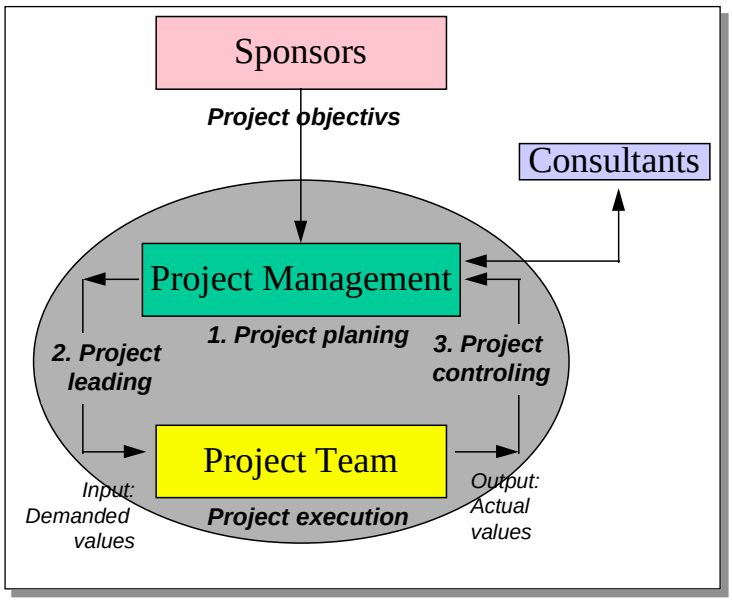

Figure 20: The Project Management Circle [~ WieczorrekMertens2007]

Depending on the assigned tasks and your experience, you may find yourself between the following extremes:

- You feel a little bit like 'Alice in Wonderland' and you begin to realise the difficulty of your new job.
- You can initially guess that the means, sponsors and management have assigned to you, are fairly inadequate to finish the project as desired.

Probably it somewhere in-between and you sit together with those people you are believe to be *Stakeholders* for this project and discuss the amount of work and perhaps the organisation of it.

As you know by experience, a project is nothing you can plan in detail and in advance, since it is driven by mostly inner forces. Thus, the first drafts) of the

- project's tasks,
- it's Work Breakdown Structure
- the it's organisation in terms of *Sub Project Leader* and *staffing*

are undoubtedly vague and probably unrealistic. There is still no good understanding, what efforts in term of time, labour force, and money needs to be spend for the identified legs. In addition, any risk assignment is still impossible. On the other side, a lot of people expecting already right now some answers from you about the project.

While you need some time to digest your impressions, already now you are involved in organisational items, regarding Computer Accounts, required file spaces for project documentation, allowed absences of the project secretary, and others.

At that point, any project has a certain amount of entropy with lots of open bits and pieces regarding scope, organisation, and time-frame. Thus, the first step as project manager is to reduce the entropy regarding the team members, and by the same token in your mind. In short, the first step is to *plan the project plan*. Even this very initial phase requires internal and perhaps external expertise. Thus in combination with the foreseen planing step it is necessary to dedicate man power here, as laid out in figure 21.

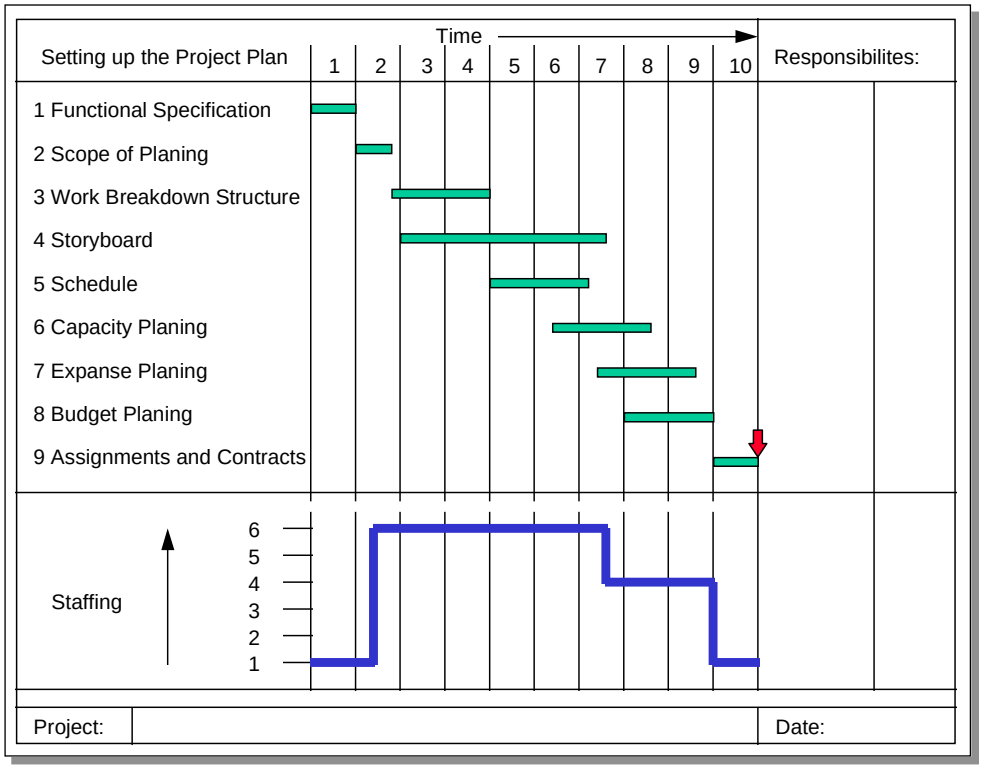

Figure 21: Setting up a project plan [~ Litke2007]

Depending on the size of the project, this initial planing phase spans weeks or perhaps months and may involve much more man power as can be depict from the above figure. However, depending on the progress of the project, already some of the evaluations have already be carried out even before the final project manager has been assigned. Thus, in that case the PL has to execute the task and participation is reduced to later adjustments.

At the end of the initial step a *Project Plan* has been established. This initial Project Plan (or *Masterplan*) will be accompanied by individual and more detailed phase plans. While the project walks through the respective phases, the Project Plan is subject for subtle or more severe corrections. Thus, the Project Plan has to revised by Project Management and revisions have to be streamlined with the SPLs. The whole plan need confirmation by upper management and finally will be brought to attention to the project members. Figure 22 show a typical evolution of a Master Plan.

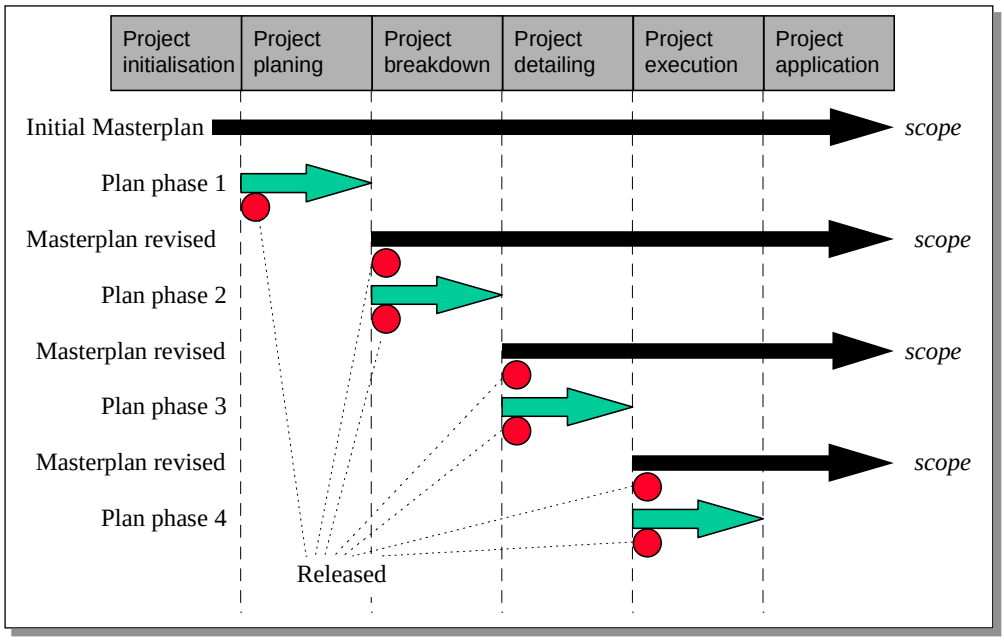

Figure 22: Evolution of the Project Plan during the project's execution [~ Litke2007]

6.2 Identifying Project Dependencies And Tasks

Any current IT project has to obey a lot of dependencies:

- Internal (given by the project itself):
  - project definition (functional description, deadline, quality),
  - realisation (team. methods),
  - budget constraints (costs, ROI)
- External (the resulting outcome of the project has to be placed in the real world):
  - compatibility (interfaces),
  - acceptance (customers), and or course
  - competition (price, market share).

Without a good understanding of both aspects, a project runs on undefined risk which may impact the project significantly. Project management is responsible to identify at least the internal dependencies, to express them explicitly against upper management and sponsors, andconsidering these in the project plan. Figure 22 provides in a bird's view the (mostly internal) dependencies to be considered.

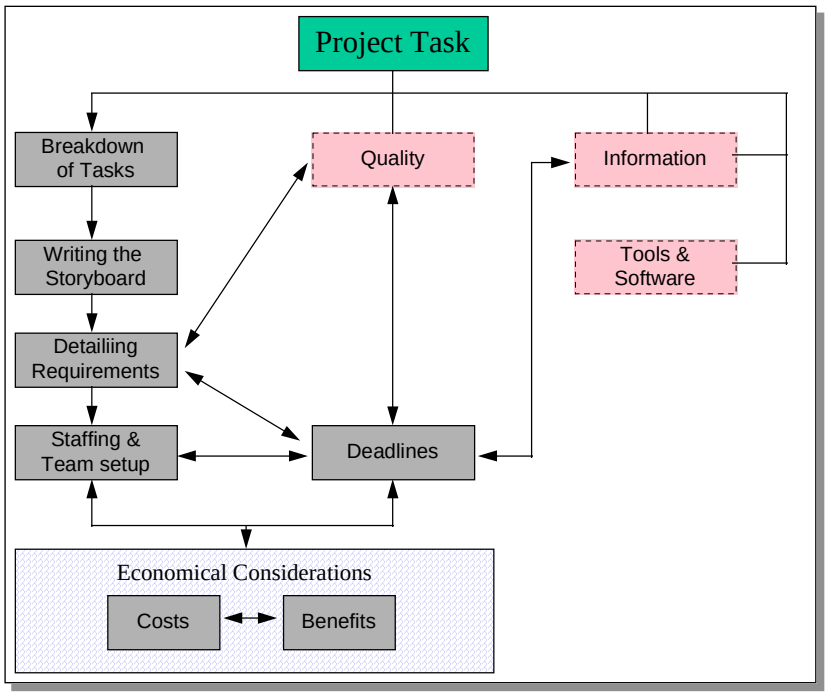

Figure 22: Internal and external dependencies of a project [~ Litke2007]

6.3 The Work Breakdown Approach

Once the dependencies are identified, it is time to start working on the breakdown of tasks and to establish the *Work Breakdown Structure* (*WBS*) base on the project's functional and/or technical description.

The WBS is effectively a breakdown of the *System Structure* and allows to

- determine and define the particular functional parts of the system,
- build (confined) subprojects according to the functional parts,
- identify dependencies and required interfaces between the parts, and perhaps
- allows to assess efforts and risks to the individual parts.

Typically, the WBS is a top-down approach starting from the general specifications and requirements to the necessary level of detail: The *structural elements*. As with most plans, the WBS will evolve during the project's lifetime. In particular, in the beginning, not all functional parts can be clearly be identified.

The dependencies and the compatibility of the individual structural elements have to be determined as well; typically while using a matrix approach. For IT software projects it might be possible now to guess the program structure, the interfaces, perhaps common subroutines and functions and also the required program classes. In fact, IT projects typically follow a object-oriented top down-approach.

Figure 23 shows a top-down approach in order to evaluate the WBS.

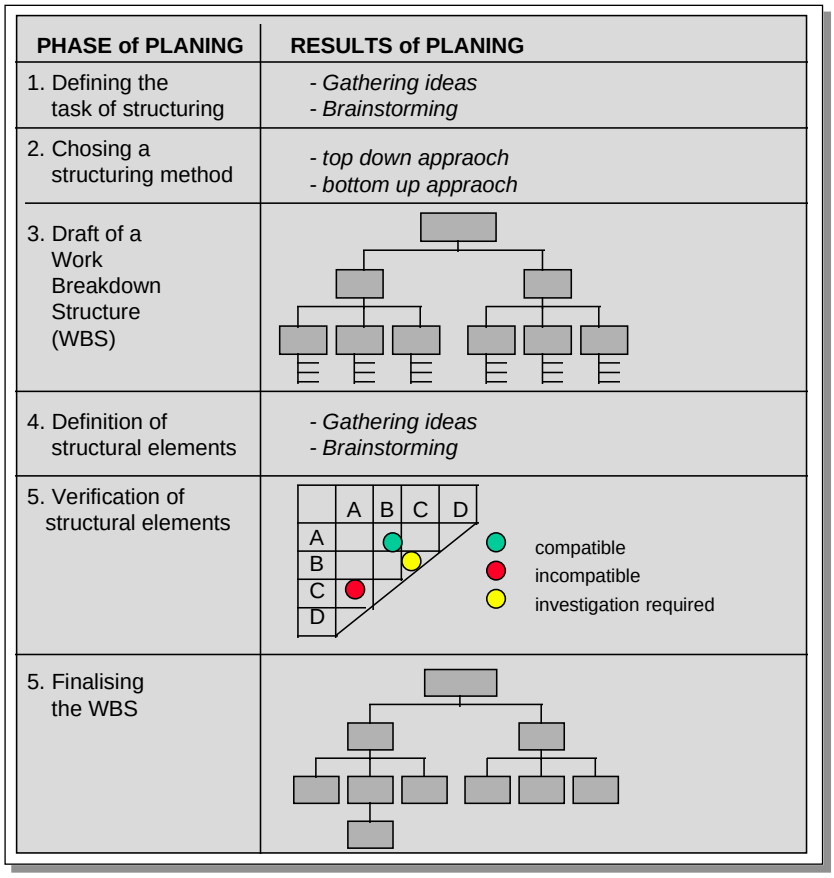

Figure 23: Top down approach for setting up the Work Breakdown Structure [~ Litke2007]

After the WBS has been established initially, it is necessary to define particular tasks which can be assigned to individual project members, groups, or perhaps external providers:

- By construction, every task is self-confined and can be completed independently from each other (except for streamlining the interfaces).
- The task should be considered as 'black box' [`A -> task -> A'`]:
  - A defined input [A] is provided to the [task],
  - the [task] has to react upon receipt in a deterministic manner as defined in the corresponding *use-case*, and
  - the [task] produces a well-defined output [A'] .
- From the point of the assigned person or group, a task is a *sub-project* and requires a comparable level of project management, in particular regarding efforts, deadlines, and quality.

Tasks packages can be assigned to *Systems* or *Subsystems* within the WBS as detailed in figure 24.

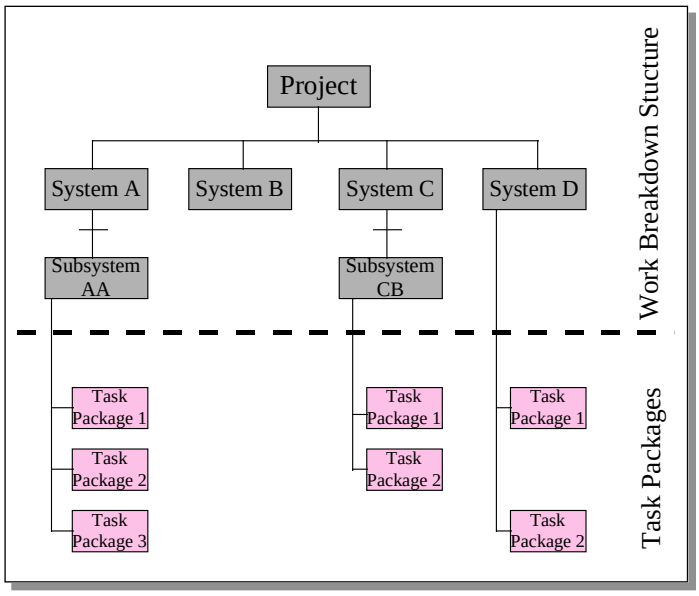

Figure 24: Assigning task packages after the breakdown of the project in a WBS [~ Litke2007]

6.4 Project Organisation Structure

In parallel with the WBS breakdown, the organisation structure of the project can now be established. Now, the staff members can be organised in groups while it is possible to estimate (at first glance) the size and the qualification of groups according to the WBS (figure 25).

As a result, not only a (final) organigram and thus the resulting reporting chains is derived, but rather the assignment of the individual groups with tasks becomes transparent and perhaps allows to estimate possible incompleteness of tasks and staffing. This diagram shows in addition the *Level of Responsibility* for the individual groups and their co-operation in the whole project.

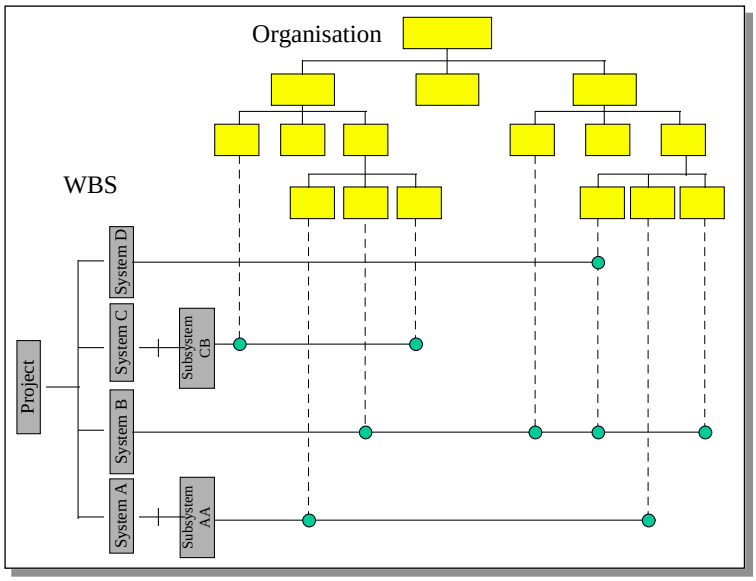

Figure 25: Correlation of task with organisation [~ Litke2007]

6.2 Scheduling: Assigning Deadlines And Resources To Tasks

Once evaluated, the WBS provides you with the layout or your project w.r.t. systems, subsystems, and tasks (as structural element). The *top-down* approach for the WBS is now complemented with a *bottom-up* approach to assign

- *Deadlines*
- *Resources*

to each activity which is known as *schedule management*.

6.2.1 Gantt Charts

This particular 'view' is typically visualised by a Gantt chart (named after the inventory Henry L. Gantt). Here,

- the duration of the individual activities are displayed as *bars*
- which have a related start and end date and showing their (linear) dependencies.
- Steps can be combined to tasks, sections or phases.
- The level of completion or critical steps can be easily visualised.

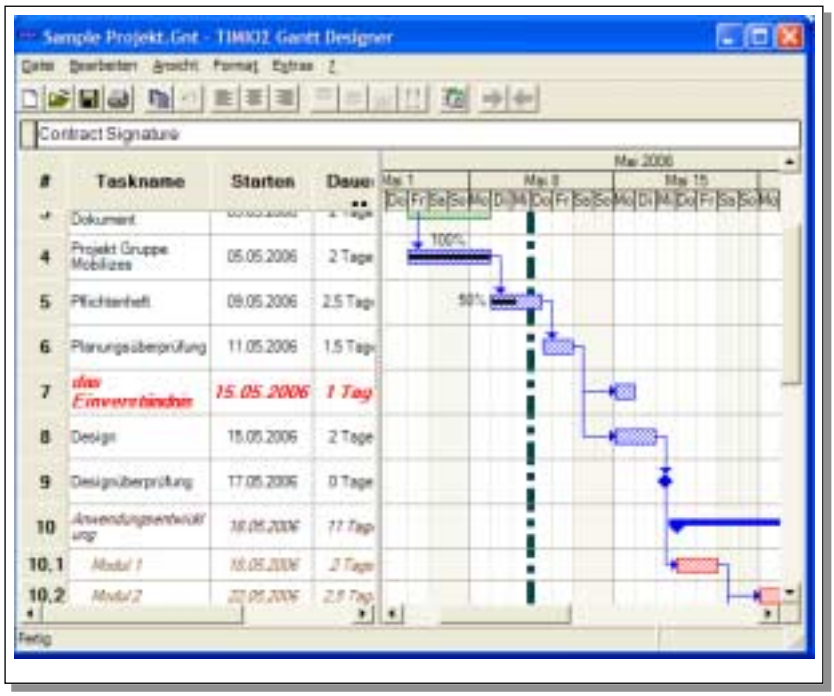

Figure 26: A snapshot view of the freeware Gantt program TIMIOS [http://timios.net/Gantt/]

Typical computer programs to provide Gantt charts techniques, allow in addition to define

- common human resources (availability's) and
- availble budgets for the whole project,
- which can be shared among the individual activities, according to definable keys (percentages).

A mathematical breakdown of those assigned numbers yields an estimate of

- the level of completeness regarding activities, section, and phase level,
- the used resources and budgets with the same level of detail.

Regarding the very advantages of Gantt charts (in particular using dedicated software, like Microsoft's Project), the disadvantages of Gantt charts result in the facts:

- Gantt charts do not show the functional dependencies (compared to the WBS).
- Larger projects result in very large Gantt diagrams and are hard to read.

6.2.2 Lists

The calculation within a Gantt chart is based on a spread-sheet; just the representation of the results follows the 'bar' scheme, known as Gantt chart or diagram.

However, the basic calculation involved can be realised with basic spread-sheet means, or can be expressed as chained lists.

Activity	Name	Predecessor	Successor	Duration [Days]	Start Date	Due Date
1	Definition		2	2	2.2.2009	8.2.2009
2	Teambuilding	1	3	10	9.2.2009	28.2.2009
3	WBS	2	4			
4						

Those simple spread sheets can be enhanced with inter-related sheets, where resources are defined can correlated with the individual activities.

Though, very easy to use and to adopt to a project, the information here is very limited and only suited for smaller projects.

6.2.3 Netplan Techniques

The Netplan technique is defined in DIN 69900 with the following scopes:

> *"The netplan technique includes procedures for project planing and control. The netplan is a graphical representation of flow-stuctures, in order to visualise logical and temporal dependencies of activities."*

There exist three types of the Netplan approach whether the activities are describes as vectors or nodes, or whether instead of activities events are meshed in nodes. Figure 27 shows an activity node netplan. Here, one node displays additionally resources and duration of the respective activity.

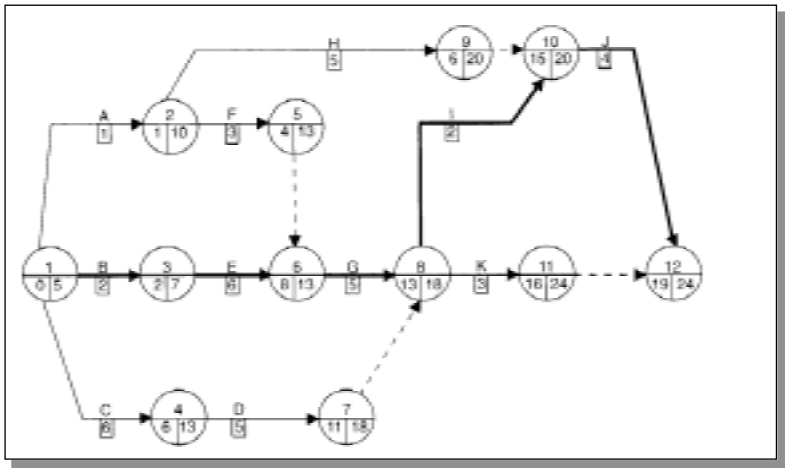

Figure 27: Sample for a Netplan with Acitivity-Node meshing [Litke2007]

6.2.4 Critical Path Analysis (CPA)

Once we have identified the resources required to achieve or complete a certain project task, we can gauge this task against all others.

Gauging means, we have to consider

- the respective resources required, and
- the effective dependencies

for every task.

In a well-behaving WBS, dependencies can be expressed simply as number of interfaces with respect to it's completion schedule. Figure 28 lists five milestones (10 to 50) and six activities (A to F). The 'criticality' is provided in term of completion time (t). Thus the paths B + C are most critical for the project. This diagram is a so-called ***PERT*** chart (*Program Evaluation and Review Technique*), which is equivalent to the Netplan 'activity-vector' approach.

As part of the risk management, PM has to have at any time a good understanding of those tasks which are considered 'critical' and shall watch those very closely.

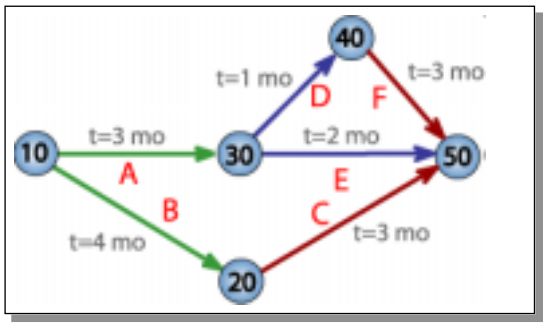

Figure 28: Critical Path Diagram [Wikipedia]

6.2.5 Summary of Project Planing

In this chapter we have essentially defined the

- [First step of Project Management] **Planning the project**.

The planning phase can be broken into four different sections:

1. Setting up a team which actually does the planning.
2. Creating a *Work Breakdown Structure* which describes the functional groups (broken down in systems, subsystems, and tasks) of the project and allows an estimate of dependencies and required efforts.
3. Creating a *Schedule* which tells, when the individual task have to be completed and in addition shows their execution dependencies.
4. *Assigning Resources* to the tasks in terms of man-power and budget.

At the end of this step, a *Masterplan* for the project has been established which includes the above mentioned ingredients. Potentially, already now it becomes apparent which organisational changes to the project are essential, thus which teams need to be set-up initially and which long-lasting SPLs need to be assigned.

7. The Master Plan - Phase Management

According to our today's understanding, a projects evolution (and perhaps progress) can be expressed in phases with five phases under responsibility of the Project Management. Figure 30 explicitly makes a reference to an IT software project.

- Initialisation (Project idea)
- Evaluation
- Draft
- Detailed Study
- Systemdesign
- Project finishing

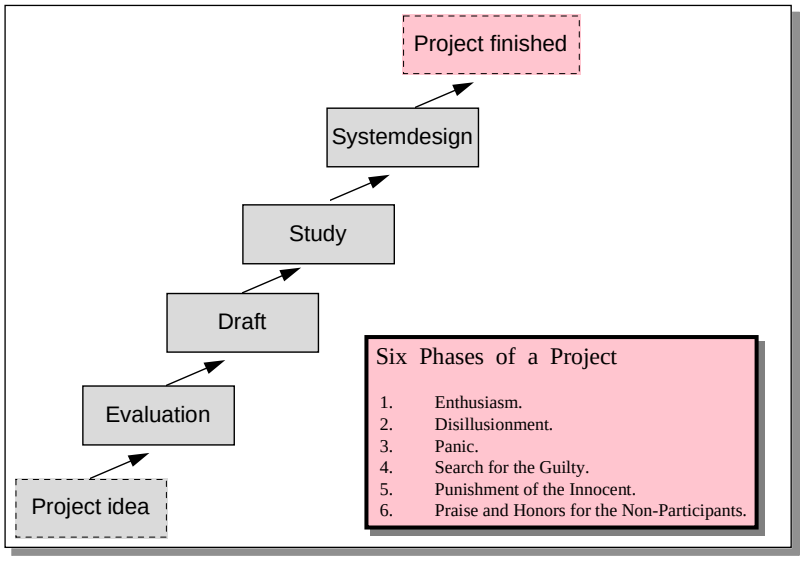

Figure 30: The phases of a IT project - official and alternative

7.1 Software Lifecycle Model

Considering the construction of special tailored software, it became soon aware that software never is delivered as expected by the sponsors:

- During the project evolution, the requirements of the software has changed; in particular because the deficiency of the original approach became apparent.
- The functional aspects of design were not met, thus, additional development is necessary.
- The quality of the software is below expectation and during operation an uncontrolled (and perhaps unrecoverable) behaviour is been observed (commonly referred to as 'bugs').

Software evolves in time, which is know as S*oftware Lifecycle Model* (figure 31):

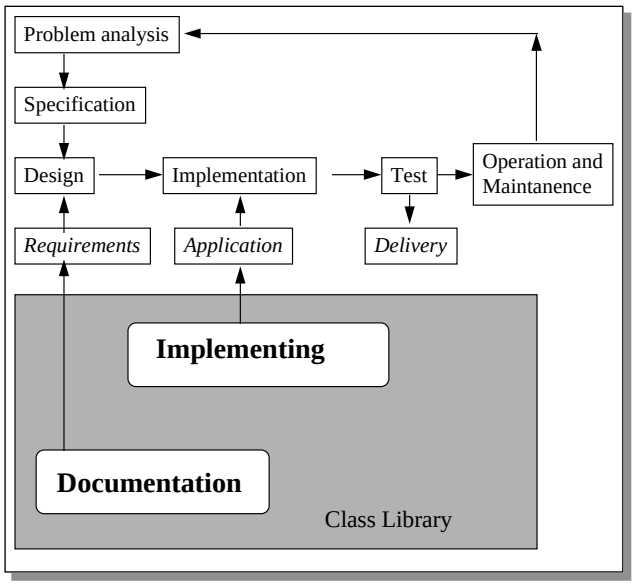

Figure 31: Software Lifecycle Model [Wahlmüller]

7.2 Spiral Model

More complex IT projects are not just focused on software development. Rather the project's tasks may include:

- Adoption and configuration of standard software.
- Considerable amount of special tailored software.
- Configuration of hardware, operating system, and middleware.
- Roll-Out of hard- and software.
- Introduction of the new solution into operation.
- Customisation of the current environment to include the new solution.
- Education of the technical staff to become familiar with the new system.
- Promoting the new solution to external customers.
- Providing a full documentation for the integrated solution.

Thus, today's IT projects require a merge of

- special software development methods and
- adopt classical project management approaches.

One way to combine both worlds is to break down the software specific methods in a *Milestone approach* in a so-called *Spiral model*:

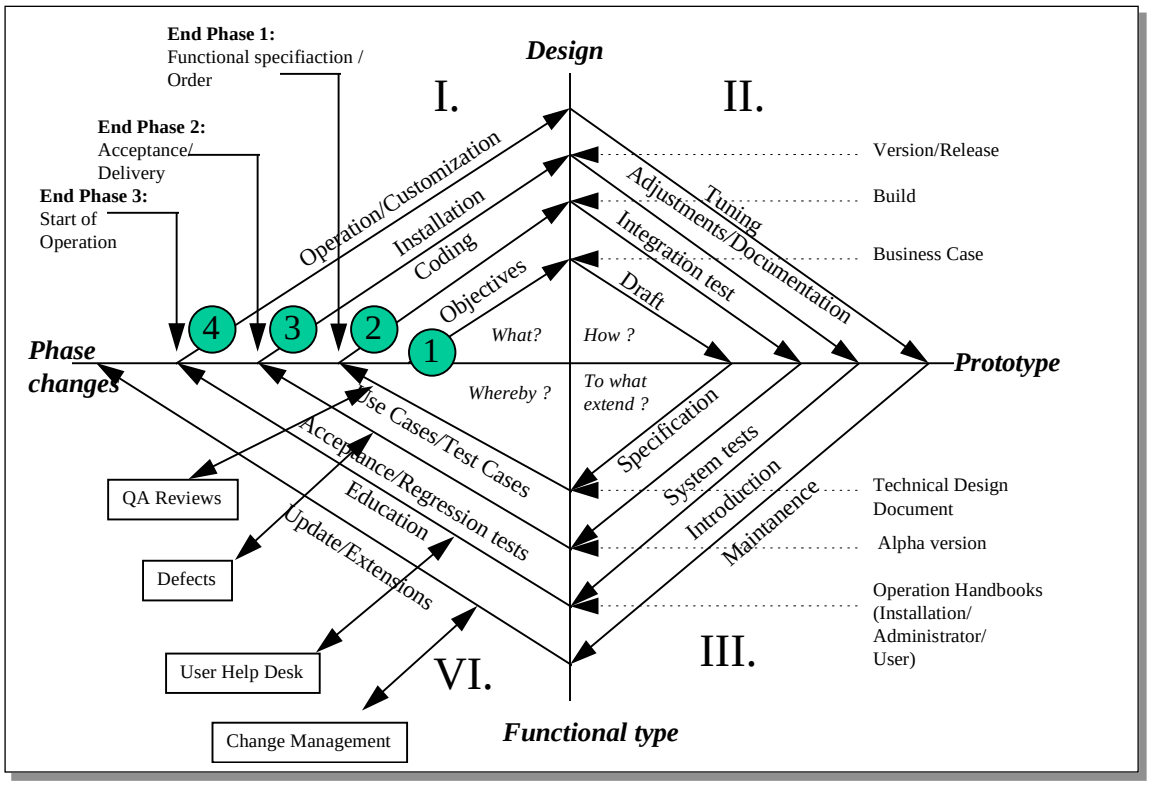

Figure 32: Spiral model for complex IT projects [Hoffmann1999]

The spiral model is essentially a phase model and describes the software development in cycles, while determing the four essential phases:

- Design
- Development
- Functional testing
- Operation

Important here is, that the phase changes are well determined in terms of milestones to be achieved. Apart for development issues, the model also includes operational aspects. The four quadrants of the spiral model indicate in particular the basic questions needed to be raised:

- What needs to be done ?
- How do we achieve the results ?
- To what extend do meet our objectives ?
- What are the relevant means ?

The milestones are additionally clearly described:

- Start is the Business Case
- Next step is the technical design document
- The build of the first (working) version, tells that coding is on it's way
- The first alpha version will be delivered to QA, thus the product should be mainly completed.
- Now, the software enters the usual release/development cycle

- And finally, not only the software but also the documents are in place for production.

Unlike the WBS or the Gantt chart, the spiral model does not try to assess the logical structure of the project, nor tries it to pinpoint the project's current state or resource exhaustion. Rather, it tells the Project Manager what is in place and what is missing, independently from the WBS. In this respect, the spiral model provides a framework for software development and can be used as a *ruler* for IT project management.