Systems Engineering practices

Now it's time to look at some practices which Systems Engineers face daily. How do we make documentation of projects? How do we apply concurrent engineering to projects? And how do we manage projects in general?

1 Documentation

1.1 Contents of documentation

Let's suppose we're creating a design, a product, or anything similar. It is our task to also create documentation. **Documentation** is a compilation of documents, giving information about the design/product. The documentation should include things like the requirements, the specifications, the planning, and so on.

You may wonder, what should we include in our documentation? The basic elements of documentation are...

- Relevance of the documentation/applicability (for example serial numbers)
- References used to build it
- References needed to use it
- Author/preparing entity (can also be a program)
- Checked by ...
- Released by ...
- Version (draft number)
- Date of preparation
- Amendments

Often computer programs are used to manage the documentation. If this is the case, we're one step closer to Knowledge Based Engineering. In fact, **Knowledge Based Engineering** (KBE) is defined as the use of advanced software techniques to capture and reuse product and process knowledge.

1.2 Structuring documentation

There should of course be a certain structure in our documentation. This is where Configuration Items come in handy. **Configuration Items** (CI's) are entities that are worthwhile to be controlled separately. A CI has a charecteristic...

- **Fit:** The physical interface of the CI to other items. (If the CI changes, other items may need to change as well.)
- **Form:** The (geometrical) shape of the CI.
- **Function:** The actions the CI is designed to perform.
- **Flow:** The origin of the CI. (Where the part comes from.)

1.3 The Object-Oriented design paradigm

The **Object-Oriented** (OO) design paradigm is a way of thinking about problems. It can also be used to structure documentation in a nice way. In the OO paradigm, we use **objects** to describe entities. These objects both incorporate the **attributes** of the item, as well as its **behaviour**.

We can classify multiple objects into a **class**. This is called **classification**. (For example, both the car that's parked in front of my house and the car that's just passing by are objects of the class **car**.) An object is said to be an **instance** of its class. Each instance of the class has the same attributes. (For example, all cars have an attribute 'brand'.) However, each instance does have different values for their attributes. (For example, car A can have as brand 'BMW', while car B has as brand 'Kia'.)

Inheritance is the sharing of attributes and operations among classes. It is based on a hierarchical relationship. (For example, both a car and an airplane are vehicles. And both the car, the airplane and the vehicle have an attribute 'weight'.) **Polymorphism** means that the same operation may behave differently on different classes. (For example, both a car and an airplane have an operation called 'open door'. However, this operation is quite different for the two vehicle types.)

2 Concurrent Engineering

2.1 What is Concurrent Engineering?

Concurrent Engineering (CE) is defined as the concurrent running of separate phases during the product definition trajectory. In other words, multiple things are done at the same time. Concurrent Engineering thus employes as much **parallel processing** as possible.

When applying Concurrent Engineering, it is important to have correct and complete **information**. If this is not available, parts might not fit during assembly. There should also be effective **communication**. Or the customer might wind up with a product he didn't ask for. The **responsibilities** should also be clear and well-documented.

2.2 The downsides of Concurrent Engineering

Concurrent Engineering can save a lot of time. But it is not without downsides. There are risks attachted to it. In CE, you start on one process, while not having finished the process before it. (Partial results are often used.) This may lead to necessary rework.

To reduce these risks, good communication is necessary. Also, often use is made of **Design/Build Teams** (DBT). These teams both participate in the design and the build phase of the project.

2.3 The Integrated Product Model

Managing information is very important, when applying CE. When using an **Integrated Product Model**, all information is located in one single source. This has several advantages. One of them is that the information can be controlled by a **Product Data Management** (PDM) system. This system guards the data and manages modifications applied to it.

Inside the PDM system, information often has a **status**. This status gives information on the stage the information is in. Is it a concept? Or is it already a final version? Can it be used for later parts in the project? The PDM system also keeps track of who needs to be notified, when information changes.

3 Project Management

3.1 What is Project Management?

Let's suppose we're working on a project. (A **project** is defined as a coherent set of activities meant to realize a defined outcome.) This project needs to be managed. This task is called Project Management.

In fact, **Project Management** (PM) is defined as a formal management discipline whereby projects are planned and executed according to a systematic, repeatable, and scaleable process. Project Management is an important task. The quality of Project Management can make or break a project.

Project Management consists of three important phases. Before the project has even started, it should already be **defined**. After this, the actual **project planning** is set up. The third phase is when the project is running. It then needs to be **controlled**. A general overview of the Project Management tasks can be seen in figure 1. In the next few paragraphs, we'll go a bit more into depth concerning those tasks.

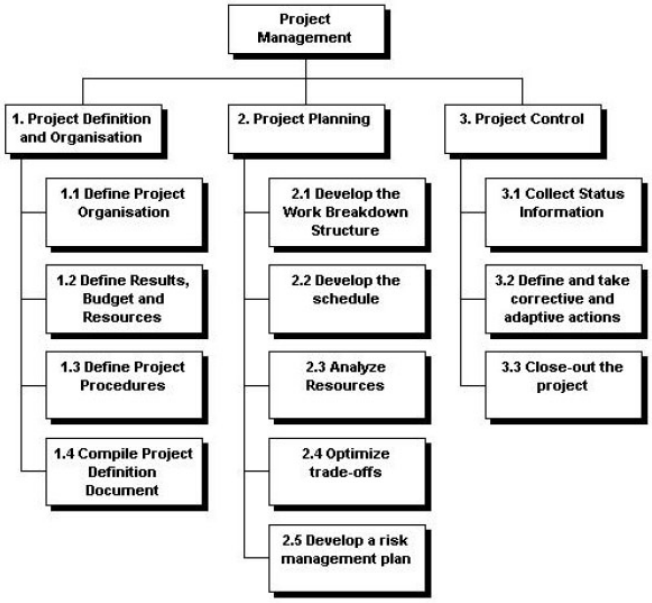

Figure 1: An overview of everything that is part of Project Management.

3.2 Project definition

The first thing to do, when managing a project, is to define what the project is. This consists of three sub-steps.

- We need to define the **project organization**. It should be clear which person has what responsibilities.
- We define **results, budgets & resources**. This is the step in which the POS is defined. It should be known when the project should be finished, and what resources are allocated to it.
- We also define **project procedures**. How do team members communicate? How often do we have team meetings? What do we do if people are late? How do we solve issues in the team? Questions like that should be answered.

Once these three subjects are defined properly, the **Project Definition Document** (PDD) can be compiled. The project is then defined.

3.3 Project planning

The second phase of project management is project planning. In this step, it is decided what processes take place at what times. There are five important parts in this planning.

- First we need to know what **work** needs to be done. For this, Work Flow Diagrams and Work Breakdown Structures are used.
- Once we know what needs to be done, we can set up a **schedule**. What team member is good at what task? And what tasks can be done simultaneously? (Think of Concurrent Engineering.)
- We then analyze the **resources**. (With resources we also mean non-material resources, like manpower.) Are there resources that are overutilized or underutilized? In this step, **Gantt charts** can be used. Although we won't go into detail on that.
- We **optimize trade-offs** between the project parameters (results, budget, resources). Will the POS be satisfied? Can we satisfy it in a more efficient way? What other parts of the planning can be optimized?
- We develop a **Risk Management Plan** (RMP). What risks are there? How can we minimize the probability of occurrence? And how do we keep the damage as low as possible, if something bad does occur?

When these five steps have been performed, the project planning is complete.

3.4 Project control

During the project, the Project Manager will not sit still. It is his task to inform key participants of the progress made. He/She should also realign the project, when things tend to go wrong. The tasks of the Project Manager can be summarized in three groups.

- The Project Manager has to **collect status information**. He should determine what information is monitored, how this is done, and how often it is done. Important subjects to monitor are the schedule status, the open issues and the risks that are present.
- The Project Manager should also take **corrective and adaptive actions**, when necessary. It is his task to determine which actions need to be taken, and how these are communicated to the team and the stakeholders.
- Finally, the Project Manager should **close out the project**, once it has been completed. What can be learned from the past project? What could have been improved? And has all the paperwork been finished? Those are questions that need to be answered.

When a Project Manager performs all these tasks succesfully, the project has a high likelihood of succeeding. However, if the tasks are not performed satisfactory, the project will most likely fail.