—— 3 ——
Basic concepts

FMV Grund-DTD is based on a few basic principles:

- *It must be highly content-oriented*
- *It must organise the information into small modules*
- *There must be a close connection between information modules and objects in a materiel breakdown structure*
- *No information should be duplicated, information should be stored once, and then be re-used wherever it's needed (within reasonable limits)*
- *HyTime syntax is used for all links*

Contents

Content orientation

In a product model environment, we have not much use for information encoded as headers and footers, heading level 1-5, numbered lists and unnumbered lists, etc. The information is used together with other product data, and could be retrieved and used for any purpose. A chunk of information and its title may be presented as a section at level 4, or at level 2, or the title may be presented as a button and the information in a pop-up window. The possible usage variations are enormous, and we must not have codes that pre-empties the possibilities.

Instead of coding the "implied layout", the codes must give information about the content, i.e. instead of answer the question "how is this piece of information presented?" the code should tell "what **is** this piece of information?".

When analysing information, there are several possibilities of describing its structure. This can be described as in the following figure.

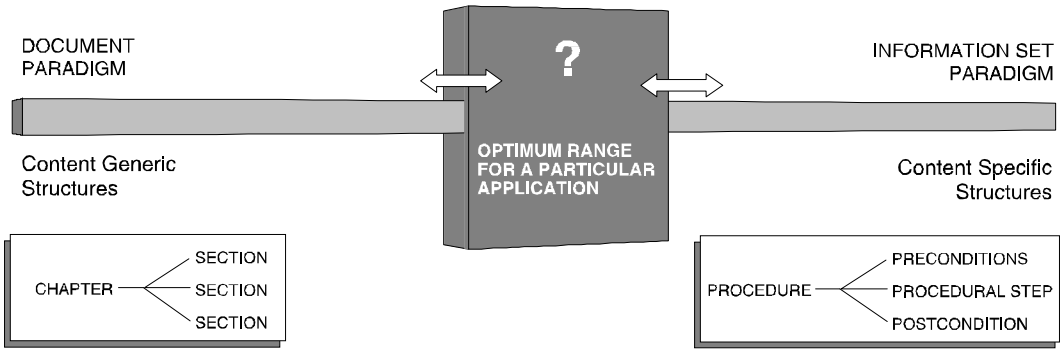

Figure 7. The continuum between document oriented coding and content oriented coding

The "document paradigm" represents the layout oriented coding, and the "information set paradigm" represents content oriented coding. There are no fixed boundaries between them. For a particular application, one must find the optimum mix of these two. For FMV Grund-DTD, the aim has been to make it as content oriented as possible (far to the right), but still, the descriptive parts are more over to the left.

Take a fault finding procedure as an example. It is normally presented as a table in today's documentation. Using the document paradigm, it would be coded as a table, with a number of rows and columns, but we would not be able to identify the information inside the table as fault finding information, and it would always have to be presented as a table. But if we use the information set paradigm instead, it would be "symptom" or "fault code", followed by "probable cause" and "appropriate corrective action". This may be processed, since we know what the information inside the code is, and it may be presented in many different ways.

FMV Grund-DTD strives to apply strict content coding of the pure, naked information. The "pure, naked information" is exactly the information that is needed to e.g. clean the pump (regardless of where, who and what for), no more and no less, and without any burdens from "implied layout" or alike.

Information modules

The information that is needed in order to understand and perform a task with or on an product (i.e. information for operation and maintenance) must be part of the product model. This type of information is traditionally found in technical publications. But, can we then argue that traditional technical publications should be placed in the product model?

No. The product model needs smaller information chunks ("modules"). Each module must be clearly identified as to what it is, and they must be discrete and clearly defined.

With FMV Grund-DTD, technical information is stored in an **information module**.

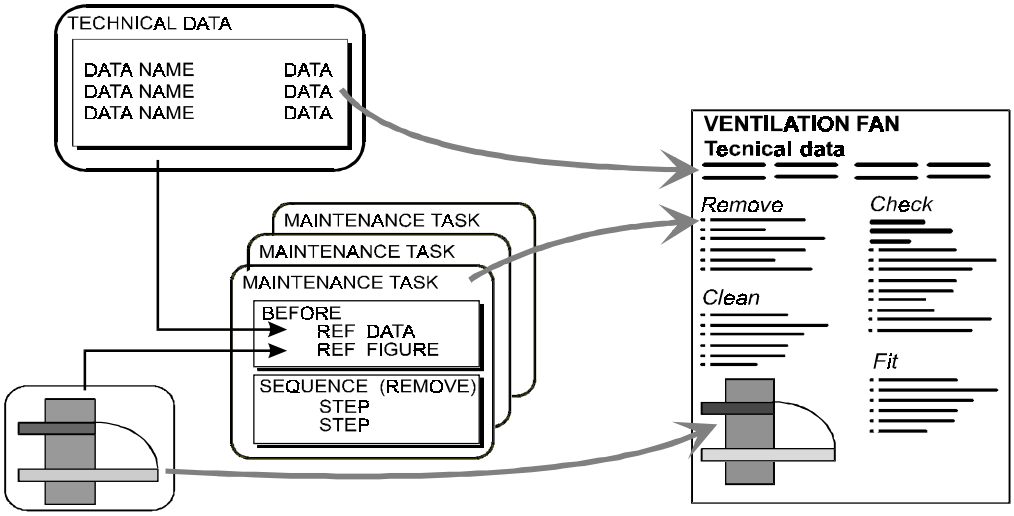

Figure 8. Information modules used to build a repair section document (procedural information

Each type of technical information, e.g. functional description, operation, fault finding, preventive maintenance, maintenance task (these are further discussed in "FMV Grund-DTD overview") is stored in a separate information module. The reason for this is basically a consequence from the product model vision, where it should be possible to retrieve separately any type of information regarding an object. It would be irritating if it wouldn't be possible to find the fault finding information at once, if for instance all technical information was placed in one module, and then you had to search through that module.

An information module contains information regarding **one** function or physical object, e.g. a spare part, a pump, a disassembly, an operational task. If the object is part of a system (which is defined as an object, too), all information concerning the system is placed at the higher level information module of the same type. This means that "removal" is part of the system's information, since the object is removed from that particular system.

Similarly, if the object is broken down into components (also objects), the information concerning one of the components should be connected to that component.

An information module is **not** the same as a document, whether digital or printed. A document is a collection of one or more information modules, with a specific layout or presentation, and possibly with the information pieces slightly re-organised. A publication is a collection of one or more documents, in this sense. Typically, an information module is a smaller piece of information than a traditional document.

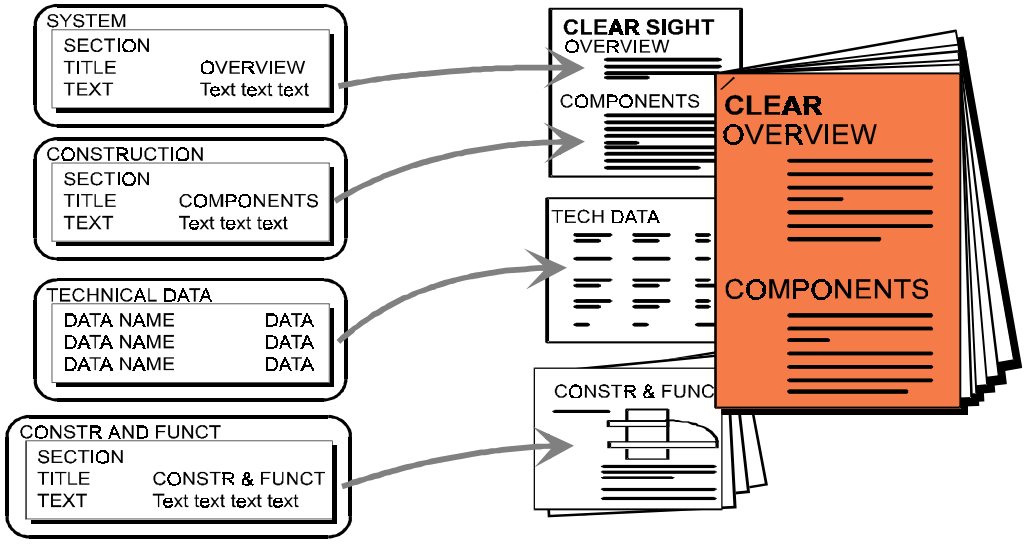

Figure 9. Information modules used to build a descriptive document

It is therefore essential that the technical writers **think** information modules, and free themselves from the traditional "document thinking".

Connection to objects

An "object" is defined here as a logistic candidate, i.e. a part of the materiel system that will be treated as a separate unit in operation and logistics, defined in the logistic support analysis (LSA), or equivalent. It may be a physical object (something you can touch: pump, engine, etc.), a functional object (e.g. lubrication system, fire control system, etc.), or a function (starting, stopping, dialling out, firing, etc.). In this document, "object" is used in the sense "a physical or functional object, or a function".

The hierarchical structure of objects in a materiel system is not part of the FMV Grund-DTD structure, such information is given by the product model (or e.g. the materiel system breakdown structures in an LSAR – logistics support analysis record – database). The DTD is equally suitable for radios, armoured vehicles, ships, air planes, etc. The level of granularity used in the materiel system breakdown may be different between materiel systems.

The DTD does not give any guidance as to how the object fits into the materiel system, it only covers the structure of the technical information itself, i.e. inside an information module.

The information inside an information module is concentrated to deal exactly with what the module is intended for, in relation to the object. The information must be organised in an object oriented way to fit in with the product model, which is achieved through the use of information modules. An information module is always connected to the object it describes.

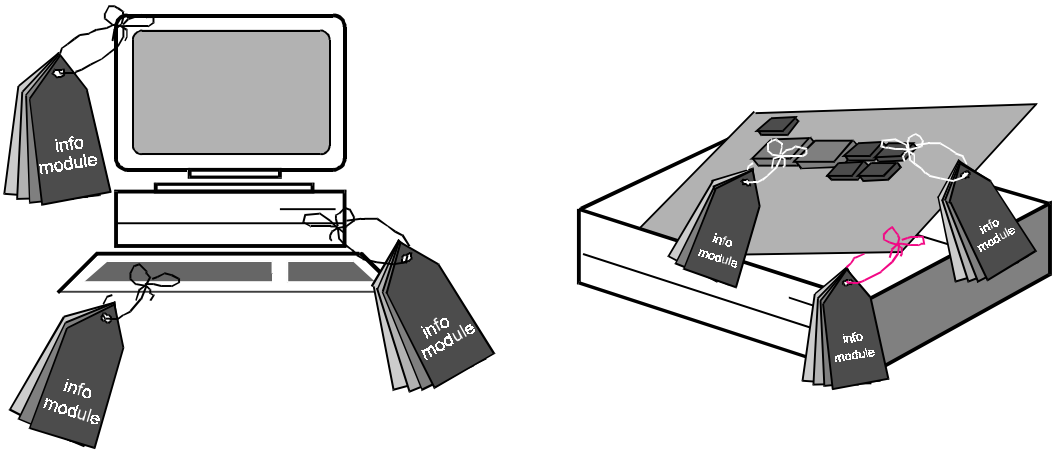

Figure 10. Information modules are "connected" to the object

Since the object have other types of information connected to it, as well as technical information in information modules, any other information in the product model can be accessed from within an information module, through the object connection.

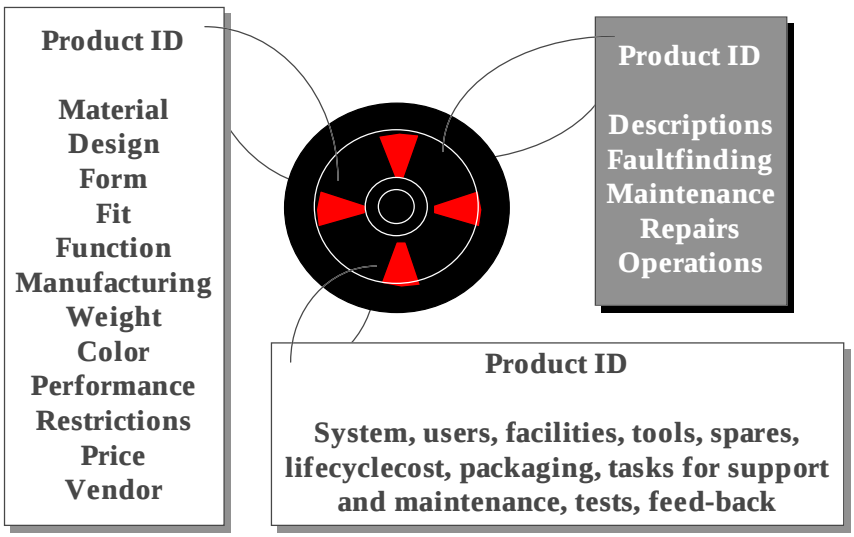

Figure 11. From the object, any of its information can be retrieved

Because of the organisation of different information types in separate information modules, several modules will be connected to each object. And vice versa, if two objects share the same information, e.g. operating instructions, one module could be connected to several objects.

The actual connection between information modules an objects are implemented in HyTime (see section 4).

Avoiding duplication of information

Another requirement for the DTD is the reduction of duplicated information. In a product model world, information that is used in more than one place should be stored once and linked to where it is used. This is to a great extent achieved with the use of information modules as described above.

But FMV Grund-DTD also gives the possibility to work with smaller information chunks than information modules, through the use of **information fragments**.

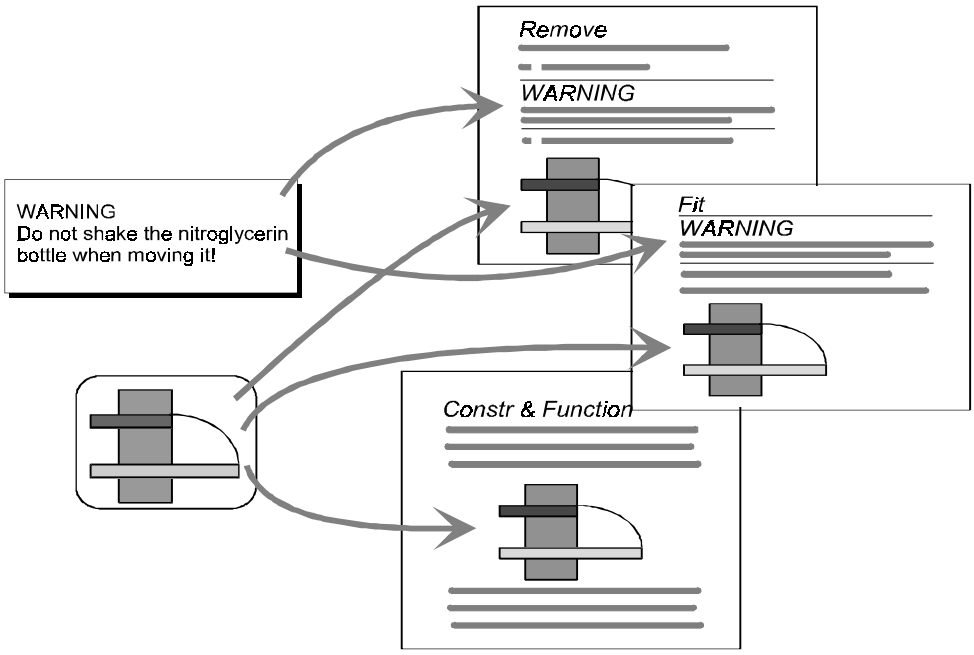

Figure 12. Information fragments used in information modules

An information fragment is a re-usable chunk if information, smaller than an information module. It is not self-contained as the information module, but is truly a fragment. It is used in several information modules, but is only stored once (as a separate entity or inside one of the entities that use it).

Examples of information fragments would be graphics (which even today in other SGML applications are handled as information fragments), standard warnings, general introductions, a value from a database, id-number of a tool, etc. The use of information fragments to create versions and variants of information modules may soon become cost-effective.

Through HyTime links, any identified information chunk may be included in another module. Whether the fragment is stored separately, or as part of one of the information modules is of no concern to the HyTime link.

Methods for linking

Storing the technical information in small information modules result in high demands for thorough linking mechanisms. The number of links between information modules will be much greater than between traditional documents, simply because the level of granularity is much finer.

The way in which software tools manage links is also more crucial for an application of FMV Grund-DTD. Therefore it became a basic principle to handle all links in a standardised and consistent way. Thus, all links are encoded using HyTime syntax.