

# Generic Product Modeling for Mass Customization

Freek J. Erens<sup>1</sup> & Hans C. Wortmann

## **Abstract**

This paper discusses the impact of increased product variety in various business processes. It is argued that increased product variety threatens continuous improvement of quality and productivity. A solution is suggested by focusing on *generic product modeling*. The idea is that product development develops product families with well-defined interfaces between subsystems and components of these families, in such a way that these subsystems and components are also developed as families. In a similar way, it is suggested that the supplying factories are organized in a way that mirrors the product family structure. In this way, product variety can be combined with learning and continuous improvement. This idea leads to intelligent product documentation in line with the (generic) product structure. The paper argues that such intelligent product documentation is helpful in all business processes in order to cope with variety.

## **1. Introduction: the role of product variety**

The variety in products offered to the market is dramatically increasing in almost each branch of industry. There are several reasons for this increasing variety. First of all, many companies face a sharp decrease in the commercial life cycles of new product types introduced to the market. This trend is visible both in consumer markets and in capital goods markets. Of course, decrease in life cycles results in an increase of new product launches. However, the commercial life cycle of product types should not be confused with the economic life time of products installed in the field. Although the life cycle of product types is decreasing, the economic life of the installed is not -- as a rule. Therefore, the variety of products which are actually alive in the market, is *increasing in proportion to the new product launches*. Secondly, increased competition in mature markets leads to an intensive search for *niche markets*, with dedicated product types. This increases the variety of products once more substantially. Thirdly, increased variety of products is a way to obtain customer satisfaction. The general idea is, that customers prefer to choose for themselves, and that a large set of options to choose from improves customer satisfaction. These options can take the form of either "add-on" gadgets sold with the main product or as compulsory choices in determining the nature of the main product to be delivered. This customization trend leads to *mass customization*, and is superimposed on the two other trends mentioned (see Figure 1).

---

<sup>1</sup> E-mail [freek@erens.net](mailto:freek@erens.net)

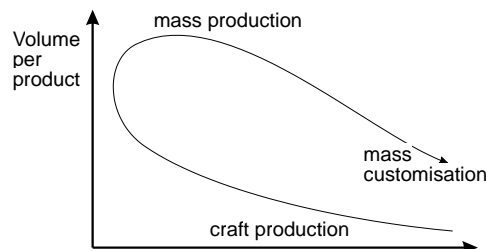


Figure 1. From craft production to mass customization

Adapted from Womack et al. [1990]

However, factories have been designed for a given set of product types. Of course, jobbing factories have always been able to produce a large product variety, but these factories are known for their difficulties in combining short lead-times and low costs with high quality. Nowadays, the design of production systems is always based on careful analysis of product types and routings (process steps) under the assumption of stability. *Continuous improvement by learning is only possible, if product types and routings are not changing continuously.*

The difficulty of handling variety is not only true for factories. Rather, all well-established business processes require some form of stability. Unlimited variety and unbounded dynamics threatens quality control, efficiency, and throughput time of any process.

Thus, this paper discusses a dilemma: product variety is necessary for survival, but unlimited product variety inhibits well-established business processes. The way out of this dilemma, to be explored in this paper, is developing product families. A product family caters for the individual wishes of customers by introducing variety within a predefined architecture. A large commercial variety can be created by a limited number of reusable modules that can be arbitrarily combined to create new variants of the product family. However, the product family concept does not only apply to the end-product, but also to the modules from which the end-product is composed. Each module can be a product family again by introducing variety on the component level within the stable architecture of the module.

## 2. Consequences of variety for business processes

Nearly all business processes are heavily influenced by product variety. Therefore, this section presents a short outline of selected business processes. The business processes to be discussed are: product development, manufacturing, marketing, order acquisition, delivery and after-sales service.

### 2.1. Product Development

The classical problem in product development is, that the link between the company's strategy and the individual product development projects is missing. This link is called *product planning* here, but it is sometimes called by other names such as program management or multi-product management. Symptoms of this problem are the inability to reuse knowledge and components and the continuous need for radical innovation. Furthermore, a symptom is that individual product development projects are shooting at a moving target, driven by the product announcements of competitors. Finally, a symptom is that functional specifications require much effort when being translated into technical solutions, especially because these technical solutions are continuously changing.

It is the task of product planning to decide how to exploit the firm’s core capability, which basic technology, product platforms and families to develop, which technology to acquire, which strategic alliances to start and which resources to allocate for a product family to introduce in the market.

When product planning has positioned a product family in the company’s long-term strategy, it is the responsibility of product development to develop and introduce it. Besides project control, the main interest of product development lies in:

- defining the product family’s core functionality; and,
- defining additional options to cater for individual customer wishes.

The core functionality of a product family should distinguish it from competitor’s products and should determine in which market segment it can be positioned. The core functionality is also closely related to the applied technology and governs in that respect possibilities for future extensions. This also means the specification of life-cycle issues as backward and forward compatibility. Options, on the other hand, give possibilities to provide a range of product variations which cater for anticipated individual differences between customers.

Henderson and Clark [1990] define innovations that change the way in which the components of a product are linked together, while leaving the core design concepts untouched, as architectural innovation. It greatly diminishes the usefulness of a firm’s architectural knowledge but preserves the usefulness of its knowledge about the product’s components. Table 1 classifies innovations along two dimensions. The horizontal dimension expresses the impact of an innovation on core concepts, while the vertical captures its impact on the linkages between modules.

	Core concept reinforced	Core concept overturned
Linkages unchanged	Incremental Innovation	Modular Innovation
Linkages changed	Architectural Innovation	Radical Innovation

Table 1. A framework for defining innovation

As can be seen from Table 1, radical and incremental innovation are extreme points along both dimensions. Furthermore, two other types of innovation are shown. Innovation that only changes the core concepts of certain modules is called modular innovation, for example the replacement of an engine type in an existing car design. Innovation that changes the linkages between modules is called architectural innovation, for example the design of a small copier out of existing modules of a large copier.

The above classification shows that the concept of product families requires a certain maturity of both market and technology. This maturity is reflected in a stable product architecture that addresses a well-known market. Variety is created by changing components in a pre-defined architecture. This requires the acceptance of a dominant design which incorporates a range of basic choices about the design that are not revisited in every subsequent design, for example when a specific variant is derived from the product family concept. Usually, these dominant designs do not exist yet for innovative products, as the relationships between function and technology are not fully understood and still subject to improvement.

If the basic function, technology and architecture of a product are familiar enough to create variations that cater for individual customer requirements, a product family can be developed. If the architectural knowledge is stable, it tends to become embedded in the practices and procedures of the (learning) organisation. In these cases, only incremental and modular innovations seem to be possible. A radical or architectural innovation requires the development of a new product platform causing many changes for a company. Real innovative products are usually not designed as a product family. New technologies, or new combinations of technologies, require considerable experience before they can be applied such that they meet a diverse range of customer requirements.

Intelligent product documentation is extremely important in the development phase as product families are intangible. The communication about product families is even more difficult than communication about product variants or any other single product that is developed: a product family does not exist physically. It is not possible for a group of people, representing different domains, to surround a product family and discuss how and where their views are present in the artifact. It is unfeasible to indicate which components are responsible for a certain function and how these components are assembled as there will be a variety of functions, a variety of components and a variety of manufacturing processes.

In other words, a product family only exists as a mental picture or a formal model. It has been created for a variety of contexts (again an abstraction) and can only be validated by studying the fit of its variants in their respective contexts. One of the goals of this paper is to enable such a formal model of a product family that it approximates the mental model, not the actual world, to the maximum possible extent. This formal model is named generic product modeling.

## **2.2. Manufacturing**

Problems in manufacturing are all centered around *change*. It is important to note, that factories should not be confused with machines: machines are fully automated systems which are not designed for change. Factories are man-machine systems which should continuously improve by learning (Sol [1992]), and at the same time adapt and reorganize themselves. Now it is well known, that *learning* always happens within a certain context of stability. Therefore, unlimited and unbounded variety of products prohibits learning.

In this context it is interesting to mention the “law of requisite variety” [Ashby, 1969]. He states that only variety can control variety. In other words: if one wants to master the variety in an environment, the complexity of the control capacity must match the complexity of the system being controlled. If there is insufficient control capacity, one can:

- reduce the need for regulation; and,
- improve the control capacity.

The classical way to deal with problems of variety in component manufacturing is through production flow analysis and group technology. In *group technology* [Burbidge, 1989], a seemingly endless variety of components is grouped based on similarity in routing. This group of components is matched with a part of the factory responsible for the complete manufacture of the group -- *a focused factory* [Skinner, 1974]. However, such a group of similar products can be described much more powerful as a product family.

In a similar way, a family of similar sub-assemblies will be matched with an assembly line or work area, focused on precisely this family. Therefore, it is possible to match the generic product family

structure with the delivery structure of focused factories. If focused factories match product family structures, it becomes possible to link product innovation (see Table 1) to production innovation.

- In case of *incremental* product innovation, where linkages in products remain unchanged and the core concept of product families is reinforced, the factory can remain a learning organization and work for continuous improvement.
- In case of *modular* product innovation, where the core-concept of a particular module is overrun, but linkages remain unchanged, it is very clear which part of the factory has to be restructured.
- In case of *architectural* product innovation, where modules are not immediately changed, but new interfaces are defined, parts of the factory remain the same, but the lay-out or routing may dramatically change.
- However, *radical* product-innovation leads to restructuring a whole factory.

We may conclude, therefore, that proper matching of product family structures with factory organization structures makes innovation in production processes manageable and makes design-for-production feasible, rather than running from one accident to the next one.

An important concept in relation to mass customization is the customer-order decoupling point. The customer-order decoupling point separates the customer-order driven part of the activities from the activities that are based on forecast and planning. In general, the decoupling point will coincide with a main stock point. Hoekstra and Romme [1992] distinguish five different positions of the customer-order decoupling point to describe the most distinguishable product-market situations in the control concept. These are graphically depicted in Figure 2.

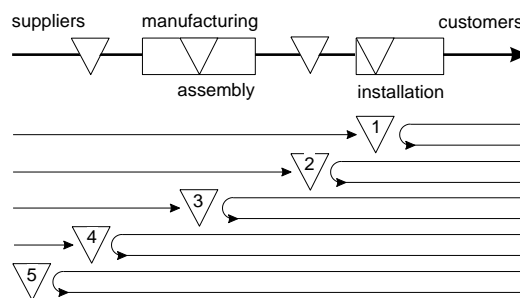


Figure 2. Customer-order Decoupling Point

Usually, products that are made or shipped to stock have a limited variety, as it is not possible to keep all variants of a large product family on stock. On the other hand, products that are assembled, made or purchased to order give the opportunity to create, with a limited number of modules, an almost infinite variety. The variants of a product family are often assembled to customer-order.

### 2.3. Marketing

This business process could consist of Market Analysis, Product Planning, Promotion, Sales Analysis, and Forecasting.

- *Market Analysis* is complimentary to the activities "idea generation" and "idea selection" of the business process Product Development. Market Analysis tries to investigate comprehensively the market needs.
- *Product Planning* translates these needs into planned product development projects, in such a coherent way that the portfolio of future products is "in balance" -- in terms of both coverage and life cycles. In addition, Product Planning estimates required investments in production, marketing, physical distribution, sales support, as well as potential revenues, with the purpose of guaranteeing profitability.
- *Promotion* accompanies product launches, but should remain a sales support activity throughout the commercial life cycle of the product, in order to manage these life cycles according to the product plan.
- Product Planning and promotion get feedback from *Sales Analysis*, in which activity market shares, competitor's behavior and short term market trends are analyzed.
- *Forecasting*, finally, is also based on Sales Analysis, and yields input for several of the marketing activities but especially for manufacturing and service.

All these activities require meaningful ways of *aggregation*, and in mass customization aggregation is a difficult problem. The reason is, that each product is different. Therefore, marketing forces sales to specify a whole list of attributes of each customer order for later analysis. This is the reversed world. The correct approach is to develop product families from which different product variants can be derived. The differences between these variants are indicated with commercial parameters that are used for sales purposes but can also be used for aggregation purposes.

A distinction between a commercial view and a manufacturing view is well suited for market analysis and aggregation, because it provides the possibility to release new versions of products with regular technological upgrading, without changing the functional description of the product in the market.

Finally, *promotion* of a large number of different variants is only possible if the common and discriminating characteristics are known and promoted. Again, product families and consequently generic product modeling is a must here.

A research issue is, how *accounting* systems should work with product families. In many companies, accounting systems are entirely based on code numbers for products. Other companies have experience in selling customized products, but calculate customer-order specific materials and activities as a basis for costing. In principle, these techniques can be used in the case of product families. However, in many cases there are virtually no customer-order specific activities (such as when customization is a matter of software installation) or the variable costs are not related to the value of the product (for example, again, in the case of software).

## **2.4. Order acquisition**

This business process consists of finding the customer, identification of the customer's needs, preparing an offer, and communicating the contract to manufacturing and delivery. Classical problems around product variety occur first of all, when preparing an offer for a customer. Ever changing products (and product documentation) is a main cause of difficulties, especially if products are not sold from the shelf, but built and shipped in a customer-order driven way.

The customer's needs may include system's integration problems, financial conditions such as leasing, and system's upgrading -- also in functional terms.

Sales support systems are perhaps gaining most benefits from generic product modeling. This is not only, because electronic documentation and multi-media applications cannot be made for millions of variants separately, but only for families. It is also because *configuring* variants is one of the oldest applications of generic product models. A variant which is described by a number of parameters should be fed into the production control system, in order to get a delivery date, a cost price, and other information which is dependent on the state of the production system.

An important advantage of such a configurator is, that a change in the customer's preferences is easily entered into the computer system, and easily translated into money or delivery time.

We conclude that intelligent product documentation is most crucial in this business process. Furthermore, business applications based on this documentation should support this business process. Generic product modeling provides the basis for intelligent product documentation.

## **2.5. Delivery and after sales services**

Delivery (or physical distribution) represents all activities to ship a tested system and get it installed, tested and integrated at the customer's site with full customer satisfaction. In most companies, customer-driven physical distribution is difficult. The reason is, that planning and control systems in physical distribution have traditionally been based on batches of identical code-numbers to be shipped in a unit of shipment (pallet or truckload), and not on different variants of the same family to be shipped in a unit of shipment.

Testing and integration of a system at the customer's site requires excellent product documentation, probably in electronic form. Generic product modeling constitutes a first step towards intelligent, electronic documentation.

After sales service does not only include service and maintenance, but also creating opportunities for upgrading, where this process triggers order acquisition activities. Again, excellent product documentation of the installed base is vital here. By conventional means, it is almost impossible to have system's documentation for each installed system, especially if the system is a variant which has been upgraded. Intelligent product documentation, such as generic product modeling, makes it possible to aggregate in an meaningful way, for example by clustering individual product variants to their original families, thereby providing better opportunities for preventive and corrective maintenance actions and, as mentioned before, for upgrading. In a similar way, generic product modeling provides a basis for tele-diagnosis and tele-upgrading (for example by down-loading new software).

## **3. Generic Product Modeling**

A *product family* is usually designed for a specific market but caters for the individual wishes of customers by introducing variety within a defined product architecture and within a defined manufacturing process. The variety of a product family depends on the variety of its modules. The idea is that combining module variants results in larger number of variants on the next higher level of the product structure. This is depicted in Figure 3. However, at lower levels of the product structure, the variety of modules does not originate from a smaller variety of components as modules do not share a high proportion of components.

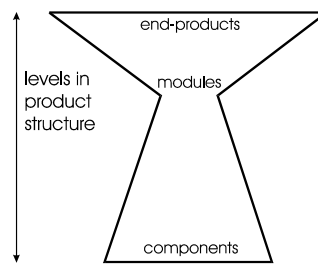


Figure 3. Diabolo

Product families are intangible products, they do not exist physically. This stresses the importance of a modeling language in which the variety of the product family can be described without redundancy and without losing insight in the generic structure of the product family.

The generic product modeling language is in essence a parametrised data-structure. A (unique) *variant* of this family is defined if all parameters have a value. Generic product modeling is a way to describe product families with well-defined interfaces between components of these families, in such a way that these components are also described as product families. The idea has been elaborated for a "product description language" that is used in the operational manufacturing process, namely bills-of-material. However, the idea can also be applied to product description languages that are used in non-operational processes, e.g. the development process. Both CAD systems and Engineering Data Management Systems can benefit from a modeling language for product families. This will be demonstrated in the next section.

Throughout this paper, an example from Philips Medical Systems will be used. Figure 4 gives an example of a cardio-vascular system, which is used for examining a patient's heart and vessels. This system can be manufactured in a few million variants by making combinations of stands, X-ray tubes, image-intensifiers, TV-chains, software modules and tables.

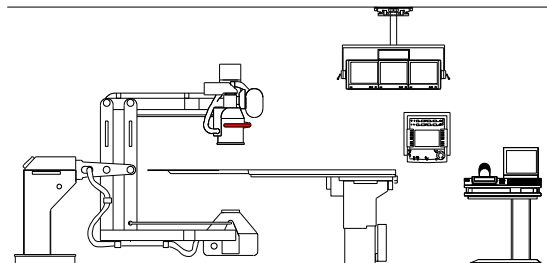


Figure 4. Cardio-vascular system

The generic bill-of-material (GBOM) concept integrates the commercial and assembly view on a product family into a single intelligent product model. It has as objectives to:

- model a product family from both a commercial and an assembly viewpoint;
- model a product family without data-redundancy and without knowledge redundancy;
- generate complete bills-of-material for the family's variants;
- enable the use of applications for production control;
- improve product transparency, e.g. for product management and development.

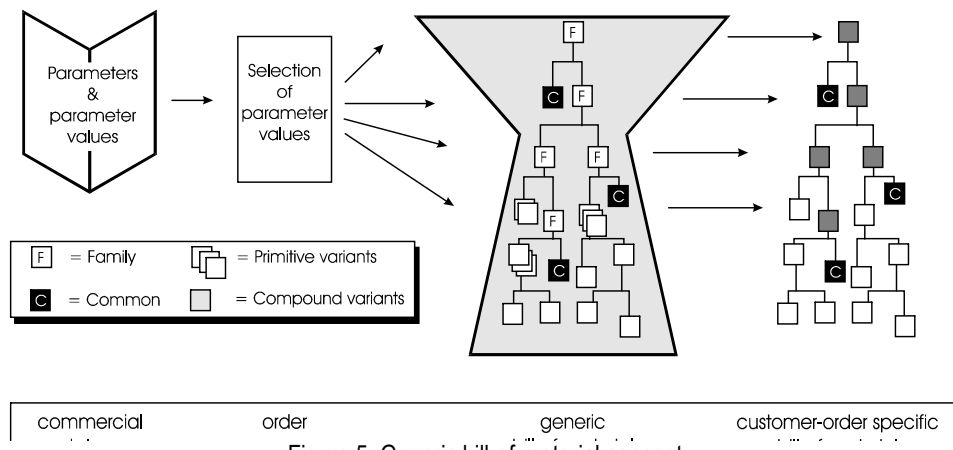


Figure 5. Generic bill-of-material concept

These objectives are summarized in figure 5. The realization of these objectives is as follows:

1. Functional characteristics describe best the commercial viewpoint on a product family and its variants;
2. The product structure which is similar, however not identical, for all product variants can be used for modeling the assembly viewpoint;
3. The customer and assembly view on a product are different, but can be linked in a shared product model;
4. Specific bills-of-material (product variants) for specific customer orders can be generated from the product family description.

These characteristics of the generic bill-of-material concept are elaborated in the following sections.

### 3.1. Functional characteristics and the commercial viewpoint

The commercial viewpoint is based on the functional characteristics of a product family and its variants. Functional product characteristics:

- abstract from the technical components;
- are normally understood by both sales and manufacturing; and,
- are more stable than the ever-changing component and end-product specifications.

Functional product characteristics can best be defined using parameters and parameter values. Together, all parameters and parameter values describe a complete product family from a commercial viewpoint. Constraints on these parameter values prohibit technically impossible or commercially unwanted product variants. Research on this issue was originally done by Digital Equipment. The XCON configuration system [Barker, 1989] was used to validate the technical correctness (configurability) of customer orders and was the first expert system in daily production use in industry. Other rule-based configuration systems are discussed by Bourke [1991, 1992, 1994].

Figure 6 shows a simplified choice-sheet (the commercial view) of the aforementioned cardio-vascular system. It shows that an order for a medical system is created by marking the relevant parameter values. An electronic implementation of this commercial catalogue will check the possible violation of constraints automatically.

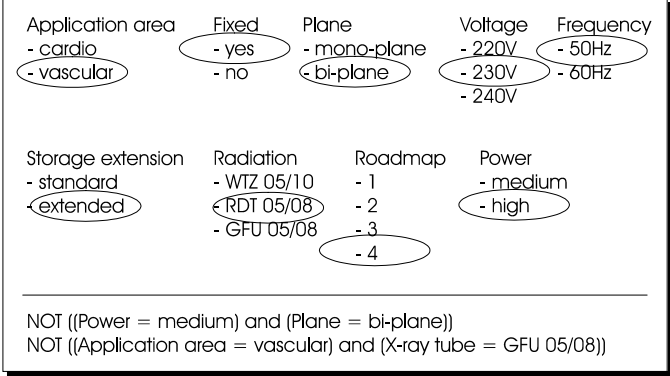


Figure 6. Customer order specific product variant

The advantage of parameters, parameter values and constraints in a sales process is that the variety can be expressed in a more formal way which is eventually easier to maintain, especially for complex product families.

**3.2. Similar product structures and the assembly viewpoint**

The different variants of a product family have a similar product structure and are often assembled on the same production facilities with the same people and production machinery. This means that the assembly and manufacturing operations have a repetitive nature (which adds to efficiency and productivity of the manufacturing process), but towards the market the product variety can be maintained, thereby meeting the demanded product proliferation.

A cardio-vascular system, for example, is a product family with millions of possible variants. However, all these variants bear much resemblance to each other. They all have a stand, a table for the patient to rest on, an X-ray tube, a generator, an image-intensifier and control software, although the exact specifications might be different for the individual variants. A classical bill-of-material describes these facts over and over again. Therefore, classical BOMs contain much redundancy in data and knowledge. A GBOM abstracts from the detailed differences between the different variants. Due to its knowledge oriented data structure, it avoids redundancy and it enables intelligent applications. A simplified example is shown in Figure 7.

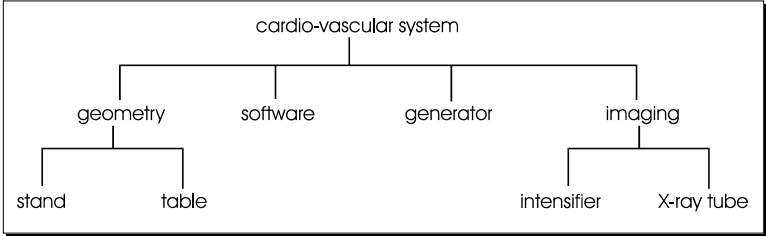


Figure 7. GBOM for a medical system

Both the final product such as a cardio-vascular system and its sub-assemblies and components can be regarded as product families, since they have a number of variants. A stand, for example, might be delivered in a cardio and in a vascular design, while the X-ray tube might occur in different variants for different types of medical examination. This variety at lower levels in the product structure is the reason for the proliferation of variety at higher levels in the product structure.

**3.3. Different customer and assembly views**

When a customer wants a stand suitable for cardio examinations, (s)he is normally not bothered with the detailed technical implementation of this requirement. In other words, the customer orders a "product feature" rather than the collection of physical parts which are used to realize that feature. The customer view is usually represented with parameters, parameter values and constraints as was discussed before.

On the other hand, the manufacturing view is based on the physical architecture of the product family. As both the customer and the manufacturer have different views on a product family, they should be kept consistent; a change in one view is often accompanied by a change in the other view. In order to create a generic bill-of-material it has to be determined precisely which parameters influence which (component) families. This is visualized in Table 2 for the simplified cardio-vascular example.

Parameter	Component Families
application area	stand, table, software
fixed	stand, table
plane	stand
voltage	generator
frequency	generator, intensifier
power	generator
X-ray tube	X-ray tube, intensifier
storage extension	software
roadmap	intensifier, software

Table 2. Cardio-vascular parameters and the component families they affect

The parameter *fixed* influences 2 component families, as can be seen in Table 2. Such a parameter is best controlled at the first common parent of the component families that are influenced. This means that the parameter *fixed* will be related to the geometry sub-assembly of Figure 7. The parameter *voltage* on the other hand will be related to the generator as this is the only component family it influences.

The values of parameters as *fixed* can then be inherited through the product structure to the relevant component families. Therefore, a distinction is made between the internal and external parameters of a given product. An internal parameter is a parameter that is defined at that family. An external parameter is defined at a higher family and one of its values is inherited by the component families that make use of it.

Figure 8 shows a subset of the assembly model and the parameters that are related to the families of this Physical Model. For example, *fixed* is an internal parameter of the geometry as it is defined at this family. For the stand and the table, it is an external parameter, inherited from the geometry. The

parameter *plane* is local to the stand and is not used by other families in the structure. It is clear that such a parameter is normally easier to control from both an engineering and a logistic viewpoint. The parameter *application area*, on the other hand, is defined high up in the structure as it not only influences the stand and table, but also other parts of the system.

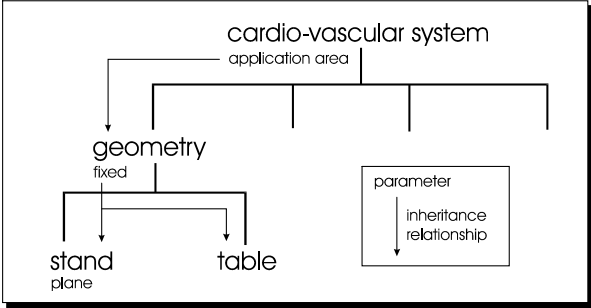


Figure 8. Inheriting parameters

**3.4. Generation of specific bills-of-material**

A customer order of which the product specification is made in terms of parameters and parameter values can be interpreted by a generic bill-of-material system to create a specific bill-of-material. A generic bill-of-material captures a product structure which is the technical twin of the commercial choice-sheet (see Figure 6), i.e. each commercial variant specification has its mirroring assembly variant specification in the generic bill-of-material system.

We have stated before that the variety at higher levels in the product structure originates from the variety at lower levels in the product structure. The product families that are not decomposed in the GBOM and are the source of product variety, are called primitive families. Their variants are called primitive variants. All other families are compound families, having compound variants. The stand, table, generator, X-ray tube and image-intensifier, for example, are all primitive families, while the geometry, imaging subsystem and the cardio-vascular system are compound families.

Parameters are related to primitive families in order to select the relevant primitive variants for a customer order (in terms of parameters and parameter values). The primitive family stand for example, has the parameters *plane* and *fixed*. The four stand variants are selected using these parameters as is shown in Table 3.

Table variant	Selection condition
stand-a	plane = mono-plane and fixed = yes
stand-b	plane = bi-plane and fixed = yes
stand-c	plane = mono-plane and fixed = no
stand-d	plane = bi-plane and fixed = no

Table 3. Selection conditions for primitive variants

The generation process of a compound variant can be summarized as follows:

1. Select a value for each internal parameter related to a product family.

2. Inherit the parameter values to the component families that have external parameters that match the parameters of the parent.
3. In case of a compound family, create a customer order specific compound variant or select a predefined variant if it happens to exist.
4. In case of a primitive family, select a primitive variant.

This process can be executed in (commercially available) generic bill-of-material systems [Bottema, 1992]. For a more thorough understanding of the GBOM concept, we refer to a paper in which the evolution of bill-of-material concepts is described [Erens, 1992]. For a paper on the use of GBOMs in operational manufacturing processes, we refer to [Erens, 1994]. Furthermore, we can recommend [Orlicky, 1975], [Schönsleben, 1985], [Hegge, 1991, 1992, 1995] and [Van Veen, 1987, 1992].

#### **4. Role of information technology**

Until the seventies of this century, applications of information technology have been dominated by accountancy purposes. This is not surprising. After all, accountants have had the role of providing management information during many centuries when information technology consisted of paper and pencil. Unfortunately, this information is usually focused on *reporting* past performance, rather than supporting future actions.

In the early eighties, *planning* of sales, production, purchasing, and delivery was the second main investment of companies in information technology. Techniques such as MRP and DRP became well known. However the actual contribution of information technology was primarily in transaction-oriented application programs supporting simple actions in business processes. More difficult actions in business processes were left to human professionals.

Most transaction-oriented application programs are based on code-numbers. A code-number is a unique identifier of a product which is offered to the market. These code-numbers provide a "complete" description of each individual product. However, code-numbers usually inhibit intelligent product documentation, because the notion of similarity and structure in products is lost.

In the late eighties, professionals have acquired workstations with dedicated professional support systems. Engineers found their ways to CAD-systems, sales people developed multi-media applications, purchasers bought office-managers, manufacturing engineers found NC-programming tools, and many others found their ways to dedicated IT-tools to get support for professional work. Many of these professional support systems embody some form of intelligent product documentation, but not in an integrated form. Consequently, these professional support systems are difficult to maintain, and it is almost impossible to link these to business support systems.

In order to clarify the notion intelligent product documentation, we present a datastructure (see Figure 9) for the generic bill-of-material (cf. Erens<sup>b</sup> [1996]). Due to lack of space, we do not present the associated functionality (see, however, Hegge [1995], or Bottema [1992]).

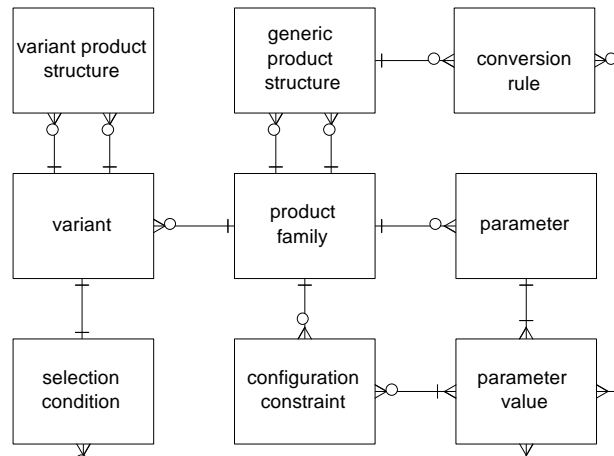


Figure 9. GBOM datamodel

The remainder of this section is used to explain the above entities in more detail:

- The core of the datamodel is the product family. Whether a product family is a primitive family or a compound family is indicated at attribute level;
- The product family is decomposed for the sake of variety into component product families. The product family relationships are recorded in a separate entity, generic product structure, as they have specific attributes, for example, the quantity-per and the effectivity dates. Two lines are connecting the product family entity and the generic product structure entity, one for implosion (component-parent) relationships, a second one for explosion (parent-component) relationships;
- Each product family has nil or more parameters. Some of these parameters will be internal and are populated with values at the product family, other parameters are external and inherit their values from parent families;
- Each parameter has one or more parameter values. If a parameter has one value, there is no explicit choice for the user;
- Conversion rules translate parameter values from parent families to component families. Each generic product structure relationship has nil or more associated conversion rules;
- Parameter values can be combined into configuration constraints, which are related to product families. Configuration constraints are expressed with Boolean logic and prohibit certain combinations of parameter values;
- Parameter values can also be combined into selection conditions, which are related to the primitive variants of primitive families. A selection condition selects the right variant for a customer order;
- Primitive families have primitive variants which are selected on customer order. Compound variants are created on customer order, and are related to other compound and primitive variants through the variant relationship entity;
- The variant relationship entity provides for a specific product structure. This product structure can be customer-order specific as far as it concerns compound variants which are created on customer-

order, but it can also be customer-order independent, for example if it concerns the product structure of a primitive variant.

Although the parameters and parameter values of the GBOM concept provide for a functional view on the product family, they only describe the family for the sake of configuring an order. They do not describe the product family with such detail that they are suitable for design purposes. Furthermore, the GBOM concept implements two views on a product family, usually a commercial view and an assembly view. In design, more views must be considered to permit balanced design decisions. In Erens<sup>b</sup> [1996] this datamodel has been extended to implement several structural viewpoints on a product family being designed. The next section gives a brief overview of other domains where the generic product modeling concept can be used.

The datamodel is called an *intelligent* product documentation system, because of the rich semantics and because of the fact that configuration constraints, conversion rules, and selection conditions can be seen as the rule base and inference engine of an expert system.

Nevertheless, we see it as an open research question, how modern IT-concepts such as multi-media, object-orientation, and artificial intelligence can be used for improved functionality of intelligent product documentation systems.

## **5. A classification of product descriptions**

In manufacturing companies, many different product descriptions can be recognized, although most product descriptions can be classified in three categories [Albano, 1992][Andreasen, 1987][Erens<sup>a</sup>, 1995][Erens<sup>a</sup>, 1996][Erens<sup>b</sup>, 1996]. These categories are represented by *product models* which act as a backbone for the combined product information. Each product model can have several viewpoints. This paper defines a domain as a product model together with its viewpoints. The product models that are shown in Figure 10 are used by different business functions and in several phases of development to organize product information in a structured way, including hierarchical and non-hierarchical dependencies:

- The *Functional Model* is a consistent description of the functionality of a product. It is strongly related to the purpose of the product. Usually, the requirements specification, created by product management, is an important input for this model;
- The *Technology Model* is a consistent description of the application of technologies (that is solution principles) to ensure the operation, but not necessarily the manufacturing, of the product. Development creates most of the information structured in this model;
- The *Physical Model* is a consistent description of the physical realization of a system. It is strongly related to the construction of the product. Manufacturing sets conditions for this realization in order to guarantee an easy assembly operation without compromising the quality level or cost level.

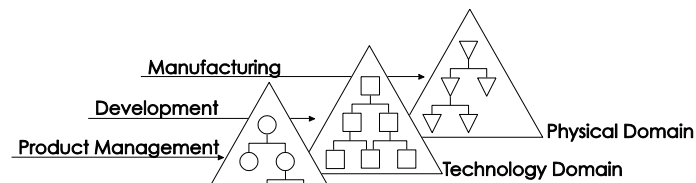


Figure 10. Product Models

Figure 10 depicts the contribution of product management, development and manufacturing to respectively the Functional Model, Technology Model and Physical Model. The depicted product models are independent of the hierarchical level of the product: systems, sub-systems, modules as well as components can be represented by these models. Each product model has several viewpoints. For example, an exploded view and a generic bill-of-material are both representations of the same Physical Model, however, they serve different purposes.

The following sections elaborate on these three domains using the example of a medical system. Medical equipment is complex, not only because of the use of advanced technologies, but also because of the combination of diverse technologies in one system. Technologies to control the position of the patient, the X-ray source and often several X-ray detectors on precise locations are to be combined with technologies that control the optimal functioning of an X-ray tube, an image intensifier, a film transport mechanism or video chains. Secondly, medical equipment is complex due to the large variety of end-products and modules. As stated before, product families are intangible, but in development even the product family's components and modules are intangible as they only exist as product descriptions before prototypes are created. This emphasizes the importance of a product family modeling language in not only the operational manufacturing process, but especially in the development process. The generic product modeling concept is appropriate for this goal.

### 5.1. The Functional Model

A consistent description of the set of functions of a system is given in the Functional Model. It comprises the hierarchical structure of functions and the interfaces between these functions. The functional requirements are primarily listed in the requirements specification in a textual form. In a second stage they are converted into a more formal description. This is to establish dependencies in an explicit form. A structured analysis checks for completeness and consistency in the Functional Model [Hatley and Pirbhai, 1987]. This analysis of functional requirements has to separate functionality from technology. In the analysis, functionality is described in a hierarchical decomposition of functions to cope with complexity (see Figure 11).

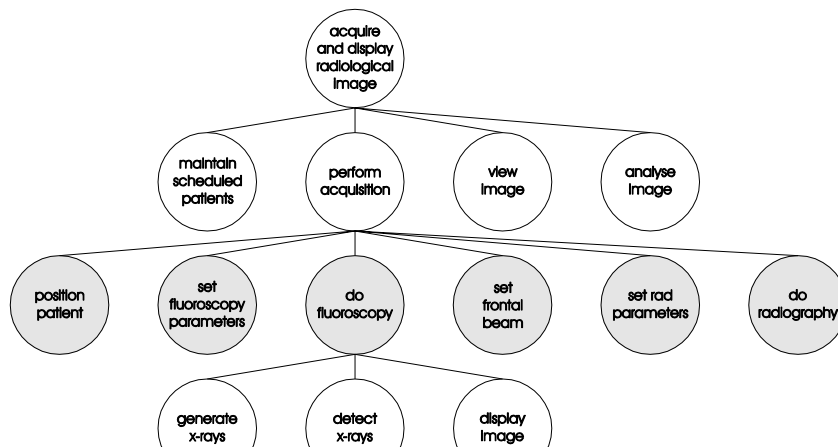


Figure 11. Functional structure

The Functional Model represents the intention of the requirements and guarantees an unambiguous realization in the design process. Each level describes the functions and their dependencies and interactions that are relevant on that level. Design decisions are taken on one level of abstraction at a time. Figure 12 gives an example of one such a level, *perform acquisition*, in the hierarchical functional structure of a medical system. Functions are depicted as bubbles. Data dependencies between functions are depicted with arrows. Functions can have variants to create even more functional variety on higher levels of the hierarchy.

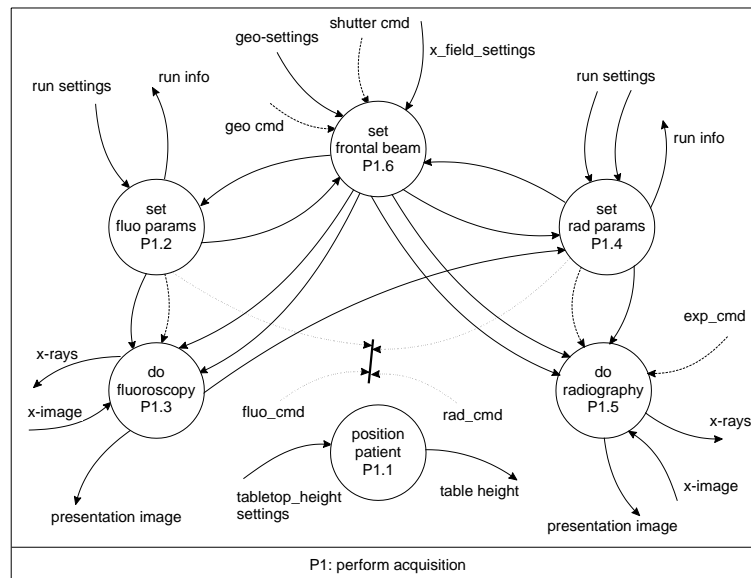


Figure 12. Functional architecture

## 5.2. The Technology Model

The Technology Model is a consistent description of all technologies that are applied in a system. It comprises the decomposition of technology modules and the interfaces between technology modules.

A block diagram describes the technological structure of the system and assists the organisation of development projects. A simplified example of the technology decomposition is given in Figure 13.

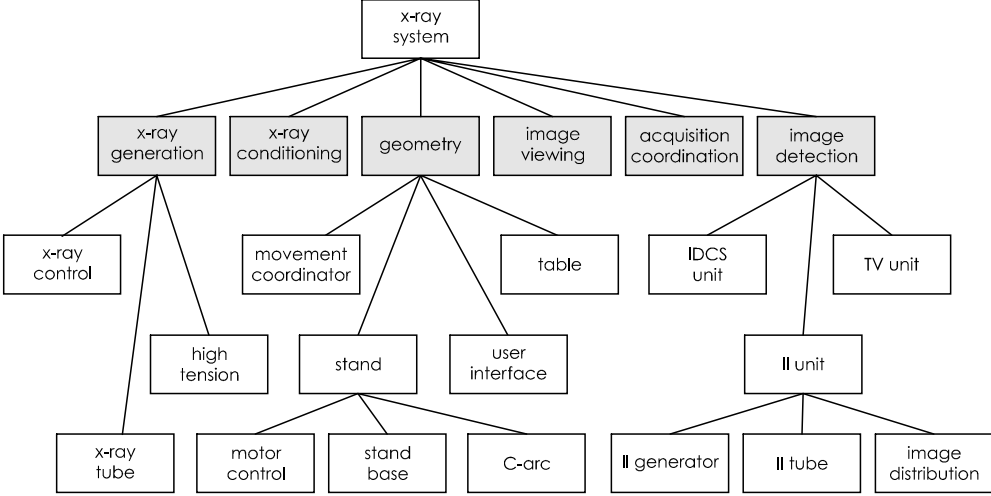


Figure 13. Technology structure

User functions are realized in a combination of modules. Similar to the development of the Functional Model, decisions with respect to the Technology Model are taken on one abstraction level at a time. The technology architecture describes the operation of a system by its modules and interfaces between those modules. In Figure 14, a technology architecture is given for the "X-ray system" on level one of the hierarchical technology structure of Figure 13.

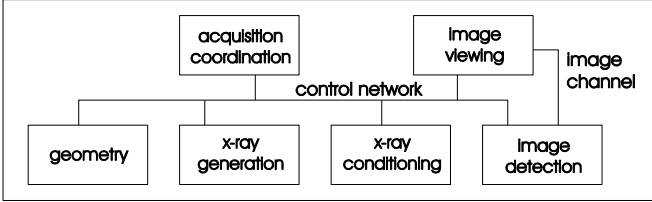


Figure 14. Technology architecture

The technology architecture consists of several modules and their interfaces. Modules are physical or logical units in which one or more functions of the Functional Model are realized, thereby taking into account constraints that have been formulated in the Requirements Specification (for example constraints on the reuse of technologies). Modules can have different variants to cater for the variety of functional requirements. Examples of modules are: a processor, a display, a video-board, a motor, etc. An interface is a specification of the interaction between modules. This can be a computer bus, an optical path or a mechanical connection. Standardized interfaces are important to create “arbitrary” combinations of module variants. If there are specific interfaces between module variants, the selection of one module variant requires the selection of another module variant. This reduces the modularity of the product family and consequently the ratio of commercial and technical variety.

### 5.3. The Physical Model

The consistent description of a system's parts and assemblies is named the Physical Model. The modules of the Technology Model are physically implemented in assemblies that can be manufactured and serviced. Design decisions in the Physical Model are taken on one abstraction level at a time. The physical architecture is described in assembly drawings. Figure 15 gives an example of the hierarchical physical structure, while Figure 16 positions the parts of the stand in the physical architecture.

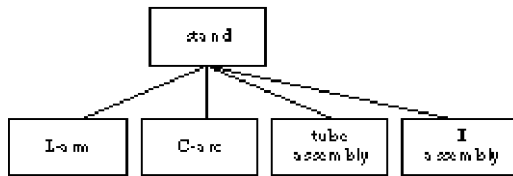


Figure 15. Physical structure

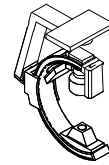


Figure 16. Physical architecture

The fact that medical equipment is manufactured with some volume is the reason for a difference between an ideal Physical Model for manufacturing and the Technology Model that is created in development. Eventually, the Physical Model is a compromise between the requirements of development, manufacturing, service and other parties that have an interest in the physical implementation of the product.

The variety that is required in the Functional Model and the Technology Model is eventually physically realized in assembly variants. Again, interfaces must be standardized to allow many combinations of assembly variants. Preferably, each function variant is realized in one module variant and each module variant is implemented in one physical assembly. If a function variant is distributed over several module variants and if these are distributed over several assembly variants, the selection of a particular function variant requires the selection of several assembly variants [Erens<sup>b</sup>, 1995].

### 5.4. Relationships between domains

The different domains must co-operate in the development process and consequently the relationships between the product models in these domains must be maintained. An important relationship for understanding the issue of product variety is the *allocation relationship*. Functions are allocated to technology modules and these in turn are allocated to physical assemblies.

Once that functions are defined at a given level of the functional hierarchy, they cannot be decomposed independently of the evolving hierarchies in the technology domain and the physical domain. Consequently, an iterative scheme of decomposition and allocation between the functional domain and technology domain must be used to incorporate new information regarding functions and solution principles. This process suggests a zigzag pattern. A zigzag pattern also governs decomposition and allocation between the technology domain and the physical domain.

Decomposition and allocation are two important steps in the design cycle that is depicted in Figure 17. This design cycle builds on the *productive reasoning model* of March [1984] and Cross [1989]. In total, there are four design steps that can be applied on all levels of the domains' product hierarchies:

- *Decomposition* is adding detail to a product model. On the highest level, the three models describe the same product. However, on that level, product descriptions are still very generic and cannot be

discriminated. Furthermore, when it is difficult to map functions onto technology modules, these functions must be detailed until they can be allocated;

- *Allocation* is the creation of relationships between elements of different product models. For example, several functions are allocated to a technology module. In the same way, several modules are physically realised in an assembly of the Physical Model;

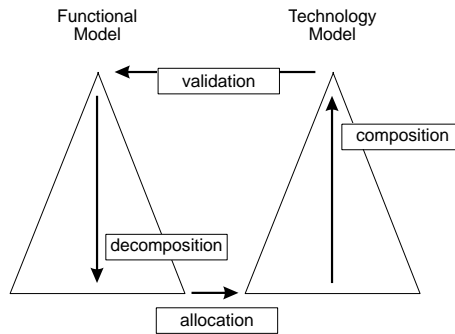


Figure 17. Four elementary design steps

- *Composition* is combining elements of a product model, for example modules of the Technology Model or assemblies of the Physical Model;
- *Validation* is a check on the realised quality level of the product model, by relating it to a previous product model. For example, a composed module is validated against the original function, while an assembly is validated against the original technology module.

The allocation and validation steps produce relationships between product architectures and consequently result in coupled domains. The decomposition and composition steps are applied to make these allocation and validation steps possible. The result is a set of harmonised product descriptions that can be unambiguously understood in the development process.

### 5.5. Consequences for Engineering Data Management Systems

In section 3, we discussed the generic product modeling concept and focused on its ability to model all variants of a product family in a non-redundant way. The examples of the functional, technology and physical domain demonstrated that the generic product modeling concept should not only be applied for modeling product families in the operational manufacturing process, but especially for modeling product families in the development process.

This gives the following requirements for an Engineering Data Management System (EDMS):

- It must be possible to distinguish different domains and viewpoints in the development process and in the operational manufacturing process;
- In each domain, the generic product modeling concept can be used to describe the product family in that domain in a non-redundant and transparent way;
- It must be possible to maintain allocation and validation relationships between product models in different domains in order to keep the entire product family description consistent;

- Traditionally, information systems for manufacturing and development focus on product hierarchies. However, the development of product families is strongly based on product architectures with standardized interfaces (non-hierarchical relationships);
- Finally, the product descriptions should be related to the design method that is applied. The four elementary design steps of the Design cycle are closely connected to both product hierarchies and product architectures.

Other requirements are given in [Erens, 1993].

## **6. Conclusions**

Product variety plays an important role in almost every business process, because it generates complexity in the execution or management of each business process. Generic product modeling is a technique which captures the knowledge of a product family in terms of parameters, rules, and constraints. As such, generic product modeling may be called *intelligent product documentation*. It seems promising to reduce the business complexity which results from product variety.

Generic product modeling has proven its value in the form of generic bills-of-material at business processes where bills-of-material are important. There is substantial evidence, that the same approach can be applied elsewhere, especially in development.

However, we have highlighted several important areas of further research. These are:

- Generic product modeling in CAD, CAE, and EDM systems
- Generic process modeling in manufacturing and other business processes
- Shop floor control systems which take full benefit of generic product modeling
- Accounting principles for generic products and processes, especially for embedded software
- Integration of multi-media promotion/sales applications with generic product modeling.

## **References**

Albano L.D., Suh N.P., 1992

*Axiomatic Approach to Structural Design, Research in Engineering Design, 4:171-183, Springer-Verlag New York Inc., 1992*

Andreasen M.M., Hein L., 1987

*Integrated Product Development, IFS Publications Ltd / Springer Verlag London, 1987*

Ashby W.R., 1969

*Self-regulation and Requisite Variety, in: Emery F.E., Systems thinking, Penguin Books, 1969*

Barker V.E., O'Connor D.E., 1989

*Expert systems for configuration at Digital: XCON and beyond, Communications of the ACM, March 1989, Volume 32, Number 2*

- Bottema A., Tang L. van der, 1992  
*A Product Configurator as Key Decision Support System, Integration in Production Management Systems, Pels H.J. and Wortmann J.C. (editors), Elsevier Science Publishers, IFIP, 1992*
- Bourke R.W., 1991  
*Configurators: Rule-Based Product Definition, American Production & Inventory Control Society, December 1991*
- Bourke R.W., 1992  
*Configurators: An Update, American Production & Inventory Control Society, August 1992*
- Bourke R.W., 1994  
*Update: Configurators II, American Production & Inventory Control Society, January 1994*
- Burbidge, 1989  
*Production flow analysis for planning Group Technology, Clarendon Press, Oxford, 1989*
- Cross N., 1989  
*Engineering design methods, John Wiley & Sons, 1989*
- Erens F.J., Hegge H.M.H., Veen van E.A., Wortmann J.C., 1992  
*Generative bills-of-material: an Overview, Integration in Production Management Systems, Pels H.J. and Wortmann J.C. (editors), Elsevier Science Publishers, IFIP, 1992*
- Erens F.J., McKay A., Bloor S., 1993  
*Shortcomings of Today's Design Frameworks, International Conference on Engineering Design ICED '93, The Hague, 1993*
- Erens F.J., Hegge H.M.H., 1994  
*Manufacturing and Sales Co-ordination for Product Variety, International Journal of Production Economics, November 1994*
- Erens<sup>a</sup> F.J., Verhulst K., 1995  
*Designing Mechatronic Product Families, in: Proceedings of the WDK workshop on Product Structuring, Tichem M., Storm T., Andreasen M.M, and MacCallum K.J. (editors), Delft University of Technology, June 22-23, 1995*
- Erens<sup>b</sup> F.J., Breuls P., 1995  
*Structuring Product Families in the Development Process, Proceedings of ASI'95, Lisbon, Portugal, 1995*
- Erens<sup>a</sup> F.J., Verhulst K., 1996  
*Architectures for Product Families, accepted for publication in Computers in Industry, 1996*
- Erens<sup>b</sup>, 1996  
*The synthesis of variety: structuring product families, Ph.D. thesis, Eindhoven University of Technology, 1996*
- Hatley D.J., Pirbhai I.A., 1987  
*Strategies for Real-Time System Specification, Dorset House Publishing, 1987*

- Hegge H.M.H., Wortmann J.C., 1991  
*Generic bill-of-material: a new product model, International Journal of Production Economics, 23 (1991) 117-128*
- Hegge H.M.H., 1992  
*A Generic Bill-of-Material Using Indirect Identifications of Products, Production Planning and Control, Vol. 3, No. 3, PP 336-342, 1992*
- Hegge H.M.H., 1995  
*Intelligent product family descriptions for business applications: production control software based upon generic bills-of-material in an assemble-to-order and make-to-order environment, Ph.D.-thesis, Eindhoven University of Technology, 1995*
- Henderson R., Clark K.B., 1990  
*Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms, Administrative Science Quarterly, 35 (1990): 9-30*
- Hoekstra S.J., Romme J.A.C., 1992  
*Integral Logistic Structures, McGraw-Hill Book Company, 1992*
- Kotler, P., 1988  
*Marketing Management: Analysis, Planning, Implementation and Control. Prentice Hall,*
- March L.J., 1984  
*The logic of design, In N. Cross (editors), Developments in Design Methodology, John Wiley & Sons, 1984*
- Orlicky, J.A., 1975  
*Material Requirements Planning, McGraw-Hill, 1975*
- Schönsleben, P., 1985  
*Flexibele Produktionsplanung und Steuerung met dem Computer, CW-Publikationen, München, 1985*
- Sol, E.J., 1992  
*CIM -- Communication and Information in Manufacturing, in: Pels, H.J., and Wortmann, J.C. (eds.): Integration in Production Management Systems, Elsevier, Amsterdam, pp. 13-23*
- Skinner, W., 1974  
*The Focused Factory, Harvard Business Review, May/June, 1974, pp. 113-121.*
- Veen van E.A., Wortmann J.C., 1987  
*Generic Bills-of-Material in Assemble-to-Order Manufacturing, International Journal of Production Economics, Vol. 25, No. 11, p.p. 1645-1658, 1987*
- Veen van E.A., 1992  
*Modeling product structures by generic bill-of-material, Elsevier Science Publishers B.V., 1992*
- Womack J.P., Jones D.T., Roos D., Sammons Carpenter D., 1991  
*The Machine that Changed the World: the Story of Lean Production, HarperPerennial, 1991*