Generative bills-of-material : an overview

Integration in Production Management Systems
H.J. Pels and J.C. Wortmann (Editors)
Elsevier Science Publishers B.V. (North-Holland)

Generative bills-of-material: an overview

F. Erens[1,2], H. Hegge[1], E.A. van Veen[3], J.C. Wortmann[1]

1. Graduate School of Industrial Engineering and Management Science
   Department of Information & Technology, Eindhoven University of Technology
2. Lighthouse Management Consultants B.V.
3. Van Doorne's Bedrijfswagen Fabriek DAF B.V., The Netherlands

Abstract

This article presents our latests developments in bill-of-material (BOM) research in the context of other and earlier bill-of-material concepts. It is demonstrated that specific bills-of-material for specific products are not capable of describing the variety that occurs in today's product families. Also modular and variant bills of material show serious shortcomings. Our more recent developments focus on a precise description of the assembly structure of a product family, while still bridging to a description of the product family in commercially understandable parameters.

Keyword Codes: E.1, H.2.8, I.6.5, J.1, J.6
Keywords: Data Structures, Database Applications, Model Development
Administrative Data Processing, Computer-Aided Engineering

1. INTRODUCTION

Bill-of-material processing systems have received renewed attention in recent years. Bills-of-material (BOMs) constitute an important part of product modelling information which is used in industry. However, bills-of-material can easily become untenable for companies that increase their product variety, with the aim of improved customer satisfaction. This may easily lead to a situation where the amount of bill-of-material data grows exponentially. This situation does not only stem from the sheer volume of the data but also from the fact that these data are needed almost everywhere in the factory, e.g. in: marketing, product development, demand forecasting, sales and order-entry, material procurement, assembly, physical distribution, and testing.

Recently, much research work has been published about *generating* bills-of-material as a key idea to manage the growing amount of data. Practical experience is also reported at several places. The aim of this paper is to present our latest developments in bill-of-material research in the context of developments in this area in the last two decades.

In the seventies, solutions focused on smartly structuring the bills-of-material to reduce redundancy in product data. In the early eighties, the first ideas for generating bills-of-material evolved. From that time on new concepts have been developed and improved for modelling product structure data and generating bills-of-material.

In section 2 the basic concepts of traditional bill-of-material systems are explained. Section 3 reviews the concept of *modularisation* of bills-of-material. Section 4 describes an architecture for *generative* bill-of-material processing systems, including the so-called variant bill-of-material system. This section also explains the idea of parametrised product descriptions. Section 5 points out that the traditional implementation of the variant BOM prohibits *recursive* usage of this concept which can inhibit the modelling of product structure data. Section 6 discusses a further development in this field, an approach which does allow recursion. These approaches have a common element, i.e. the characterisation and identification of variants within a product family is based on a list of parameters which should receive a value. Section 7 shows that this property leads to redundancy in certain circumstances and shows our most recent development in bill-of-material systems: a solution which avoids this redundancy, however at the cost of a more complex order-entry interface.

2. TRADITIONAL BILL-OF-MATERIAL SYSTEMS

The way in which a product is built up from purchased parts and semi-finished products (which in turn consist of other purchased and/or semi-finished products) is called the *product structure* of that product. Figure 2.1 depicts the imaginary product structure of a final product *truck*. The relationships between the products in a product structure are so-called *goes-into-relationships*. They represent the fact that a product is consumed in the process of manufacturing or assembling another product. The product *cab* which *is consumed* in the goes-into-relationship with product *truck* is called the *component product*. The product *truck* which *consumes* both the *cab* and the *rear axle*, is called the *parent product* in the goes-into-relationship.

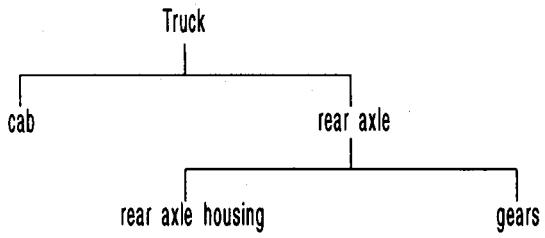

Figure 2.1. Part of the product structure of a truck

The representation of the set of all goes-into-relationships and components of a parent product *P* is called its bill-of-material. Product data are data on single products. Product structure data are data on the goes-into-relationship between two products. In traditional bill-of-material systems these different types of data are commonly represented by means of two separate entity-types, namely:
- *product*
- *bill-of-material relationship*

Tables 2.1 and 2.2 show respectively the entities of the type *product* entities of the type *BOM-relationship* which establish these BOMs. Occurrences of the type product are identified by part numbers. This way of identification is also called *direct identification*, as opposed to *indirect identification which will* will be introduced later in this paper.

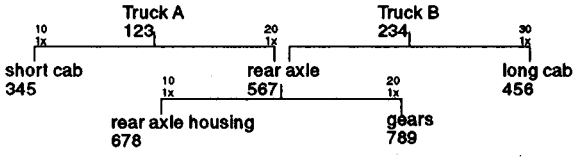

Figure 2.2. Part of the bill-of-material structure of two different trucks A and B

Table 2.1.
Entities of the type product

product part number	product description
123	truck A
234	truck B
345	short cab
456	long cab
567	rear axle
678	r.a. housing
789	gears

Table 2.2.
Entities of the type *BOM-relationship*.

parent product code number	sequence number	component product code number	quantity/per parent
123	10	345	1
123	20	567	1
234	10	456	1
234	20	567	1
567	10	678	1
567	20	789	1

The component product *rear axle* (567) is the parent product in the BOM-relationships with the products *rear axle housing (678)* and *gears (789)* as children. However, in the BOM-relationships of the two truck types, this parent product (567) plays the role of a component product of *truck A* (123) and *truck B* (234). This 'modular storage' principle allows the BOM-relationships of the rear axle to be defined only once, although it is applied in two higher level products. Still the concept of assigning an individual part number to each separate product variant will fail if product variety grows towards hundreds of thousands of product variants. In the seventies many companies which were confronted with these problems tried to find a solution in modularising their bills-of-material. The next section will explain this concept.

3. THE CONCEPT OF MODULARISATION

3.1. The principle

In traditional Material's Requirement Planning (MRP) literature the problem of large product variety is mainly approached from the viewpoint of forecasting and Master Production Scheduling (see (Orlicky et al., 1972), (Orlicky, 1975), (Mather, 1982), (Kneppelt, 1984), (Vollmann et al., 1988)). However it is often argued that modularising BOMs is also a solution to the problem of maintaining very large numbers of different BOMs of final products. To analyse the value of this technique, its key concepts, namely *option*, *feature* and *planning module*, need to be defined precisely.

- An *option* is defined as the representation of a property of a product, such as *with Anti Brake System (ABS)*, *wheel base 3.10 meter*, *rear axle ratio 4.05:1*.

- The concept *feature* is introduced to represent a set of mutually exclusive options (e.g. the feature *ABS* with the options *with ABS*, *without ABS* and the feature *rear axle ratio* with the options *3.31:1*, *3.73:1*, *4.05:1*). The features and options within a product family are defined in such a way that they can be applied to establish unique, identifying product descriptions.

- A *planning module* defines a set of component products which are required to realise certain options of a final product. The common parts of all products within a product family are defined as one planning module. It is often assumed that a planning module equals an option. Under this assumption, options (or planning modules) can be applied to uniquely identify a final product and at the same time to constitute the BOM of that final product.

The modularisation of traditional BOMs involves creating planning modules for given product families, features and options. The technique will be illustrated by a simplified example of trucks. Consider a product family of six trucks. There are two options for the cab type, *long cab*, *short cab* and three *rear axle ratio* options, namely *3.31:1*, *3.73:1* and *4.05:1*. *Cab type* and *rear axle ratio* are the features of the product family. The traditional BOMs of these final products are depicted in Figure 3.1.

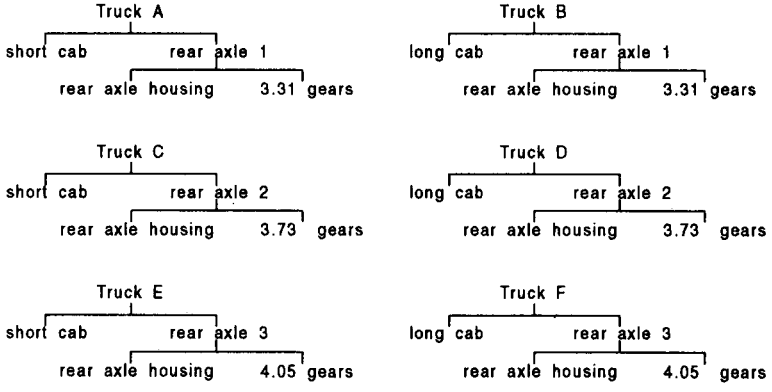

Figure 3.1. Traditional bills of material

The 6 planning modules (3+2+1) and their BOMs which can result after modularisation are depicted in Figure 3.2. Note that planning modules can easily be included in traditional bill of material processing systems, such as discussed in the previous section. The planning modules are directly identified by part numbers, and their special meaning does not lead to fundamental changes in application software.

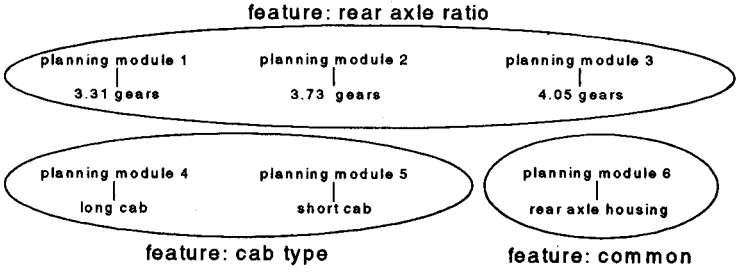

Figure 3.2. Planning modules

It may seem that modularising the BOMs as in Figure 3.2 does not establish a major enhancement because the number of resulting planning modules equals the number of final products in the initial situation. However, in more complex product families with n features each of which has 2 options, modularising the traditional BOM may

result in the reduction from 2^n (2*2*2*....) different final products to 2n+1 options (planning modules). That is, one planning module for each option (2n) and one planning module with all component products which are common (1) to all final products in that product family. In that case substantial simplification is realised. However, we still have the assumption that all component products are selected through a single option, not through a combination of options.

Often an additional BOM-structure is defined for the product family, to support the translation of the forecast volume for the product family as a whole into the volumes for the separate planning modules. This kind of BOM-structure is generally called the percentage BOM. Some standard software systems allow features to be explicitly represented in the percentage BOM, to facilitate the selection of planning modules or options in the order entry process. Figure 3.3. shows the percentage BOM of the product family trucks in the case where features are represented in the structure.

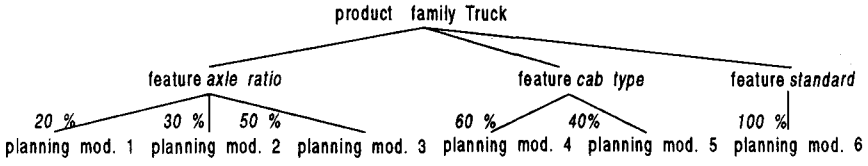

Figure 3.3. Modular / Percentage bill of material

An order dependent BOM is established in the process of selecting the required planning modules or options. For example, when a truck is required with a short cab and with a rear axle ratio of 3.31:1, one appropriate planning module or option must be selected for each of the features, and a single-level BOM is created for the customer order in question (see Figure 3.4).

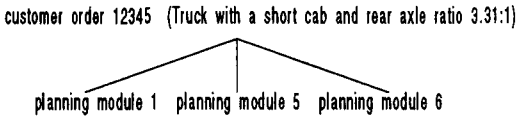

Figure 3.4. Customer order and planning modules

3.2. Implicit conditions in modularisation of BOMs

Succesfull modularisation relies on four quite far reaching conditions:

Condition 1: Planning modules and options coincide

It is assumed that planning modules can be defined in such a way that they are significant units to the market, i.e. they are suitable for forecasting (Wijngaard, 1987) and for order entry (Kneppelt, 1984). In other words it is assumed that each planning module **is** in fact an option and vice versa. In that case the BOM of a planning module defines the full material contents which are required to realise one particular option (Mather, 1982).

Condition 2: Options of different features are independent
It is assumed that in forecasting and order entry, an option can be selected for a feature without any implications for the selection of options for other features. If this would not be assumed, options could neither be forecasted nor selected in the order entry process independently from each other. Note that in that case the question should be raised whether the separate options are significant to the market in the first place.

Condition 3: Options of the same feature do not have lower-level products in common
Wortmann (1987) has pointed out that from a material planning point of view it is important that planning modules can be created which have few or no component products in common. If overplanning is applied to options within the same feature, the products which are common to these options will be overplanned for each of these options, leading to more safety stock than is actually required.

Condition 4: Assembly BOMs can be reconstructed
It is assumed that no additional information is required for assembling a final product from the component products defined in the planning modules of that particular product, or that techniques are available to compensate for the loss of information caused by abolishing higher level BOMs in the modularisation process (Vollmann e.a., 1988), (Kneppelt, 1984) (Orlicky, e.a., 1972), (Sari, 1981). Consider for example the modular structure of Fig. 3.2. If a final product has been specified, it is known that the cab, axle-housing, and gears must be assembled into a final product. However, the information represented by the traditional assembly BOMs, that the axle-housing and gears should be assembled first, is no longer available.

3.3. Conclusions

In modern industries with large varieties of very complex products, these conditions are often not all met. It is often impossible to create planning modules which can play the role of recognisable options for identifying products and the role of defining planning modules for the BOMs of the products. Consequently there is a great need for concepts which allow options and planning modules to be defined separately. This requires the need to bridge options to BOMs. Further, relationships between options are needed to restrict certain combinations between options.

Little attention has been paid to the problem of creating and maintaining assembly BOMs to support final assembly. This type of support will become more important due to the fact that the number of the processes beyond the customer order decoupling point will increase (Van Veen, 1992). In the next section a framework is presented for more flexible product specification and BOM-processing systems.

4. GENERATIVE BOM-SYSTEMS AND THE VARIANT BOM-CONCEPT

4.1. Generative BOM-systems

For generative BOM-processing systems, the concepts of *generic products* and *product variants* need to be introduced. Roughly speaking a generic product (GP) is a set of different products, which have at least one product property in common. Generic products are normally better known as families of product variants. The different products can be described by parameters. These parameters are similar to the features introduced earlier in this paper, but we want to give parameters a more specific meaning: they constitute features which are significant in the market and can be used during order-entry. A product variant represents a particular product. A product variant can be identified by a set of parameter values. Such a set of parameter values is called a *specification*. A specification, S, is valid against a generic product, X, if X contains a product variant which is uniquely identified by S. If a valid specification, *S*, is assigned to a GP, *X*, then that GP will be called *fully specified*. Note that the introduction of generic products and parameters introduces a new way of identifying (fully specified) products. Instead of part numbers, these products are now identified by sets of parameter values in addition to part numbers.

In the previous section, it was concluded that it is often impossible to define (planning) modules which are suitable for both identifying a final product and defining the BOM of that final product at the same time. Therefore, two separate processes are suggested, namely:

1. The product specification process in which a product variant is merely *identified* by means of parameter values: this process is supported by the *Product Specification System* (PSS); the primary function of this system is to evaluate whether a specification *S* is valid or invalid against a particular GP.

2. The BOM generation process in which a BOM is generated for a product variant which is identified by means of parameter values: this process is supported by the *Bill-of-Material Generating System* (BGS); the primary function of this system is to generate a BOM, given a GP and a valid specification. The BGS should not only support the generation of a single-level BOM for a fully specified GP, but also the generation of a multi-level BOM, since controlling the process beyond the customer order decoupling point, may require multiple levels in the product structure.

The distinction of these processes (PSS and BGS) is used for a basic architecture for a new kind of BOM-processing system. In this architecture two core subsystems are distinguished (see figure 4.1). The PSS relies on the set of specifications that define

the members of a GP, described by the so-called set-description of that GP. The data which represents a GP and its set-description will be called *product specification data*. The data consist of parameters, parameter values and constraints on their combinations.

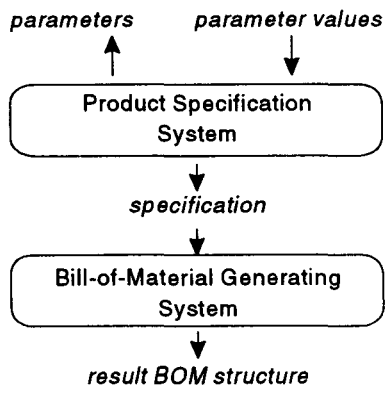

Figure 4.1. BOM processing system

The BGS relies on a so-called source BOM. In a multi-level BOM, each level can contain one or more generic products. This can be a single-level or a multi-level BOM. The *source BOM* of a GP *X* is the set of goes-into-relationships in which *X* is the parent GP. In the source BOM, *X* and its component GPs may contain one *or more* product variants. The multi-level BOM which is obtained for a fully specified GP *X*, by selecting and/or specifying goes-into-relationships and GPs from the multi-level source BOM of X, is called the multi-level result BOM.

The multi-level result BOM of *X* unambiguously represents the product structure of the product variant which is identified by a fully specified *X*. It can be manually derived from a multi-level source BOM but in the case of a BOM Generating System it is assumed that the multi-level source BOM holds data which allows multi-level result BOMs to be automatically generated given a GP and a valid specification.

The above introduction of these concepts will now be illustrated by its most simple implementation: the so called variant BOM.

4.2. The variant BOM-concept

In 1985, Schönsleben described a BOM-processing system which allows a generative multi-level source BOM to be defined. This system is called "Variantengenerator VAR". Schönsleben assumed that a range of product variants can be defined by a set of parameter values such that:

1. the parameters are mutually independent, i.e. no constraints on combinations of parameter values exist, and

2. the required component product variants can be selected unambiguously by determining the values of the parameters for a range of product variants .

Because of the first assumption it is not necessary to support a Product Specification Support system in the Variantengenerator VAR. Schönsleben assumed that

parameters and parameter values can be pragmatically defined in such a way that no constraints on combinations of parameter values occur.

The BOM-generating system VAR is based on a particular kind of generative BOM concept: the *variant BOM concept.* In the variant BOM concept the source BOM of a GP *X* is a set of goes-into-relationships in which *X* is the parent GP. However, this set is partitioned into explicitly defined subsets. Each of these subsets may have one or more members. We will call such a subset a *cluster.* The idea is, that each result BOM should consist of precisely one element out of each cluster. In the terminology of the variant BOM concept, a goes-into-relationship in the source BOM is usually called a *BOM-relationship variant.* The entire set of BOM-relationship variants which have the parent GP *X*, is called the variant BOM of *X* (the equivalent term of the source BOM of *X*). From hereon, a BOM-relationship variant will be shortened to *BR-variant.*

We will illustrate the generative multi-level source BOM according to the variant BOM concept for the six truck variants of Figure 3.1. Figure 4.2. describes the generative multi-level source BOM for the GP representing the six truck variants according to the variant BOM concept.

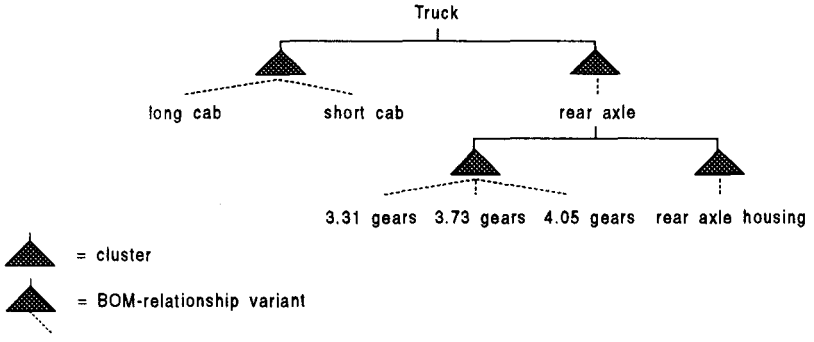

Figure 4.2. Multi-level source BOM

Table 4.1 shows the BR-variants, clusters and conditions (in terms of parameters and parameter values) of this source BOM. The combination of the parent GP and the sequence number identifies a cluster. The datastructure supporting the variant BOM concept is an extension of the datastructure of traditional BOM concepts, with the attributes "BR-variant" and "Condition".

Table 4.1.
BR-variants for the generative source BOM-structure of the six trucks.

parent	sequence number	BR-variant	quantity/ per product	component	condition
truck	10	1	1	short cab	(cab type, long)
truck	10	2	1	long cab	(cab type, short)
truck	20	1	1	rear-axle	
rear-axle	10	1	1	axle housing	
rear-axle	20	1	1	gears set 1	(ratio, 3.31:1)
rear-axle	20	2	1	gears set 2	(ratio, 3.73:1)
rear-axle	20	3	1	gears set 3	(ratio, 4.05:1)

Although Schönsleben assumed that a Product Specification System would not be necessary, others reported on BOM-systems which are based on the variant BOM-concept and which also support a Product Specification System (PSS) to allow the representation of incompatible combinations of parameter values (Digital Equipment GmbH, 1984), (Ashcroft e.a., 1988). In the remainder of the paper the PSS will be considered as part of the variant BOM-concept.

5. LIMITATIONS OF THE VARIANT BOM

The typical variant BOM concept has a number of shortcomings, which are predominantly forthcoming from the fact that parameters, parameter values and constraints can only be associated with the final generic product, thus not with a generic product at an arbitrary level in the product structure.

In the typical variant BOM concept it has been chosen only to allow set-descriptions to be maintained for product families. By this design choice a simple solution emerges, but also a number of difficulties arise. The advantage is that a BOM Generating System which allows multi-level result BOMs to be generated for *final* product variants can be realised with little effort. However, the concept may easily lead to data-redundancy and many problems with engineering changes. These shortcomings will be elaborated below.

5.1. Representing product variety at lower levels in the structure

According to the definition, a GP is a set of similar product variants. Each product variant is identified by a specification. The set of product variants of a GP is defined by a set-description in terms of parameters with their values and constraints on combinations of parameter values. In the typical variant BOM concept set-descriptions are exclusively related to generic products at the top-level of the source BOM. Lower-level GPs cannot be explicitly assigned their own

characteristic parameters and parameter values. A lower-level GP *X can implicitly* represent a set of more than one product variant because different multi-level result BOMs can be generated for it. Consider for example the lower-level GP *rear axle* in the source BOM-structure of the *truck* (Figure 4.2). The GP *rear-axle* implicitly represents three rear-axle variants because three different result BOMs can be generated for it. As a consequence of this simplification a lower-level GP cannot be assigned a valid specification and as such be treated as an independent entity. It always needs a reference to a generic product family in which it is applied.

5.2. Data redundancy

Another consequence concerns data-redundancy. The set-descriptions of the GP-families contain considerable data redundancy in parameters, parameter values and constraints because some of this data is exclusively related to lower-level product variants. For example, in practice a rear-axle variant may be determined by as many as eight parameters. For this family of rear-axles a number of constraints on combinations of parameter values exists. These constraints are independent of the final product variant in which the rear-axle is applied. Since in the variant BOM-concept this information of the truck cannot be explicitly stored once for the GP *rear-axle*, it must be recorded redundantly in the set-description of each generic product family in which such a rear axle could occur.

The obvious solution is to allow independent set-descriptions to be maintained for arbitrary GPs, or in the terminology of the variant BOM concept, to allow generic product families to be defined at arbitrary levels in the generative source BOM-structure. However additional concepts need to be developed, to be able to support the generation of BOMs for individual product variants. Another requirement of a new generative BOM-concept is that the source BOM should be defined in such a way that the transparency of the product structure is improved. For that purpose, the source BOM should reflect information of the kind that each product variant of GP *X* has a component product variant of a lower level GP Y. For example each variant of GP *truck* has a rear axle of GP *rear-axle*. The generic BOM-concept described in the next section allows this kind of definition of GPs and should improve the transparancy of the product structure.

6. A GENERIC BOM-SYSTEM WHICH CAN BE USED RECURSIVELY

The analysis of the variant BOM-concept acknowledged the importance of allowing the definition of GPs at lower levels in the product structure, independently of their application in higher level GPs. For example, it should be possible to define a set of

rear-axles independently of their application as a component in a truck. The GP *rear-axle* may even comprise one or more rear-axle variants which are nowadays not a component in any higher level variant.

6.1. Goes-into-relationships in the generic bill-of-material concept

Traditionally, goes-into-relationships are uniquely identified by a single key-attribute of the parent and a sequence number for the goes-into-relationship. In line with the objectives of the generic BOM-concept the aim is to define GPs, being sets of product variants, and to define one source BOM for a GP *X*, which applies to all product variants which are members of *X*. Goes-into-relationships should therefore be defined between *sets* of product variants, i.e. between GPs, instead of between individual product variants.

It will be assumed that GPs *are* identified by a single key-attribute, namely GP number. In the generic BOM-concept, a goes-into-relationship will therefore be identified by the parent GP number and a sequence number.

Consider the following product variants, which can be described with parameters and parameter values:

Parameters	:	{cab length, cab type, ratio, ABS}	
Parameter values	:	cab length:	{short, long}
		cab type:	{day cab, space cab, sleeper cab}
		ratio:	{3.31:1, 4.05:1}
		ABS:	{with, without}

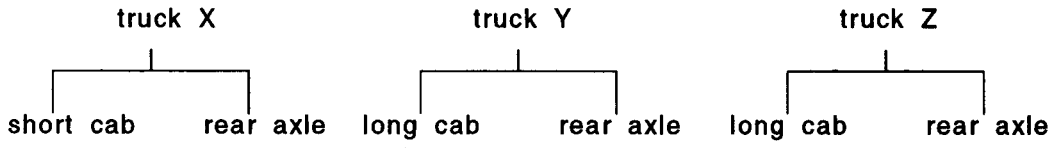

Figure 6.1. Specific BOMs of truck variants X, Y and Z

The product variants were initially identified by part numbers 123, 234, 345, 456, 567, and 678 (Table 6.1.). In the generic BOM concept, these variants are identified with parameters and parameter values (Table 6.2.).

Table 6.1.
Initial product variants and their part numbers.

product number	product description
123	truck X
234	truck Y
345	truck Z
456	short cab
567	long cab
678	rear axle

Table 6.2.
Relationships between product variants and their product specifications

product variant	product specification data
truck X	{(cab type; day cab)}
truck Y	{(cab type; space cab)}
truck Z	{(cab type; sleeper cab)}
short cab	{(cab length; short)}
long cab	{(cab length; long)}
rear axle	{(ABS; with), (ratio; 3.31:1)}

Recall that in the generic BOM-concept, these single key-attributes may not exist and that product variants are identified by a product specification. In table 6.3, the relationships of specific bills of material for trucks X, Y and Z are given. For bigger families, the redundancy in this bill-of-material concept will be much bigger than in a bill-of-material concept where individual product variants are described with parameters and parameter values.

Table 6.3.
The traditional goes-into-relationships of the bills-of material

parent product variant	sequence number	component product variant	quantity/per parent
123	10	678	1
123	20	456	1
234	10	678	1
234	20	567	1
345	10	678	1
345	20	567	1

Now we will introduce a slightly different example. Assume the following GPs with their set-descriptions are distinghuished:

Table 6.4.
A list of GPs and their set-descriptions.

GP	set-description parameters	set-description parameter values
truck	{cab type:	{day cab, space cab, sleeper cab}}
cab	{cab length:	{short, long}}
rear axle	{ABS:	{with},
	ratio:	{3.31:1}}

To be able to reconstruct the individual BOM of a product variant 234 from *truck* the goes-into-relationships in which *truck* is a parent GP must be retrieved to find its component GPs. Each of these component GPs specifies one or more candidate

product variants which may be the component in the BOM of 234. *Truck* has two component GPs, namely *rear axle* and *cab*. *Rear axle* is a GP and is already fully specified. No further specification is required: the BOM of each product variant which is a member of *truck* will contain the single product variant which is a member of *rear axle*.

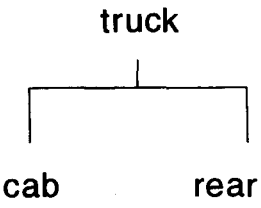

Figure 6.2. BOM relationships between generic products

6.2. Conversion functions

Determining the required product variant of the generic component product *cab* is more complicated. Obviously the goes-into-relationship between *truck* and *cab* does not provide enough information to determine unambiguously whether the product variant identified by {(cab length; short)} or the product variant identified by {(cab length; long)} is to become a component in the result BOM of *truck* {(cab type, space cab)}. For this purpose additional information must be available in the goes-into-relationships of the source BOM. This information should constitute a function which produces the required component product variant given a parent product variant. In other words, if a parent product variant has been identified by a full specification for its GP, then this function should guarantee that one full specification is available for the component GP, thereby identifying one component product variant. This function will be called the conversion function. It is the core of the process of generating result BOM-structures from a source BOM-structure.

The conversion function consists of so called *conversion rules* which define deterministic relationships between parameter values of product specifications of parent product variants and component product variants. Conversion rules must be formulated in such a way that, for each full specification of the generic parent product (GPaP), precisely one full specification for the generic component product (GCoP) is produced. In addition, if the specification of the GPaP is valid, then the specification produced for the GCoP must also be valid.

The conversion rules which define the relationships between parameter values of GPaP *truck* and GCoP *cab* are listed in Table 6.5.

Table 6.5.
The conversion rules between the parameters of GPaP *Truck* and GCoP *CAB*.

GPaP	Seq nr	GCoP	Conversion Rules
Truck	10	Cab	(cab type, day cab) ==> (cab length, short)
			(cab type, sleeper cab) ==> (cab length, long)
			(cab type, space cab) ==> (cab length, long)

Conversion rules could be related to single generic products or to the combination of a GPaP and a GCoP, i.e. a BOM-relationship. In this generic BOM-concept, the

latter has been chosen to support the case in which different GPaPs of a GCoP have the same parameters but different requirements with respect to the parameter values of their GCoP.

6.3. Generating a result BOM

The result BOM for a product variant can be generated by assigning the specification of that product variant to the GP it belongs to. In the generation process the source BOM of the GP will be exploded. Parameters of GPs at lower levels in the source BOM must already be assigned a value (for example, the rear axle in Table 6.4) or must receive a value from their parent GPs by using the conversion function. If the conversion functions and set-descriptions are correct the generation process will result in a result BOM-structure with correct variants for the fully specified GPs.

7. REDUNDANCY IN THE GENERIC BOM

The generic BOM-concept explained above allows a relatively easy product specification process. The concept of the conversion function allows that each GP can be assigned the parameters which are best suitable for product specification purposes, also from an order entry viewpoint. Also it was implied that all information required for the product specification of a GP *X* is explicitly recorded for *X*. To be more precise: any parameter or constraint regarding GCoPs of a GP *X* are in one way or another reflected in the set-description of *X* to guarantee that a valid specification for *X* will result in fully specified CoPs. For example the parameters of the rear axle in Table 6.4 do not need to be reflected in the set-description of any higher level GP because it is already fully specified. However if the parameter *ABS* would have had two values for the rear axle, the set-description of *X* and the conversion function in the source BOM should take into account that the parameter *ABS* must be assigned a value. The same applies to constraints on parameter values. Consider the following set-description for a rear axle:

ABS: {with, without}
ratio: {3.31:1, 3.73:1, 4.05:1}
NOT((ABS, without) AND (ratio, 4.05:1))

Then the set-descriptions of all GPs in which the rear-axle is a component product should not only reflect the two parameters of the rear-axle but the constraint as well. This will of course result in large data redundancy in the source BOM. The redundancy will become obvious if the parameters of a GP are also used for its

higher level GPs. However if complex conversion rules are applied, the redundancy may not be that easily recognisable, but it does exist. Hence the price for a relatively simple product specification process is the data-redundancy that occurs in the source BOM. The alternative is not to explicitly record the complete set-description of a GP, but to restructure the set-description and distribute the parts of the description over the generic products in the structure.

7.1. A distributed set-description

To explain this principle, a new term has to be introduced, namely *generic primary products*. Generic primary products are found at the lowest level of a generic bill-of-material structure. A generic primary product consists of a number of variants of a primary product. A generic primary product (GPP) is therefore a set of all the variants of a primary product. All generic products which are not GPPs are called: *generic secondary products* (GSPs). These are found on higher levels in the generic bill of material structure. Note carefully that the distinction between generic primary products (GPPs) and generic secondary products (GSPs) is not the same as the distinction between generic component products (GCoPs) and generic parent products (GPaPs), made earlier in this paper. The term GPP is used for each product which has no generic components, whereas the term GCoP is used for each product which acts somewhere as a component.

All variants of a generic secondary product (GSP) are assembled in a more or less similar way out of GPPs or other GSPs. However the variety exclusively originates from the GPPs. A consequence of this is that the traditional assembly relationships between the individual product variants can be aggregated into assembly relationships for the generic products to which the individual product variants belong. In order to identify variants within a generic assembly we could assign to each product variant a code number. However, we would not do this for all possible variants without having orders for them, as that would add tremendously to the data redundancy.

In our approach to avoid redundancy, the part of the set-description of GP X which solely comes forth from representing the set-description of a GCoP Y, will only be explicitly defined for Y. Just before product specification the set-description of X is reconstructed from the lower-level set-descriptions. It will often be impossible to reconstruct the *entire* set-description of *X* from the set-descriptions of its GCoP, because constraints may exist which apply not only to one GCoP, but to the parameter values of several GCoP (e.g. the constraint **NOT ((cab length, short) AND (ABS, without))**). These constraints must be defined at the point at which the GCoPs in question are applied together, i.e. at the first common GPaP (in this case *Truck*).

We use the example of the truck product family to illustrate the different aspects of the generic BOM-system based on the principle of defining set-descriptions as low as possible in the generic bill of material structure. The common product structure of all the truck variants is graphically represented in Figure 7.1. In this example, the generic products are coded with three-digit numbers with the letter G for generic. The variation of the truck is caused by the variation of the cab and the rear axle. Similarly, the variation of this rear axle is caused by the variation of the gears.

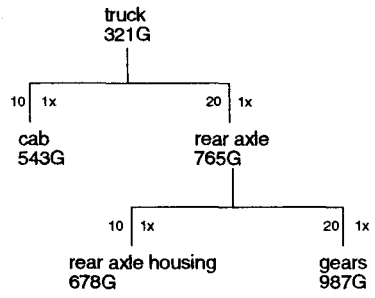

Figure 7.1. Common product structure of a truck

A *cab* variant in the example is a primary product. The generic primary product *cab* is a set of *cab* variants. Each *cab* in the GSP *truck* can be delivered long or short. Thus there are two variants for the *cab* available. The GSP *rear axle*, as described earlier in section 7, has five variants. This means that the number of *truck* variants is equal to ten (assuming that all combinations are allowed).

A *rear axle* variant with ABS and ratio 4.05 only differs from a *rear axle* variant without ABS and ratio 3.73 in the primary generic product *gears*. The *housing* is the same for both, which implies that the variety of the generic product *rear axle* entirely depends on the variety of its generic component product *gears*.

7.2. Identification of generic primary products

The variants within a GPP can be either identified directly by a code number or alternatively by parameters in addition to a GPP number, in other words, a specification (list of parameters and constraints) assigned to each GPP. Each parameter has a number of parameter values. A GPP variant can be identified by any valid specification (combination of parameter values) against the constraints in the set-description. For example, the GPP cab can be described by the code number 543G and the parameter cab-length with the permitted values (short, long). A particular variant is obtained by choosing one permitted value for the parameter cab-length.

7.3. Identification of generic secondary products

Since the bills-of-material of the variants within a GSP differ in content, but are the same with respect to structure, the variation within a GSP comprises the variation within its GPPs. We will use this for yet another form of identification of product variants, that is, indirect identification. Indirect identification means that a variant of a GSP is described by the (multi-level) bill of material of the GSP, together with the parameter values of the generic products in this BOM. A

specification of a GSP variant is complete once all the GPPs at the beginning of the paths, leading to the GSP, have been specified. We shall illustrate this with the rear axle example.

For identification of a *rear axle* variant, it is sufficient to identify the variant with ABS and ratio of 4.05 of the GPP *gears*. Further, indirect identification requires the indication of the path between the GPP and the GSP. This is depicted in Table 7.1 under the heading sequence number. The *rear axle housing* is not necessary for the *rear axle* identification because there is a single *rear axle housing* variant in all the rear axles.

Table 7.1.
Identification of a rear axle with ABS and ratio of 4.05

Generic product number	Sequence number	Description	Parameters	Parameter value
765G	1	rear axle		
987G	. 20	gears	ABS	WITH
			RATIO	4.05

In general, for identification of variants, we can distinguish four different types for identification of variants within a product family. In the generic bill-of-material concept based on direct specification of GCoPs, we use three of these types. These are summarised below, illustrated by the truck example.

Table 7.2.
Three different identification types used in the generic bill-of-material concept

identification	**without parameters**	**with parameters**
direct	empty cab	gears with ABS and ratio of 4.05
	543	987G[ABS=with and ratio=4.05]
indirect	not used in the generic BOM concept	rear axle variant 1 765G .20 987G[ABS=with and ratio=4.05]

7.4. Using one GP in one generic bom structure more than once

Since different variants can be generated from one generic product (occurring in different places in the generic bill of material structure), the information must be contained which variant is used in which parent variant.

The next example illustrates that this is supported by indirect identification. It relates to the identification of a *truck* with a *long cab* and two *rear axles.* The first *rear axle* has *ABS* and the other one has not. Both *rear axles* have the same *ratio* of 3.73. The generic bill-of-material for this *truck* can be shown in the multilevel explosion in Table 7.3. Since the GPP *rear axle* appears twice in this bill-of-material, it must be specified two times. In this example we will select *ABS* for the first *rear axle* and leave the second *rear axle* without *ABS.*

Table 7.3.
Identification of a truck with two different rear axles

Generic product number	Serial number	Description	Parameters	Parameter value
321G	1	truck	CAB	
543G	. 10	cab	LENGTH	LONG
765G	. 20	rear axle		
987G	. . 20	gears	ABS	WITH
			RATIO	3.73
765G	. 30	rear axle		
987G	. . 20	gears	ABS	WITHOUT
			RATIO	3.73

7.5. Concluding remarks

The generic bill-of-material concept provides the user with a means of describing a large number of variants with a limited amount of data, while leaving the product structure unimpaired. For this reason generic bills-of-material are particulary suitable for describing variants of a product family. Generic bills-of-material can be used for describing both final products and components. Generic components that are already recorded can be reused in new product families, that make use of these components. Other aspects of the generic bill-of-material have not been dealt with in this paper:

- description of interfaces between parent variants and component variants
- recording non-permitted parameter value combinations
- inheritance mechanism between parent and component
- use of numerical parameters
- use of algorithms for deriving parameters values from other parameter values

For the future, our research will focus on the following topics:

- using result BOMs for configuration management
- using generic bills of material as an inter-company communication framework
- creating generic bills of material in the engineering process
- relating generic bills of material and parametrised CAD systems
- implementing generic bills of material in object-oriented database systems

Literature

Ashcroft, M., Barber, T., Flynn, R., Levy, H., 1988, Regaining competitive advantage through an integrated approach to product definition, Willis Faber Award for Manufacturing Effectiveness, Leyland-DAF

Digital Equipment GmbH, 1984, VAX-profi Zentrale Stammdaten Benutzershandbuch, version 1.3. AA-AV31B-TE, München

Hegge, H.M.H. and Wortmann, J.C. 1991, Generic bills of material: a new product model. Production Economics, Issues and Challenges for the 90's

Kneppelt, L.R., 1984, Product structuring considerations for master production scheduling, Production and Inventory Management, first quarter 1984

Mather, H. 1982, Bills-of-material, Recipes and Formulations, Wright Publishing Company Inc.

Mather, H. 1986, Design, bills of material and forecasting the inseparable threesome. Production Inventory Management, 90-107

Orlicky, J.A. and Plossl, G.W. and Wight, O.W. 1972, Structuring the bill of material for MRP. Production Inventory Management, 13(4)

Orlicky, J.A. 1975, Material Requirements Planning, McGraw-Hill

Schönsleben, P., 1985, Flexibele Produktionsplanung und Steuerung met dem Computer, CW-Publikationen, München

Veen, E.A. 1992, Modelling product structures by generic bills of material. Elsevier Science Publishers

Vollmann, T.E., Berry, W.L., Whybark, D.C., 1988, Manufacturing planning and control systems, Dow Jones-Irwin, Illinois

Wijngaard, J., 1987, Production control in a consumer electronics factory, Engineering Costs and Production Economics, nr. 12

Wortmann, J.C., 1987, Information systems for assemble-to-order production: an application, Engineering Costs and Production Economics, nr. 12