Configuration for mass-customisation and e-business

Ying-Lie O[1]

Abstract. Mass-customisation and e-business impose new requirements on trading. In mass-customisation, products are variations of configurations in a product family. Model-based configuration can be seen as a combination of product modelling and solving the configuration problem.

The product model can be presented using modelling constructs. Different constructs allow different kinds of selection. Instead of requiring the customer to choose from technical specifications, understandable characteristics are used.

The problem solving part is finding the configuration by a combination of selection and approximate matching. An incremental approach allows customer interaction. At each step, the customer may select components and characteristics. In the resulting matching, the list of components and non-matching characteristics are indicated. The proposed approach is attempted to be simple, such that customers can understand the proposed solutions of the specified characteristics.

1 INTRODUCTION

Nowadays, there is an increasing demand on products to customers order. These products are not fully tailored in the conventional way, but are "variants". This is called mass customisation, involving variations in a product family. There are three types of compositions:

- *Pick-to-order* is providing the components selected by the customer. The customer is responsible for making the product ready to use. Example: Home AV (audio-video) systems are sound systems, television sets, and combinations of these intended for non-professional use in an average size living room.
- *Assemble-to-order* is assembling of products according to customers configuration from components. Example: Office computers are computer systems for one or more persons for office work in an office or at home.
- *Make-to-order* is the production to customers specification according to existing product type definitions. Example: digital printworks are the preparation, printing, and postprocessing of a document or artwork to a printed product using professional digital printing equipment.

Mass-customisation and e-business impose new requirements on trading. Customers must be able to specify their requirements online without knowing details of product design and technical specifications. Mostly, the customer has to select a specific product type first, and then select the desired parts from lists. In reply, a quote with the matching product is offered. This situation is far from being customer-friendly. It is even difficult to compare a number of similar products. In a shop, products are on display along with the most important specifications.

[1] University of Twente, Enschede, email: ying@cs.utwente.nl

E-business should provide additional value by employing AI. Instead of directly selecting the parts, the customer should be able to specify characteristics of the intended product. In reply, a quote with several top-most approximately matching products are offered. The products are ordered according to their "goodness of match", indicating the non-matching characteristics. This approach requires a suitable product modelling, and a suitable problem solving method.

2 PRODUCT AND CONFIGURATION MODELS

Product modelling and configuration have been around for some time to support manufacturing. It is used for design, production, and service of the product during its life-time. Product data management and product models describe the product at certain stages "as is". The configuration problem is finding the combination of product components such that it matches the product requirements.

2.1 Product models

There are many product modelling methods, some are specifically tailored for a certain industry or family of products. Most product models represent the hierarchical composition of the product. The bill of materials (BOM) gives the composition of an individual product. Each component is represented by a code or serial number. Associating the components of the BOM with its functional descriptions [9] gives a more detailed description. The BOM can be extended to support choices in assemble-to-order products [3] by including the list of possible components.

STEP (Standard for the Exchange of Product Model Data) is a conceptual standard for representing the configuration of individual products based on standard data. The STEP method and the specification language EXPRESS is useful for the definition of general models of products [2], including product variants [8]. The EXPRESS language can be mapped to the object-oriented UML (Unified Modeling Language) [1].

2.2 Configuration models

Configuration systems are mainly intended for supporting the manufacturing process, in particular product design. These systems are mainly focussed on problem solving [5]. There are many product configurations methods [10]. Early systems were rule-based systems, but the number of rules tend to explode, and the relation between rules and components of the product are not as clear.

In model-based systems, there is a separation between knowledge about components and the way it is used. Commonly used logic-based methods are description logics and the constraint-based satisfaction problem. A combination of these methods is a rule-based language equipped with description logics [11].

Model-based configuration can be seen as a combination of product modelling and reasoning to solve the configuration problem. A structured modelling procedure [4] by decomposition and analysis of assembly relations allow detailed specification.

A conceptual model for the configuration of mass-customisable products employing the UML notation has been proposed by [6]. It includes consistency and validity checks and can be automatically translated to an executable logic representation for a knowledge base.

3 CONFIGURATION MODEL FOR MASS-CUSTOMISATION

Product configuration is a combination of product modelling and solving the configuration problem. In most configuration systems, customers must enter product specifications and select components from detailed lists. This requires sufficient knowledge about the product. Instead, customers should be able to specify their requirements. To get more precise, a terminology is employed. The terminology, or more general *ontology* can be extracted from existing e-business data [7] such as online catalogs. Unfortunately, most of the results regard terminology for component names, component brands, and technical specification.

The product *characteristics* is an intermediate between the customer requirements and product specification
$specification \longrightarrow characteristics \longleftarrow requirements$.
Each component has a number of characteristics. The characteristics specified by the customer are matched to that of the components. The resulting configuration is an instance of a product type.

This model is specifically intended for mass-customisation, thus limited to variations of a product family. In most product configurators, the complete set of characteristics must be specified beforehand. Here, an incremental approach at selection level is proposed. This approach allows customer interaction and update of the requirements. At each step, the customer may select components and characteristics. The proposed approach is attempted to be simple, such that customers can understand the proposed solutions of the specified characteristics.

3.1 General model of product families

Product families are modelled in a hierarchical way. Each product family consists of several clearly identifiable product types. This should limit the number of possible variations, and thus reducing the complexity. The properties of the general model are:

- A *product family* consists of mutually independent product types that may consist of other product types.
- The structure of a *product type* is a composition of subassemblies and parts. These components may be ready parts or produced according to specification.
- Matching of the user requirements to the product specification is performed through an intermediate terminology, the *product characteristics*.
- Each component has characteristics associated to its functionality and technical specification.
- The customisation is in the selection of components and characteristics. A component may be mandatory or optional.

The general model of product families as depicted in Figure 1 employs the UML notation for static structure diagrams. The modelling constructs are described below.

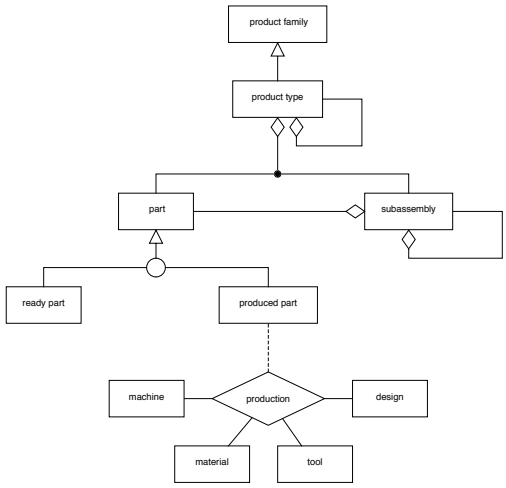

Figure 1. General model of the configuration of a product family consisting of metaclasses

- A *rectangle* represents an object *class* by its name. A class consists of objects, while a *metaclass* indicated by the << >> signs consists of classes. Metaclasses are used to represent the product and component families. Product and component models are given by classes, the products are the object instances.
- The second compartment in a divided rectangle gives the *attributes* of the class. The *product characteristics* are represented by attributes.
- The hollow triangle △ and circle ○ on the connecting line represent shared *generalisation*. Child components inherit properties from the parent component. The letter "d" (disjoint) in the circle indicates that there may be only one child in an instance. In the top layers of a product model, a *product family* is divided into *product types* using disjoint generalisation. A product type may consists of other product types.
  Selection components of a part is represented by generalisation. A *part* may be ready parts or produced parts.
- The hollow diamond ◇ and bullet • on the connecting line define *aggregation*. An expandable *composition of components* is given by aggregation. A *product type* consists of subassemblies and parts. A *subassembly* may consist of subassemblies and parts.
- The solid diamond ◆ and bullet • on the connecting line represent composition, a strong form of aggregation into a *composite class*. In product models a *composite component* is built of subcomponents that solely belong to that component and the composition yields its functionality as a whole. This specific property may be valid for some subassemblies, and is therefore not indicated as metaclass in the general model.
- A large diamond connected with lines to rectangles represent an *association*. The *association class* is given by a rectangle connected with a dashed line. The production of a *produced part* is represented by an association of components required in the production.
- The permitted number of components is specified by the multiplicity as a range of natural numbers. Thus, $1..n$ means 1 to n, 1..1 means exactly 1, 0..1 means at most 1.

The general product model in Figure 1 consists of metaclasses only, therefore the << >> signs are left out. Specific composition properties that apply on classes are therefore not visible.

3.2 Product characteristics

As an attribute, a characteristic has an identifying generally meaningful name and a domain of possible values as a user-friendly translation of the technical specification.

Characteristics are preferably objective, and the domain consists of comparative terms. This makes the characteristics invariant to rapid changes, for instance in technology. $quality\{plain, good, high\}$ in printing gives the resolution of the printer and grain size of the paper; $size\{small, medium, large\}$ related to a PC disk gives the storage size with respect to current technology; $speed\{medium, fast, super\}$ specifies the PC computing power, and is related to the technical specification of the CPU frequency and internal memory.

Inheritance and association of the characteristics are determined by the modelling constructs. *Child components* in a *generalisation* inherit properties from the parent component and may have additional specialisation characteristics
$\{child\ chars\} = \{parent\ chars\} \cup \{specialised\ chars\}$.
A *composite component* has all the characteristics of its subcomponents $\{composite\ chars\} = \{component_1\ chars\} \cup \cdots \cup \{component_n\ chars\}$.
Similar to the composite component, a *composition of components* has all the characteristics of its selected components. Also similar to the composite component, the *produced part* has all the characteristics of its associated components.

Except for inheritance properties, the scope of the characteristics may be *global*, or *local*. In the global mode, all characteristics of the same name get the same value
$char_x = a \Rightarrow \forall$ components $component\ char_x = a$.
In the local mode, it is limited to the selected component. In a generalisation, characteristics regarding the specialisation are always local. The mode may be changed during the selection process. For instance, the desired colour of a product may be set global at the beginning, and set to local regarding a specific part.

3.3 Selection and matching

The problem solving part is finding the configuration of the desired product by a combination of *selection* and *approximate matching*.

The approximate matching problem is formulated as finding the components with the most matching characteristics to the user requirements. This will result in a list of products, that may have several non-matching characteristics. Customisation by selection occurs at three levels:

1. Class-level in *aggregation* constructs by selection of a combination of desired components.
2. Class-level in *generalisation* constructs by selection from a list of possible components.
3. *Object*-level in all classes by selection of an object instance in each class.

Matching the user requirements to the product characteristics are set-based operations. Set union $\cup$ gives all the characteristics, and set intersection $\cap$ gives the matching characteristics.

The incremental approach allows (de)selection of both components and characteristics at each step. Non-matching characteristics are indicated, allowing the customer to modify the selections.

The selection of components yields the associated characteristics and their value range.
$\forall$ selected components $\{total\ chars\} = \cup \{total\ chars\} \cup \{component_1\ chars\} \cup \cdots \cup \{component_n\ chars\}$.
On the other hand, the selection of characteristics gives the matching components and non-matching characteristics.
$\forall$ selected characteristics $\{total\ chars\} = \cup \{total\ chars\} \cup \{char_1\} \cup \cdots \cup \{char_n\}$.
Given the $\{total\ chars\}$ $\exists$ matching components with $\{component\ chars\} \cap \{total\ chars\} \neq \emptyset$. Non-matching characteristics are not present in the set of selected characteristics $\{nonmatch\ chars\} : \{component\ chars\} \notin \{total\ chars\}$.
This process is alternately repeated on components that have a path to the selected product type.

The selection starts with choosing from an index of product families and product types. As in a large department store, customers are usually able to find the department of a particular product family.

1. Class-level top-down matching to determine the product types. The requirements of the product types generally regard the intended use of the product.
2. Class-level characteristics from the selected product types are retrieved and presented for selection. Product types may be deselected.
3. Then for each selected product type, advance downwards on class-level. This involves tracing the connected paths to the product types. The selection procedure is carried out for aggregation constructs and generalisation constructs.
4. Class-level bottom-up analysis of association classes and composition classes that are connected to the selected classes.
5. Class-level characteristics from the selected components are retrieved and presented for selection. At this stage, characteristics may be altered locally only for a particular component.
6. Object-level selection from the list of matching objects of all remaining classes.
7. Determine a list of matching product compositions.

In the above, in the first three steps, the characteristics are global, and may become local in the last four steps.

3.4 Reconfiguration

Reconfiguration is altering an existing product configuration for replacement, upgrading, or changing the functionality. There are two modes of reconfiguration:

Addition or replacement of components that does not change the remaining of the existing configuration such as addition of a component in an aggregated class, replacement of a child component in a generalisation class, or replacement of an object instance. Most of the functionality and global characteristics remain the same.

Changes that affect the configuration such as upgrades of an earlier version of a product type typically regard composition classes and aggregation classes. This can be considered as a change of the product type. Therefore, an "upgrade model" containing the possible replacements is required.

4 EXAMPLES

Each of the composition type is illustrated by a familiar example. To avoid cluttered diagrams, the models are only partly presented and detailed. For the same reason, the characteristics that mostly yield the "customisation" are indicated by empty compartments. Also, the multiplicities are left out.

To illustrate the matching process, the most relevant characteristics are presented in tables. In the first table, the characteristics are

described. In the next table, the selection process is illustrated by showing the requirements characteristics entered by the customer, the matching product type and associated characteristics.

4.1 Home AV systems

"Home AV systems" is an example of pick-to-order products. It is

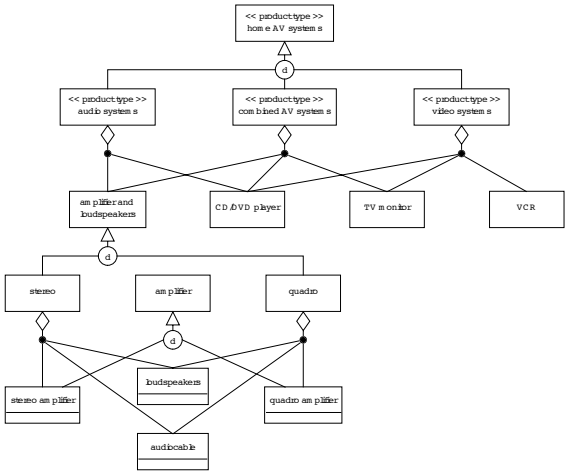

Figure 2. Part of the configuration model of home AV systems

a product type of the product family "AV systems". Usually, components have matching connectors. In Figure 2 part of this model is presented, along with the most relevant part of the characteristics in Table 1.

Table 1. Part of the characteristics of home AV systems

class	construct multiplicity	characteristics constraints
home AV systems	product type	use = non-professional
audio systems	product type	AV = audio
AV systems	product type	AV = (audio, video)
audio systems	aggregate	AV = audio
CD/DVD player	part	type {CD, DVD}
television	part	AV = video
amplifier and	generalisation	type {stereo, quadro}
loudspeakers	1..1	power {low, med, high}
stereo	aggregate 0..1	type = stereo
quadro	aggregate 0..1	type = quadro
amplifier	generalisation	quality {plain, good, high}
		power {low, med, high}
loudspeaker	part 2..2	quality {plain, good}
		power {low, med}
		connector = RL pin
stereo amplifier	part 1..1	quality {plain, good, high}
		power {low, med, high}
		connector = RL pin
audiocable	part 2..2	connector {RL pin, coax}
		connector1, connector2

In pick-to-order, there usually is an abundance of choices on object-level. Therefore, additional characteristics such as brand, and colour may help to restrict the choices.

1. The selection starts at the level of home AV systems.
2. The choice of AV = audio yields 2 choices, one with a non-matching characteristic.
3. Advancing downwards is rather straight-forward, allowing the selection in "amplifier and loudspeakers" and "CD/DVD players".
4. There are no association classes and composition classes.
5. Selection of characteristics, for instance a local value is selected: loudspeaker.quality = good.
6. Selection of stereo amplifiers, loudspeakers, and CD players with the selected characteristics.
7. The matching products are stereo amplifiers, "loudspeakers (2)", "audio cables (2)", and "CD players" with matching connectors.

Table 2. Example of the selection and matching of home AV systems

requirements characteristics	class	characteristics of components
AV = audio	audio systems	AV = audio
	AV systems	AV = audio
deselect	AV systems	
type = CD	CD/DVD player	type = CD
type = stereo	amplifier and	type = stereo
	loudspeakers	power {low, med, high}
type = stereo	stereo	type = stereo
quality = good	loudspeaker	quality = good
power = med		power = med
–		connector = RL pin
quality = high	stereo amplifier	quality = high
–		power = med
–		connector = RL pin
–	audiocable	connector1 = RL pin
–		connector2 = RL pin

4.2 Office computers

A familiar example of assemble-to-order products are "office computers" a product type of the product family "Computers". This prod-

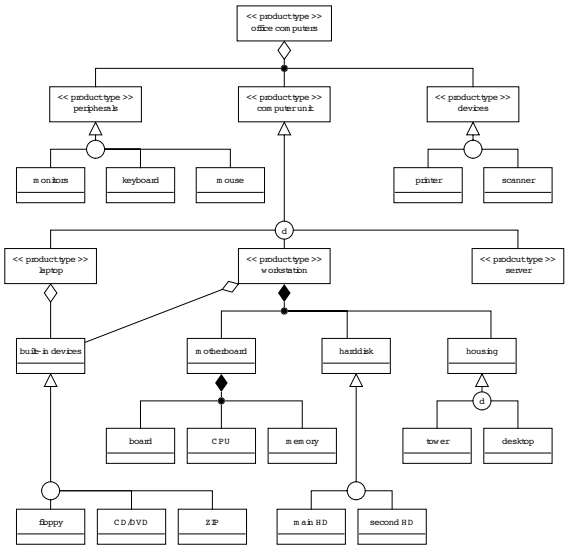

Figure 3. Part of the configuration model of office computers

uct type is further divided into specific product types. In addition to

the assemble-to-order part, there are pick-to-order parts to combine the configured system with other devices. It is partly represented in Figure 3, with the most relevant characteristics in Table 3.

Table 3. Part of the characteristics of office computers

class	construct multiplicity	characteristics constraints
office computers	product type	use = office work
computer unit	product type	main unit
peripherals	product type	addition
devices	product type	addition
laptop	product type	type = client
workstation	product type	place = fixed, type = client
server	product type	place = fixed, type = server
workstation	composition	type
built-in devices	generalisation 0..*	type {flop, CD/DVD, ZIP} slot $\leq$ housing.slot
motherboard	composition 1..1	type
board	part 1..1	
CPU	part 1..1	speed {med, fast, super}
memory	part 1..4	speed {med, fast, super}
harddisk	generalisation 1..2	main HD = 1..1 second HD = 0..1
main HD	child 1..1	size {small, med, large} speed {med, fast}
second HD	child 0..1	size {small, med} speed {med, fast}
housing	generalisation 1..1	type {tower, desktop} slot 1..*
tower	child	second HD = 0..1, slot = 5
desktop	child	second HD = 0, slot = 2

Table 4. Example of the selection and matching of office computers

requirements characteristics	class	characteristics of components
main unit place = fixed type = client	computer unit workstation	main unit place = fixed type = client
– type = CD/DVD type = ZIP	built-in devices	slot = 2 type = CD/DVD type = ZIP
– – –	workstation motherboard board	type type
speed = fast speed = fast	CPU memory	speed = fast speed = fast
second HD size = large speed = fast size = med speed = med	harddisk main HD second HD	second HD = 1 size = large speed = fast size = med speed = med
– –	housing tower	type = tower second HD = 1, slot = 5

In this example, only the assemble-to-order part is tracked.

1. The selection starts with “computer unit”.
2. The choice of place = fixed, type = client yields the only product type “workstation”.
3. Advancing downwards only regards the “built-in devices”. The choices of “floppy”, “CD/DVD”, and “ZIP” require 2 slots.
4. Composition classes that are connected to “workstation” are “motherboard”, “harddisk”, and “housing”.
5. In the selection of characteristics, the “second HD” has less characteristics than the “main HD”.
6. Object-level selection from all remaining classes. There may be additional constraints, such as the number of slots for “built-in devices” and “hard disks” that fit in the “housing”.
7. Find matching “workstations” with fast performance, 2 “harddisks”, “tower housing”, and the desired “built-in devices”.

4.3 Digital printworks

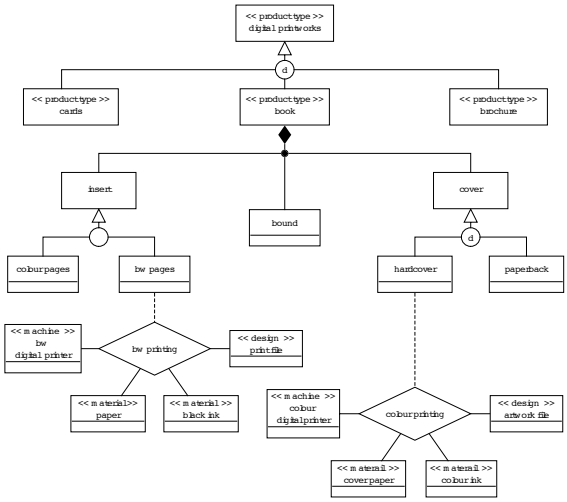

Figure 4. Part of the configuration model of digital printworks

Table 5. Part of the characteristics of digital printworks

class	construct multiplicity	characteristics constraints
digital printworks	product type	printing = digital ink = cartridge
book	product type	
card	product type	
brochure	product type	
book	composition	quality {plain, good, high}
insert	generalisation 1..2	colour {colour, bw} quality {plain, good, high}
cover	generalisation 1..1	type {hard, paperback} quality {plain, good, high}
bound	part 1..1	if cover.type = paperback then bound = adhesive if cover.type = hard then bound = {stitch, sew}
bw pages	association 1..*	colour = bw
printfile	design	quality = good paper size {A3, A4, A5}
paper	material	quality {plain, good, high} paper size {A3, A4, A5}
black ink	material	
digital printer	machine	colour = bw
cover	association 1..1	colour = colour
artworks	design	quality = good paper size {A3, A4, A5}
cover paper	material	quality {plain, good, high}
colour ink	material	colour {3-colour, 6-colour}
digital printer	machine	colour = colour

"Digital printworks" is an example of make-to-order products. It is a product type of the product family "Press works". Printed pages are generated by printing machines from the file delivered by the user. As partly presented in Figure 4, the product type is further divided into different types. The most relevant characteristics associated to the figure are given in Table 5.

Table 6. Example of the selection and matching of digital printworks

requirements characteristics	class	characteristics of components
book	book	
quality = good	book	quality = good
colour = bw	insert	colour = bw
–		quality = good
type = hard	cover	type = hard
–		quality = good
–	bound	bound = {stitch, sew}
–	bw pages	colour = bw
–	printfile	quality = good
paper size = A5		paper size = A5
quality = good	paper	quality = good
–		paper size = A5
black	black ink	
–	digital printer	colour = bw
–	hard	colour = colour
–	artworks	quality = good
–		paper size = A5
quality = high	cover paper	quality = high
colour= 3-colour	colour ink	colour= 3-colour
–	digital printer	colour = colour
bound = sew	bound	bound = sew

The product type "book" is taken as an example to illustrate the selection and matching in a composite component and produced parts.

1. The selection of the product type "digital printworks" has been made.
2. The product type "book" is selected.
3. Going downwards only regards this product type.
4. The components in the composite component "book" are related by the way it is produced. The associated components in the "bw pages" and "cover" represent material and settings of the printing process.
5. Selection of the characteristics of the "bound" is limited by the choice of the "cover".
6. Object-level selection basically is the selection of the "paper".
7. Matching products mainly regard the choices in the printing process and the use of paper and ink. There may be some matching between the printing machines and paper type and size.

5 CONCLUSION

A simple incremental product configuration method for mass-customisation has been proposed. The product model consists of modelling constructs for each configuration feature. For customer requirements, characteristics are used instead of technical specification. The characteristics selected by the customer are then used to find matching components that have a connection path to the selected product type. The approximate matching is performed using simple set-based methods. The method allows non-matching characteristics.

From the examples it is found that in some cases business rules are still needed. It should be further investigated how such rules can be replaced by a set construct.

A possible improvement of the matching process, is by ranking the characteristics in a sequence of importance as a partial order of sets $\{charset_1\} \succeq \ldots \succeq \{charset_n\}$.
The selection and matching then becomes a combination of tracking the paths of selected and associated components, ordering of the characteristics, and matching these ordered characteristics.

Further research should also include consistency analysis of the proposed solutions. The method should then be improved to handle conflicting situations. It should also be investigated, whether additional constraints or rules are needed to maintain consistency in conflicting situations.

Another important aspect in the application of this method is model management. It is expected that if the modelling constructs are applied consistently, then the model definition part will not pose any problems. However, problems are expected in the definition of the characteristics. Even using a proper ontology, maintenance is still elaborate. Updates of ontologies, and merging different ontologies is still a research topic at this very moment.

It is also of interest to include different ways of representing the modelling constructs for specification and implementation. Also, the possibilities to extend this simple model to a full product design model should be considered.

ACKNOWLEDGEMENTS

The author is indebted to Rob van de Weg for his valuable detailed comments, and colleagues of the group of Information Systems for their inspiring discussions.

The UML figures were drawn using TCM (Toolkit for Conceptual Modelling) partly developed by this group, available at $www.cs.utwente.nl/tcm$ under the GNU General Public License.

REFERENCES

[1] F. Arnold and G. Podehl, 'Best of both worlds - A mapping from EXPRESS-G to UML', in *The Unified Modeling Language, UML'98 - Beyond the Notation. First International Workshop, Selected Papers*, eds., J. Bézivin and P.-A. Muller, volume 1618 of *LNCS*, pp. 49–63. Springer Verlag, (1999).

[2] M. Ashworth, M.S. Bloor, A. McKay, and J. Owen, 'Adopting STEP for in-service configuration control', *Computers in Industry*, **31**, 235–253, (1996).

[3] J.W.M. Bertrand, M. Zuijderwijk, and H.M.H. Hegge, 'Using hierarchical pseudo bills of material for customer order acceptance and optimal material replenishment in assemble to order manufacturing of non-modular products', *Int. J. Production Economics*, **66**, 171–184, (2000).

[4] P.Y. Chao and T.T. Chen, 'Analysis of assembly through product configuration', *Computers in Industry*, **44**, 189–203, (2001).

[5] B. Faltings and E.C. Freuder, 'Special issue on configuration', *IEEE Intelligent Systems*, **13**(4), 29–85, (1998).

[6] A. Felfernig, G.E. Friedrich, and D. Jannach, 'Conceptual modeling for configuration of mass-customizable products', *Artificial Intelligence in Engineering*, **15**, 165–176, (2001).

[7] A. Kayed and R.M. Colomb, 'Extracting ontological concepts for tendering conceptual structures', *Data & Knowledge Engineering*, **40**, 71–89, (2002).

[8] T. Männistö, H. Peltonen, A. Martio, and R. Sulonen, 'Modelling generic product structures in STEP', *Computer-Aided Design*, **30**(14), 1111–1118, (1998).

[9] T. Männistö, H. Peltonen, T. Soininen, and R. Sulonen, 'Multiple abstraction levels in modelling product structures', *Data & Knowledge Engineering*, **36**, 55–78, (2001).

[10] D. Sabin and R. Weigel, 'Product configuration frameworks – a survey', *IEEE Intelligent Systems*, **13**(4), 42–49, (1998).

[11] T. Soininen and I. Niemelä, 'Developing a declarative rule language for applications in product configuration', in *PADL'99 Practical Aspects of Declarative Languages Proc. First International Workshop*, ed., G. Gupta, volume 1551 of *LNCS*, pp. 305–319. Springer, (1998).