Product-driven Workflow Design

W.M.P. van der Aalst[1,2], H.A. Reijers[2,3], and S. Limam[1]
[1] Department of Technology and Management, Eindhoven University of Technology,
PO Box 513, NL-5600 MB, Eindhoven, The Netherlands.
[2] Department of Mathematics and Computing Science, Eindhoven University of Technology,
PO Box 513, NL-5600 MB, Eindhoven, The Netherlands.
[3] Deloitte & Touche Bakkenist, PO Box 23103, NL-1100 DP, Amsterdam, The Netherlands.
E-mail: w.m.p.v.d.aalst@tm.tue.nl, hreijers@win.tue.nl, s.limam@tm.tue.nl,

Abstract

In manufacturing the interaction between the design of a product and the process to manufacture this product is studied in detail. Consider for example Material Requirements Planning (MRP), which is mainly driven by the Bill-Of-Material (BOM). For information-intensive products such as insurance, loans, permits, and many other services, the relationship with the supporting workflow process is often neglected. Typically, the workflow processes are designed without careful consideration of structure and characteristics of the product. In this paper, we present a method for designing efficient and effective workflows based on the products generated by the process rather than subjective interpretations of managers, consultants, and IT experts.

1. Introduction

From a logistical point of view, there are many similarities between administrative processes and production processes (cf. Platier [13]). Both kinds of processes, focus on the routing of work (workflow) and the allocation of work to resources. In a production system, the products are physical objects and the principal resources are machines, robots, humans, conveyor belts and trucks. In an administrative process the products are often informational (e.g. documents) and most of the resources are human. Although there are many similarities, there are also some logistical aspects in which an administrative process differs from a typical manufacturing process [1]:

- *Making a copy is easy and cheap.* In contrast to making a copy of a product like a car, it is relatively easy to copy a piece of information (especially if it is in electronic form).
- *There are no real limitations with respect to the in-process inventory.* Informational products do not require much space and are easy to access (especially if they are stored in a database).
- *There are less requirements with respect to the order in which tasks are executed.* Human resources are flexible and there are few technical constraints.
- *Quality is difficult to measure.* What is the quality of the decision to accept an insurance claim?
- *Quality of end-products may vary.* A manufacturer of cars has a minimal quality level that any product should satisfy. However, in an administrative process it might be attractive to skip certain checks to reduce the workload.
- *Transportation of electronic data is timeless.* In a network information travels at the speed of light.
- *Production to stock is seldom possible.* Every product is unique, therefore it is difficult to produce in advance. It is not possible to process an insurance claim before it arrives.

Nevertheless, the two types of processes have a lot in common. Consider for example performance indicators such as throughput time, waiting time, service level and utilization. These performance indicators play a prominent part in both domains.

In manufacturing, the Bill-Of-Material (BOM) is used to drive the production process [12]. Consider for example Material Requirements Planning, often referred to as MRP-I, which determines the production schedule based on the ordered quantifies, current stock, and the composition of product as specified in the BOM. Contemporary Enterprise Resource Planning (ERP) systems such as SAP also take resource availability into account and use more refined algorithms. Nevertheless, production is driven by the structure of the product.

Administrative processes generate information-intensive products such as mortgage loans, driving permits, customs declarations, salary payments, etc. Typically, support for the process execution is provided by workflow management systems [9][10][14]. In contrast to manufacturing, the relation between the product and the process is seldom made explicit in administrative processes. Clearly, the interaction between the product and the process is in the back of everyone's mind. However, there is not a standardized way to describe the product and the workflow process is usually designed without proper consideration of the product. Consider for

example the processing of insurance claims. The product is basically a decision: Either the claim is accepted (followed by a payment) or the claim is rejected. All kinds of data elements may play a role in making this decision. One can think of these data elements as raw materials or subassemblies. The workflow process should manufacture the decision while taking criteria such as average flow time, service level, handling costs, and product quality into account. As mentioned, the latter may be hard to measure.

Driven by management principles such as Business Process Reengineering (BPR) [7][11], the focus has shifted from data and organization to processes. The problem is not this focus but the way processes are designed [15]. In many BPR projects, the existing process is taken as a starting point. In workshops, small groups of consultants, managers and specialists try and improve such a process into a new design. As a result of this approach, the processes are usually sub-optimal, incomplete, subjective, and at a too abstract level ("The devil is in the details"). A careful analysis of the product in terms of data elements and a design which is driven by the structure of the product and the characteristics of the data elements, can solve most of these problems. Therefore, we propose a method for *product-driven* workflow design.

2. Product/data structures for workflow processes

The BOM used in manufacturing is a tree-like structure with the end product as root and raw materials and purchased products as leafs. In the resulting graph, the nodes correspond to products, i.e., end-products, raw materials, purchased products, and subassemblies. The edges are used to specify composition relations (i.e., *is-part-of* relations). The edges have a cardinality to indicate the number products needed. Figure 1 shows the simplified BOM of a car which is composed of an engine and a subassembly. The subassembly is composed of four wheels and one chassis.

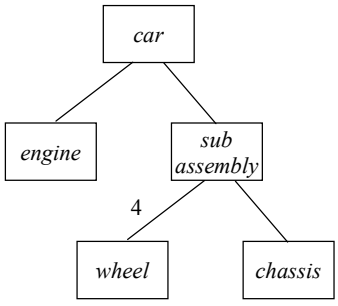

Figure 1 : The BOM of a car.

For information-intensive process, the traditional BOM is not very useful. As was indicated in the introduction there are several differences between informational products and physical products. First, making a copy of information in electronic form is trivial from the process perspective since it takes hardly any time or resources to do this. Therefore, cardinalities make no sense. Second, the same piece of information may be used to manufacture various kinds of new information. Therefore, also non-tree-like structures are possible. For example, the age of an applicant for life insurance may be used to estimate the health risks and the risks of work related accidents. Finally, there are no physical constraints and therefore there are typically multiple ways to derive a piece of information. For example, health risks may be estimated using a questionnaire or a full medical examination. These observations lead to the following product/data model.

Product/data model

A product/data model is a tuple (D, C, *pre*, *F, constr, cst, flow, prob*):

- D: set of data elements, $\underline{\text{top}} \in \text{D}$,
- C: set of constraints, $true \in \text{C}$,
- $pre : \text{D} \rightarrow \mathcal{P}(\mathcal{P}(\text{D}))$,
  - $\text{R} = \{(p, c) \in \text{D} \times \text{D} \mid c \in \bigcup_{es \in pre(p)} es\}$ is connected and acyclic,
  - $\forall (p,c) \in R : c \neq \underline{\text{top}}$,
  - $\forall e \in \text{D} : \varnothing \in pre(e) \Rightarrow pre(e) = \{\varnothing\}$,
- F : set of production rules, $F = \{ (p, cs) \in \text{D} \times \mathcal{P}(\text{D}) \mid cs \in pre(p) \}$,
- $constr : \text{D} \times \mathcal{P}(\text{D}) \not\rightarrow \text{C}$,
  - $\text{dom}(constr) = F$,
  - $\forall e \in \text{D} : pre(e) = \{\varnothing\} \Rightarrow constr(e, \{\varnothing\}) = true$,
- $cst : \text{D} \times \mathcal{P}(\text{D}) \not\rightarrow \mathbb{N}$,
  - $\text{dom}(cst) = F$,
- $flow : \text{D} \times \mathcal{P}(\text{D}) \not\rightarrow \mathbb{N}$,
  - $\text{dom}(flow) = F$,
- $prob : \text{D} \times \mathcal{P}(\text{D}) \not\rightarrow (0..1]$
  - $\text{dom}(prob) = F$,
  - $\forall e \in \text{D} : pre(e) = \{\varnothing\} \Rightarrow prob(e, \{\varnothing\}) = 1$,
  - $\forall (p, cs) \in F : constr(p, cs) = true \Rightarrow prob(p, cs) = 1$.

The product/data model expresses relations between data elements. These relations can be used to produce a value for the data element $\underline{\text{top}}$. The *pre* function yields for each data element d zero or more ways to determine a value for d. If we suppose for data-elements $d, e, f \in \text{D}$, that $\{e, f\} \in pre(d)$, then a value of d may be determined on basis of values of e and f. We say that $(d, \{e, f\})$ is a *production rule* for d. The expected probability that this production rule yields a result for d is given by $prob(d, \{e, f\})$.

Furthermore, the rule can only be applied if *constr*(*d*,{*e*, *f*}) evaluates to *true*. The cost of producing a value for *d* with this rule is specified by *cst*(*d*,{*e*, *f*}). Likewise, its flow time (throughput time) is specified by *flow* (*d*, {*e*, *f*}). Note that a data element *d* for which holds that *pre*(*d*) = {∅} is special; it is called a *leaf*. No other data elements are required to determine the value of a leaf. There are also no constraints to determine the value of a leaf; the probability to determine a leaf's value is 1. Note that there may be costs associated with obtaining the value of a leaf – just as is the case for any other data element. Also note that for any data element the probability that a value can be determined is 1 if there are no constraints to determine its value (i.e., *true*).

Note that the probabilities given in a product data model are assumed to be independent. Because values can be less than 1, it is generally not ensured that the data element top can be determined for a given set of data element values. For example, suppose in a real-life situation that there are two alternative data elements that can be used to construct a value for some top, both with a probability of 0.9. Even if the values of both elements are available there is still a (1-0.9)·(1-0.9) = 0.01 probability that no value for the top element can be determined.

3. An example

An example of a product/data model is depicted in Figure 2. All nodes in this figure correspond to data elements. Arcs are used to express the *pre* relation that is in use to decide whether a candidate is suitable to become a helicopter pilot in the Dutch Airforce. Unlike the BOM in Figure 1 there are no cardinalities in effect. On the other hand, constraints, cost, flow time, and probabilities are associated with production rules. The meaning of the data elements is as follows:

- *a*: suitability to become a helicopter pilot,
- *b*: psychological fitness,
- *c*: physical fitness,
- *d*: latest result of suitability test in the previous two years,
- *e*: quality of reflexes,
- *f*: quality of eye-sight.

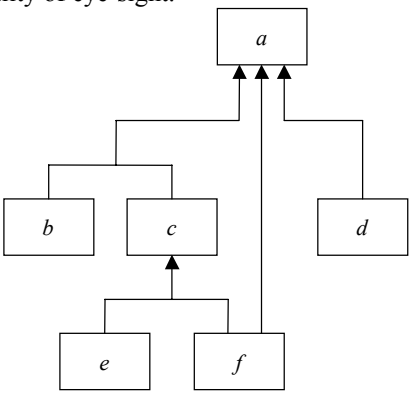

Figure 2 : Helicopter pilot product/data model.

One of the things that follows from the figure is that there are three different production rules for the top element *a*. The suitability of a candidate can be determined on basis of:

- The results of the psychological test (*b*) and the physical test (*c*),
- The result of a previous test (*d*), or
- The candidate's eye-sight quality (*f*).

The relations *constr*, *cst*, *flow*, and *prob* for this example are as shown in Table 1. If *a* is a data element, the value of *a* is denoted with **a*.

From this table it follows that obtaining values for leafs is much more time-consuming in this example than other values. This represents a common phenomenon that actions that involve communication with external parties take more flow time than internal actions.

Furthermore, it can be concluded that if a candidate's eyes are worse than –3.0 or +3.0 dioptries this information can be used as a direct *knock-out* [3] for the test result. The probability that this will happen for an arbitrary case is 0.4. Note that each production rule for the top element can be a knock-out for a specific case.

X	***constr*(*x*)**	***cst*(*x*)**	***flow*(*x*)**	***prob*(*x*)**
(*a*,{ *b*, *c* })	*True*	80	1	1.0
(*a*,{ *d* })	**d* ∈ {suitable, not suitable}	10	1	0.1
(*a*,{ *f* })	**f* < -3.0 or **f* > +3.0	5	1	0.4
(*b*,{ ∅ })	*True*	150	48	1.0
(*c*,{ *e*, *f* })	*True*	50	1	1.0
(*d*,{ ∅ })	*True*	10	16	1.0
(*e*,{ ∅ })	*True*	60	4	1.0
(*f*,{ ∅ })	*True*	60	4	1.0

Table 1 : Relations *constr*, *cst*, *flow*, and *prob* for testing a helicopter pilot candidate.

4. Conformance

Up to this point, no specifications are given of the production rules, other than their signature. For each element of the *pre* relation such a specification should be available to produce working process models. However, to consider the optimality of process models the exact specifications are not relevant. What is relevant is that a given process model conforms with the product/data model under consideration.

Process model
A process model PM on a product/data model (D, C, *pre*, *F*, *constr, cst, prob*) is defined by (T, *prod*, Ω) where:
- T is a set of tasks,
- *prod*: T → *F*, the production rule applied in the task,
- Ω is a set of firing sequences.

An execution sequence $r \in \Omega$ is a sequence $t_1 t_2 \ldots t_k$ of tasks with $k \in \mathbb{N}\backslash\{0\}$, such that for any $1 \le i \le j \le k$ holds that $t_i, t_j \in$ T and $t_i = t_j \Rightarrow i = j$ (single executions of each task).

Conformance
A process model (T, *prod*, Ω) conforms to the product/data model (D, C, *pre*, *F, constr, cst, flow, prob*) if and only if for each sequence $r = t_1 t_2 \ldots t_k \in \Omega$, $k \in \mathbb{N}\backslash\{0\}$, holds that:

1. $\forall 1 \le i \le k, (p,cs) \in F : \left[prod(t_i) = (p,cs) \Rightarrow \forall c \in cs : \exists 1 \le j < i, ds \in \mathcal{P}(D) : prod(t_j) = (c,ds) \right]$,
2. $\exists 1 \le i \le k, cs \in \mathcal{P}(D) : prod(t_i) = (\underline{top}, cs)$.

The first requirement ensures a proper order of the application of production rules. The last requirement guarantees that the top element *may* be produced.

5. Design criteria

Given a product/data model there are many conformant process models. Different subsets of tasks executed in some order may lead to the production of the top element. When designing product-driven workflows, the following three design criteria need to be taken into account: (1) *quality*, (2) *costs*, and (3) *time*. Costs and time are defined according to the functions *cst* and *flow*. In this paper we use a rather narrow definition of quality. Quality is defined as the probability that the value of the top element can be determined. Note that quality depends on the structure of the graph (i.e., function *pre*) and the probability that a production rule leads to a value. To allow for a formal definition of these design criteria we introduce the notion of a *plan*.

Plan
Let (*D, C, pre, F, constr, cst, prob*) be a product/data model. Any subset *S* of *D* is called a plan.

One can think of a plan as a sub-graph of the graph denoting the product/data model. The elements of *S* are the data elements that should be produced. The set {*a, d*} is a plan corresponding to the product/data model shown in Figure 2. In this plan the production rules $(d,\{\emptyset\})$ and $(a,\{d\})$ are executed in some order. The set {*a, e*} is also a plan although this plan will never lead to a value for data element *a*. For any given plan, we can determine the probability that a value for the top element is determined.

Quality of a plan
Let (*D, C, pre, F, constr, cst, prob*) be a product/data model. The quality of a plan $S \subseteq D$ is defined as $p\_quality(S) = q_{top}$ for all $d \in S$:

$$q_d = 1 - \prod_{(d,cs) \in F} \left(1 - \left(prob(d,cs) \cdot \prod_{e \in cs} q_e \cdot \delta(e) \right) \right), \quad \text{(i)}$$

where $\delta(e) = \begin{cases} 0, & e \notin S \\ 1, & e \in S \cup \{\emptyset\}. \end{cases}$

The quality of a plan is the probability that the value of the top element can be determined successfully assuming that all production rules only referring to elements in *S* are executed. Note that for any production rule (*p*, *cs*) ∈ *F* holds that all elements in *cs* should be part of the plan in order to contribute to q_p.

Consider the product/data model shown in Figure 2 and three plans S_1 = {*a*, *d*}, S_2 = {*a*, *b*, *c*, *e*, *f*} and S_3 = {*a*, *e*}. For plan S_1 holds that the quality of this plan is $p\_quality(S_1) = q_{top} = q_a$. According to formula (i), $q_a = 1-(1-prob(a,\{d\}).q_d.\delta(d))$ with $q_d = 1-(1-prob(d, \{\emptyset\}))\cdot q_\emptyset \cdot \delta(\emptyset)) = 1$. So, $p\_quality(S_1) = q_a = 0.1$. Similarly, for plan S_2, $p\_quality(S_2) = 1$ and for plan S_3, $p\_quality(S_3) = 0$.

Costs of a plan
Let $S \subseteq D$ be a plan. The costs of *S* are:

$$p\_csts(S) = \sum_{(p,cs) \in F} cst(p,cs) \cdot \delta(p) \cdot \prod_{e \in cs} \delta(e) \quad \text{(ii)}$$

The costs of a plan are simply given by the sum of all production rules costs relevant for the plan. Note that again it is assumed that production rule (*p, cs*) is executed if {*p*} ∪ *cs* is a subset of plan *S*. These costs can be interpreted as the maximum costs that are associated with the execution of a plan.

The costs of plans S_1 = {*a*, *d*}, S_2 = {*a*, *b*, *c*, *e*, *f*} and S_3 = {*a*, *e*} are as follows. For plan S_1 the only production rules relevant are (*a*, {*d*}) and (*d*, {∅}). So, according to equation (ii), $p\_csts(S_1) = cst(a, \{d\})\cdot\delta(a)\cdot\delta(d) + cst(d, \{\emptyset\})\cdot\delta(d)\cdot\delta(\emptyset) = 20$ (see Table 1). Similarly, $p\_csts(S_2)$ = 405 and $p\_csts(S_3) = 60$.

Flow time of a plan

The actual time required to produce all data elements of a plan depends on the order in which the production rules are executed. In a worst-case scenario where all production rules of the plan are executed sequentially, the total flow time is:

$$\sum_{(p,cs)\in F} flow(p,cs)\cdot\delta(p)\cdot\prod_{e\in cs}\delta(e) \qquad \text{(iii)}$$

By executing some of the production rules of the plan in parallel, the total flow time can be reduced.

Consider plan $S_4 = \{a,b,c,d,e,f\}$. Assume that this plan is executed in the following order: $(d,\{\varnothing\})$, $(a,\{d\})$, $(f,\{\varnothing\})$, $(a,\{f\})$, $(e,\{\varnothing\})$, $(c,\{e,f\})$, $(b,\{\varnothing\})$, $(a,\{b,c\})$. Then the average worst case $flow\_time(S_4) = flow(a, \{b, c\})\cdot\delta(a)\cdot\delta(b)\cdot\delta(c) + flow(a, \{f\})\cdot\delta(a)\cdot\delta(f) + flow(a, \{d\})\cdot\delta(a)\cdot\delta(d) + flow(b, \{\varnothing\})\cdot\delta(b)\cdot\delta(\varnothing) + flow(c, \{e, f\})\cdot\delta(c)\cdot\delta(e)\cdot\delta(f) + flow(f, \{\varnothing\})\cdot\delta(f)\cdot\delta(\varnothing) + flow(e, \{\varnothing\})\cdot\delta(e)\cdot\delta(\varnothing) + flow(d, \{\varnothing\})\cdot\delta(d)\cdot\delta(\varnothing) = 76$ time units. Now suppose that the production rule $(a, \{d\})$ leads to a value for a, then the $flow\_time(S_4) = flow(a, \{d\})\cdot\delta(a)\cdot\delta(d) + flow(d, \{\varnothing\})\cdot\delta(d)\cdot\delta(\varnothing) = 17$ time units only. So, the average flow time of a plan may be much smaller because a value for a data element can be obtained before all elements of the plan are derived.

6. Product-driven reengineering rules

In the previous section we introduced the notion of a plan. Given a plan S, it is easy to calculate its quality $p\_quality(S)$ and the associated costs $p\_costs(S)$. To calculate the total flow time of a plan is more complex, because one has to make assumptions about the order in which the production rules are executed. Note that a plan is not a process model: it is merely a subset of data elements. However, the notion of a plan and the criteria $p\_quality(S)$ and $p\_costs(S)$ can be used for a heuristic approach towards product-driven workflow design. The heuristic uses the fact that $p\_quality(S)$ and $p\_costs(S)$ allow for the definition of a cost optimal plan given a quality level.

Cost optimal plan

Let (*D, C, pre, F, constr, cst, prob*) be a product/data model and $q \in [0, 1]$ be a quality level. Plan $S \subseteq D$ is cost optimal if and only if

1. $p\_quality(S) \geq q$, and
2. $\forall S' \subseteq D: p\_quality(S') \geq q \Rightarrow p\_csts(S') \geq p\_csts(S)$.

Consider R the set of plans that can be derived from the product/data model of Figure 2. $R = \{S_1, S_2, S_3, S_4, S_5\}$ where $S_5 = \{a, f\}$. Assume $q = 0.8$. We already know the quality level of plans S_1, S_2 and S_3: $p\_quality(S_1) = 0.1$, $p\_quality(S_2) = 1$, and $p\_quality(S_3) = 0$. It is easy to calculate the quality level of plans S_4 and S_5: $p\_quality(S_4) = 1$ and $p\_quality(S_5) = 0.4$. Only plans S_2 and S_4 fulfill condition (1). For those plans, costs are $p\_csts(S_2) = 405$ and $p\_csts(S_4) = 425$. According to the definition of cost optimality, it appears that plan S_2 is the cost optimal plan.

A cost optimal plan gives the least costly subset of data elements that needs to be calculated to obtain a given quality level. Note that the costs associated to such a plan are the maximal costs, i.e., the costs that are made if all corresponding production rules need to be calculated. However, a knock-out may result in a value for the top element before the whole plan is executed. Therefore, it is important to order the production rules. The ordering of the production rules can be based on the time criterion or on the cost criterion mentioned in the previous section. The two extremes are:

1. *Breadth-first*. Start with the leaf nodes in the plan and execute as many production rules in parallel as possible.
2. *Depth-first*. Start with the part of the plan which has the best quality/cost ratio, i.e., execute the production rules sequentially and start with the most promising branches first.

Assuming sufficient capacity the breadth-first approach optimizes the process with respect to flow time but at high costs (in principle all production rules associated to the plan are executed). The depth-first minimizes the average costs but may result in substantial longer flow times. For the breadth-first approach there is no need to order the activities executing the production rules: The graph structure is used to maximize parallelism. For the depth-first approach these activities need to be ordered sequentially. To decide on the ordering of activities within a given plan S, we introduce the notion of a cost optimal chain of plans.

Cost optimal chain of plans

Let (*D, C, pre, F, constr, cst, prob*) be a product/data model and $q \in [0, 1]$ be the required quality level. Let $S \subseteq D$ be cost optimal with respect to q and $1 \leq K \leq |S|$. $S_1, S_2, \ldots, S_K$ is a *cost optimal chain of plans* if and only if

1. $S = S_K \supseteq S_{K-1} \supseteq S_{K-2} \supseteq \ldots \supseteq S_1$
2. $\forall\ 1 \leq i \leq K, S' \subseteq S: p\_quality(S') \geq p\_quality(S_i) \Rightarrow p\_csts(S') \geq p\_csts(S_i)$

A cost optimal chain gives the order in which the production rules should be executed. First execute the activities associated to S_1 in parallel, then the activities associated to S_2 but not S_1 in parallel, etc. The number K gives the number of steps in this sequential process. If K equals the number of elements of the set S, then the plan is executed sequentially, i.e., a pure depth-first strategy. If K=1, a pure breadth-first strategy is used.

If we consider the example of Figure 2 and a quality level $q = 0.8$, the cost optimal plan was $S_2 = \{a, b, c, e, f\}$. A breadth-first strategy leads to executing production rules $(e, \{\varnothing\})$, $(f, \{\varnothing\})$ and $(b, \{\varnothing\})$ in parallel. Then execute production rule $(c, \{e, f\})$ and finally $(a, \{b, c\})$. A depth-first strategy leads to executing production rules $(b, \{\varnothing\})$ then $(e, \{\varnothing\})$ and $(f, \{\varnothing\})$ in parallel, then $(c, \{e, f\})$ and finally $(a, \{b, c\})$.

A cost optimal plan identifies the class of process models that are best suited to implement a given product specification. Depending on the preferred ordering of production rules, a specific process can be derived.

8. Conclusion

In this paper, we presented a new way of looking at workflow process design. By taking the product as a starting point many of the problems mentioned in the introduction can be avoided. A formal model, a corresponding notion of conformance and a quantification of design criteria have been given. Finally, a heuristic approach towards product driven workflow design has been presented.

By now, Deloitte & Touche Bakkenist applied the method presented in this paper four times in client engagements in 2000 and 2001. Business processes of a large Dutch bank and a national social security agency have been reengineered. In each occasion, at least a 50% cost reduction and a 30% flow time reduction has been achieved, while maintaining the same quality level.

The method of product-driven workflow design is a promising new way of executing BPR initiatives. A practical boost would come from the development of tools to model product/data models and derive process models. Further research can lead to the incorporation of other relevant design criteria, techniques, and algorithms.

In Section 2, we compared the classical BOM with our product/data model for workflow processes. Current ERP systems allow for so-called *generic/variant* BOM's [6][8]. This way, it is possible to deal with large product families having millions of variants [16]. As is shown in [2], this concept can be translated to workflow processes facing ad-hoc and structural changes. It is interesting to extend the approach presented in this paper to deal with product families. Another topic for future research is the issue of variable cardinalities. In many cases, multiple instances of the same type of data element are required, e.g., multiple eyewitness reports for an insurance claim. The product/data model presented in Section 2 does not take these cardinalities into account.

References

[1] W.M.P. van der Aalst. On the automatic generation of workflow processes based on product structures. *Computers in Industry*, 39:97-111, 1999.

[2] W.M.P. van der Aalst. Generic Workflow Models: How to Handle Dynamic Change and Capture Management Information. In M. Lenzerini and U. Dayal, editors, Proceedings of *CoopIS'99*. IEEE Computer Society Press, pages 115-126, 1999.

[3] W.M.P. van der Aalst. Reengineering Knock-out Processes. *Decision Support Systems*, 30(4):451-468, 2001.

[4] W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors. *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2000.

[5] J.A. Buzacott. Commonalities in Reengineered Business Processes: Models and Issues. *Management Science*, 42(5):768-782, 1996.

[6] F. Erens, A. MacKay, and R. Sulonen. Product modelling using multiple levels of abstraction - instances as types. *Computers in Industry*, 24(1):17-28, 1994.

[7] M. Hammer and J. Champy. Reengineering the corporation. Nicolas Brealey Publishing, London, 1993.

[8] H.M.H. Hegge. *Intelligent Product Family Descriptions for Business Applications*. PhD thesis, Eindhoven University of Technology, Eindhoven, 1995.

[9] S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation.* International Thomson Computer Press, 1996.

[10] P. Lawrence, editor. *Workflow Handbook 1997, Workflow Management Coalition*. John Wiley and Sons, New York, 1997.

[11] R.L. Manganelli and M.K. Klein. The Reengineering Handbook: A Step-by-step Guide to Business Transformation. Amacom, 1996.

[12] A. Orlicky. Structuring the bill of materials for MRP. *Production and Inventory Management*, pages 19-42, Dec 1972.

[13] E.A.H. Platier. *A logistical view on business processes: BPR and WFM concepts (in Dutch)*. PhD thesis, Eindhoven University of Technology, Eindhoven, 1996.

[14] G. Poyssick and S. Hannaford. Workflow Reengineering. Adobe Press, Mountain View, CA, 1996.

[15] H.A. Reijers and K. Voorhoeve. On the Optimal Design of Processes and Information Systems (in Dutch). Informatie, 42:50-57, December, 2000.

[16] E.A. van Veen and J.C. Wortmann. Generative bill of material processing systems. *Production Planning and Control*, 3(3):314-326, 1992.