Product-Based Workflow Design

HAJO A. REIJERS, SELMA LIMAM, AND
WIL M.P. VAN DER AALST

HAJO A. REIJERS is an Assistant Professor of Business Process Management at the Faculty of Technology Management of Eindhoven University of Technology. He holds a Ph.D. in Computing Science from the same university. During the past eight years he has worked for several management consultancy firms, most recently as a manager within the Deloitte & Touche consultancy practice. His main interests are business process redesign, workflow management systems, Petri nets, and simulation.

SELMA LIMAM is an Assistant Professor of Information Systems at the Faculty of Technology Management of Eindhoven University of Technology. She obtained her Ph.D. in industrial engineering in 1999 at the National Polytechnic Institute of Grenoble. Her current research interests are business process redesign, workflow management systems, Petri nets, and simulation.

WIL M.P. VAN DER AALST is a Full Professor of Information Systems and head of the Department of Information and Technology of the Faculty of Technology Management of Eindhoven University of Technology. He is also a part-time full professor in the Computing Science department of the same university and has been a consultant for Deloitte & Touche for several years. His research interests are simulation, Petri nets, process models, workflow management systems, verification techniques, enterprise resource planning systems, computer-supported cooperative work, and interorganizational business processes.

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) as part of current enterprise resource planning (ERP) systems, which is mainly driven by the bill of material (BOM). For information-intensive products such as insurances, and many other services, the workflow process typically evolves or is redesigned without careful consideration of the structure and characteristics of the product. In this paper, we present a method named product-based workflow design (PBWD). PBWD takes the product specification and three design criteria as a starting point, after which formal models and techniques are used to derive a favorable new design of the workflow process. The ExSpect tool is used to support PBWD. Finally, using a real case study, we demonstrate that a full evaluation of the search space for a workflow design may be feasible depending on the chosen design criteria and the specific nature of the product specifications.

KEY WORDS AND PHRASES: business process redesign, formal methods, process design, workflows, workflow systems.

Journal of Management Information Systems / Summer 2003, Vol. 20, No. 1, pp. 229–262.
© 2003 M.E. Sharpe, Inc.
0742–1222 / 2003 $9.50 + 0.00.

ATTENTION FOR THE SUBJECT OF REDESIGNING business processes emerged during the early 1990s (see, e.g., [9, 13]). At the beginning of the twenty-first century, "process thinking" and business process redesign (BPR) have become mainstream thinking for business people and systems people alike [28]. This paper presents a method to design or redesign administrative business processes—also known as *workflow processes* or *workflows* for short—inspired on manufacturing principles. In the manufacturing world, the bill of material (BOM) is used to derive the production process [23]. Consider for example material requirements planning (MRP), often referred to as MRP-I, which determines the production schedule based on the ordered quantities, current stock, and the composition of a 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.

In contrast to manufacturing, the product and the process have often diverged in actual workflows. Workflows found in banks and insurance companies for products like credit, savings and mortgages, damage and life insurance, and so on may well exist for decades. Since their first release, those processes have undergone an *organic evolution.* For example, historical problems in performing certain computations have resulted in the addition of checks. Another example is the effect that a historical case of fraud may have on a process. The case of fraud may result in the addition of an additional check enforcing a rather restrictive type of control. Aside from the evolutionary changes of the process, the state of technology of some decades ago has considerably influenced the structure of these workflows permanently. For example, it used to be laborious to make a copy of a client file. Therefore, in many actual workflows a highly sequential structure of tasks can be distinguished, where at most one person at a time works on such a file. So, the structure of an actual workflow may not be related to the product characteristics any more.

Concerning the design of a process, there are principally two important choices one has to make before a redesign can start: (1) the starting point and (2) the design method. With respect to the *starting point,* one can take the *existing* process as a starting point, take a *clean sheet* approach, that is, the process is designed from scratch, or use a *reference model* as a template for the new process. The latter can be seen as a mix between the first two. There is considerable discussion in the literature on the choice between taking the existing process and a clean-sheet approach (for an overview, see [22]), with clear advocates of the clean-sheet in Hammer and Champy [14]. The *design method* that is chosen is by nature either dominantly participative or analytical. *Participative* approaches aim to develop a process design within the settings of workshops, where small groups of consultants, managers, and specialists work together. An *analytical* approach aims at using formal theory and techniques to model and derive the process design.

Prevailing practice for the design of business processes is a participative approach that takes the existing process as a starting point (e.g., [7]). The method we propose in this paper, on the other hand, is analytical and clean-sheet. It is explicitly presented to counter the problems of prevailing practice. In our opinion, workshops much too

often generate high-level designs of which the expected gains cannot be quantified. Such designs incorporate enough uncertainty to satisfy all workshop members, but they are too vague to implement: "the devil is in the details." Taking the existing process as a starting point also introduces the danger of analyzing the existing processes in too great a depth and becoming constrained by it when trying to think of new ways of working (e.g., [25]). The organic evolution of workflows we identified earlier only intensifies this danger. With this paper we hope to stimulate more rational and quantitative ways of process design. However, the positive effects of participative design methods (e.g., [17, 29]) may very well be used to enhance the application of the method we propose. In fact, de Crom and Reijers [8] describe how prototyping that involves end users may help to validate a new process design that results from our approach.

To our knowledge, the only known analytical design methods are described in van der Aalst [3], Aldowaisan and Gaafer [7], Herrmann and Walter [17], Hofacker and Vetschera [18], and Orman [24]. All authors introduce formal models of the business process design problem. All their approaches, however, suppose a preexisting notion of the process tasks, which should be ordered in the final design. The method described in Aldowaisan and Gaafer [7] even explicitly takes the existing process as a starting point. The method we propose in this paper is truly clean-sheet, as it uses the product specification to derive a minimal number of required tasks to produce the product in question. Principles of optimally ordering the tasks that are applied in Orman [24] and van der Aalst [3] can be integrated in our approach, as illustrated by the case study in this paper. A particular difference with the method as presented in Herrmann and Walter [17] is that we (1) allow for multiple optimization criteria to be included simultaneously in ordering the tasks in a design and (2) the evaluation of the process design may incorporate the structure of the process—not only the particular task occurrences.

The method presented is named product-based workflow design (PBWD). It has been introduced in van der Aalst et al. [5] and was inspired on earlier work [2]. The method distinguishes four phases:

1. *Scope:* In this initial phase, the workflow that will be subject to the redesign (or design) is selected. The redesign objectives for this workflow are identified, as well as the limitations to be taken into consideration for the final design.
2. *Analysis:* A study of the product specification leads to its decomposition into data elements and their logical dependencies. The existing workflow—if any—is studied to retrieve data that is both significant for designing the new workflow and for the sake of evaluation.
3. *Design:* Based on the redesign (or design) objectives, the product specification decomposition and some estimated performance figures, one or several alternative workflow structures are derived. A workflow structure consists of tasks that retrieve or process data elements.
4. *Evaluation:* The alternative workflow structures are verified, validated with end users, and their estimated performance is analyzed in more detail. The most promising designs are presented to the commissioning management to

assess the degree in which objectives can be realized and to select the design to be implemented.

These phases are proposed to be executed in a sequential order, but in practice it is very plausible and sometimes desirable that iterations will take place. For example, the evaluation phase explicitly aims at identifying design errors, which may result in rework on the design.

In order to support the PBWD method, we have used ExSpect (see [6, 10, 11]). This tool allowed us to implement the automated part of the analysis phase using the functional language of ExSpect (see the seventh section) and to design a prototype for the evaluation phase using the simulation capabilities of ExSpect (see the eighth section).

The Scope Phase

Workflow Selection

AN IMPORTANT AIM FOR THE SCOPE PHASE is to select the workflow that is to be designed or redesigned. More specifically, it aims at identifying the *product* of which the corresponding workflow is to be designed. The selection of a product–workflow combination can be made on various grounds [14, 19]; Sharp and McDermott [28] list the criteria that are used in practice. If, for example, there is a new product developed by a company's marketing department, then the motivation for designing the corresponding workflow is clear. If an existing workflow is taken to be redesigned, selection criteria may be the following:

- *Dysfunctionality* of the workflow. Typically, clear indicators of dysfunctional workflows are: extensive information exchange, data redundancy, long throughput times, high ratio of controls and iterations, many procedures for exception handling and special cases, poor service quality and customer satisfaction, and conflicts across departments.
- *Importance* of the workflow. A workflow may contribute more or less to the critical success factors of a company, its profitability, customer satisfaction, market share, and so on.
- *Feasibility* of redesign. A redesign effort is more likely to succeed when the workflow is directly linked to the needs of customers, when the scope of the workflow becomes smaller (but then the payoff drops), when expected redesign costs become less, and when knowledge about the product, design approach, and the existing workflow are available in larger quantities.

In practice, the various criteria for selecting a workflow to be redesigned are different for each company and even for each BPR effort.

Workflow Boundaries

After selecting the proper product–workflow combination it is important to fix the *boundaries* of the workflow to be redesigned. Important for these boundaries are the

logical, location, and customer-centered viewpoints. We will shortly discuss each of these. Note that in actual settings, other criteria may be more relevant.

In practice, what may be seen by different departments as the logical start and end of a workflow may differ. From a salesperson's perspective, the workflow for mortgage applications ends when a signed contract is returned by the customer. However, various operations in the back-office may still be required to fulfill the mortgage offering. A logical start state and end state should be determined for the workflow that is to be redesigned prior to the design itself.

The second viewpoint for the boundaries concerns the location of the workflow. Similar existing workflows may be executed at different locations, such as in different offices or countries. The question should be answered for which locations the redesign will be effectuated. This issue will determine the types of systems that are incorporated, which kind of regulations are in effect, which performance is desirable, and which people are involved.

The last important viewpoint for the boundaries of a workflow concerns the customer. Seemingly similar products may be offered to very different types of customers. A typical distinction within a banking environment is to distinguish between well-to-do and other clients. Depending on the type of customer, different procedures or product characteristics can be relevant.

Redesign Objectives

An important and often neglected activity during the scope phase of a redesign initiative is to explicitly formulate the redesign objectives (see [4]). Aside from the performance targets such as throughput time, operational cost, and required labor hours that should be met by the newly designed workflow, the redesign initiative itself may have to be executed within a certain time, quality, and budget framework. Something that is even less frequently executed is the *null measurement* (see [4]). A null measurement establishes the score of the performance targets just before the redesign is carried out. It establishes, for example, the average speed with which a company handles customer complaints by observation, data analysis, and so on. Such a measurement enables the formulation of sensible target values and makes an ex post evaluation possible by measuring the relevant entities again.

Feasibility

The discussed elements of the scope phase so far are rather general and applicable for all kinds of redesign efforts. To determine the feasibility of the PBWD approach to design a particular workflow, it is of the utmost importance that there is a well-defined notion of the product to be delivered by this workflow. Actual manifestations of such a product notion are handbooks, internal regulations, procedure codifications, legislations, commercial brochures, and so on. It is inevitable for a proper assessment of the feasibility of a redesign of PDWD that during the scope phase a collection takes place of the materials that define the product specification.

Even if there is no physical manifestation of a product specification, it may very well be that the concept of a product does exist with marketers, product managers, or general management. It is important to check the maturity and concreteness of these notions. If they are sufficiently mature, it is required before the next phase of analysis starts that an explicit product specification is defined.

In summary, the deliverables of the scope phase are:

- a precisely demarcated workflow process to be designed,
- the performance targets for the new design, and
- a product specification.

The Analysis Phase: Product/Data Structures for Workflow Processes

IN THE ANALYSIS PHASE, all distinguished material that classifies as product specification is analyzed to identify the data elements, their dependencies, and the logic involved. We use a structure for administrative products that is somewhat similar to the BOM we discussed in the introduction. 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, that is, 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 of products needed. Figure 1 shows the simplified BOM of a car that is composed of an engine and a subassembly. The subassembly consists of four wheels and one chassis.

For information-intensive processes, the traditional BOM is not very useful. 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 to express that exactly the same information is required multiple times. 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 both 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:* a set of data elements, with a special top element:

$$top \in D,$$

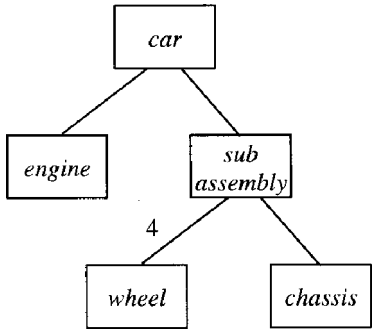

Figure 1. The BOM of a Car

- *C:* set of constraints; a constraint can be any Boolean function; the function that always yields *true*—denoted *true*—is part of *C*:

$$true \in C,$$

- the function *pre* gives for each information element the various ways of determining a value for it on basis of the values of different sets of other information elements:

$$pre : D \rightarrow \mathcal{P}(\mathcal{P}(D)), \text{ such that}$$

  - there are no "dangling" data elements and a value of a data element does not depend on itself:

$$R = \left\{ (p,c) \in D \times D \mid c \in \bigcup_{es \in pre(p)} es \right\} \text{is connected and acyclic,}$$

  - the top element cannot be used for determining the value of any other data element:

$$\forall (p,c) \in R : c \neq top,$$

  - if there is a data element that does not require any other data element, we denote for ease of analysis the set of required data elements as the empty set:

$$\forall e \in D : \varnothing \in pre(e) \Rightarrow pre(e) = \{\varnothing\},$$

- *F:* a set of production rules, based on the definition of *pre*; *F* consists of all ordered pairs of data elements between which a dependency may exist:

$$F = \{(p, cs) \in D \times \mathcal{P}(D) \mid cs \in pre(p)\},$$

- the function *constr* that associates a constraint to each production rule:

$$constr : F \rightarrow C, \text{ such that}$$

- there are no constraining conditions on the production of data element values that does not require the values of other data elements:

$$\forall e \in D : pre(e) = \{\varnothing\} \Rightarrow constr(e, \{\varnothing\}) = true,$$

- a function *cst,* which gives the cost of using a production rule:

$$cst : F \rightarrow \mathbb{N}$$

- a function *flow,* which gives the time it takes to use a production rule:

$$flow : F \rightarrow \mathbb{N}$$

- a function *prob,* which gives the probability that a production rule will yield an acceptable result when used:

$$prob : F \rightarrow (0..1], \text{such that}$$

- if there are no constraints on using the production rule or no other data elements are required, then it will *always* lead to an acceptable result:

$$\forall e \in D : pre(e) = \{\varnothing\} \Rightarrow prob(e, \{\varnothing\}) = 1$$
$$\forall (p, cs) \in F : constr(p, cs) = true \Rightarrow prob(p, cs) = 1.$$

The product/data model expresses relations between data elements. These relations can be used to produce a value for the data element *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 $d, e, f \in D$, 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 rule can *only* be applied if $constr(d, \{e, f\})$ evaluates to *true*. A constraint specifies the circumstances when the production rule is applicable. Such a constraint typically depends on the values of information elements that are used to produce another value. The expected probability that a production rule yields an acceptable result for d when it is indeed applicable is given by $prob(d, \{e, f\})$. In other words, there is a probability of $1 - prob(d, \{e, f\})$ that the production rule either does not lead to a result at all or that the result is unacceptable for further processing. 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 $pre(d) = \{\varnothing\}$ 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 one. 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 one 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. This is infused rather by on-the-job experience with PBWD than by the

belief that this is generally a realistic assumption. In actual applications of PBWD (see, e.g., [8, 26, 27]), we found that probabilities are hard to determine—let alone that their relations are well understood. In this sense, the product/data model is really a best effort to grasp the stochastic nature of business process execution, which is mostly ignored in more intuitive redesign approaches. However, it is possible to adapt the product/model to incorporate a dependency notion. This will make the design and analysis of workflows (see the sixth and eighth sections) more complex, but it leaves the idea behind PBWD intact. Note also that because probability values can be less than one, 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 $(1 - 0.9) \times (1 - 0.9) = 0.01$ probability that no value for the top element can be determined.

The applicability of the PBWD method in practice relies heavily on the ability to collect the type of data that is part of the product/data model. In our experience with applying PBWD within the financial services and governmental agencies, the needed effort to collect this type of date differs from company to company. Some companies maintain a very detailed administration of time durations and flow fractions. For example, a Dutch health insurance company we know uses this data for capacity planning purposes and has therefore implemented procedures to manually record the data. Other companies obtain this information almost for free, because it is stored automatically by their workflow management system that supports existing processes. On the other hand, this type of data may be totally absent too. Then, constructing a product/data model requires an intensive preceding trajectory of collecting such data. Observations, interviews, and expert estimations are instruments that may be of use within this context. Note that in the prevailing design practice this type of information is for the most part ignored. This clearly eliminates the possibilities of any quantitative support for the design.

The Helicopter Pilot Example

AN EXAMPLE OF A PRODUCT/DATA MODEL is depicted in Figure 2. All boxes 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 Air Force. Multiple starts of an arrow are joined into one by small black dots. For example, the outgoing arrows of data elements *b* and *c* are joined into one single arrow leading to data element *a*. It represents a production rule for *a* with data elements *b* and *c* as inputs. 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,

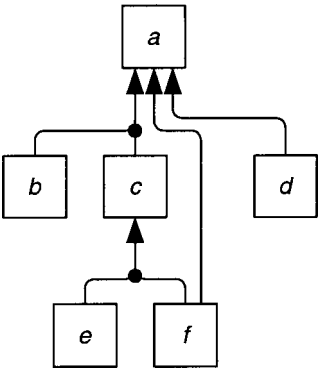

Figure 2. Helicopter Pilot Product/Data Model

Table 1. Relations *constr*, *cst*, *flow*, and *prob* for Testing a Helicopter Pilot Candidate

x	*constr*(x)	*cst*(x)	*flow*(x)	*prob*(x)
(a, {b, c})	True	80	1	1.0
(a, {d})	$*d \in$ {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

- *c:* physical fitness,
- *d:* latest result of suitability test in the previous two years,
- *e:* quality of reflexes, and
- *f:* quality of eyesight.

One of the things that follow 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 either of the following:

- the results of the psychological test (b) and the physical test (c);
- the result of a previous test (d); or
- the candidate's eyesight quality (f).

The relations *constr, cst, flow,* and *prob* for this example are as shown in Table 1. If x is a data element, the value of x is denoted with $*x$. 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 *knockout* for the test result [3]. 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 knockout for a specific case.

From the helicopter pilot product data model, it becomes clear how the dependencies between data may be used to derive a favorable design. For example, suppose that our redesign target is to minimize the cost and we have to decide what to do first: checking the eyes of the candidate or looking for a previous test result. From an analysis of the product/data model, it follows that first looking up the test result and, if still required, subsequently testing the candidate's eyes is a cheaper approach than the other way around. The expected cost of pursuing these scenario's is, respectively, 66.8 (= 10 + (0.1 * 10) + (0.9 * (60 + 0.4 * 5)) and 68.6 (= 60 + (0.4 * 5) + 0.6 * (10 + 0.1 * 10)). Note how hard it would be to make such a decision on basis of intuition alone.

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 a working workflow design. However, to consider the optimality of a workflow model the exact specifications are mostly not relevant. Much more relevant is that a given workflow model conforms with the product/data model under consideration. We define a correctness notion for workflow models.

Workflow Model

A workflow 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,
- $prod: T \rightarrow F$, the production rule applied in the task, and
- Ω is a set of execution sequences.

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

Conformance

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

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

$$\exists 1 \leq i \leq k, cs \in \mathcal{P}(D) : prod(t_i) = (top, cs).$$

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

In general, given a product/data model there is an infinite number of conformant top-level workflow models. Different subsets of tasks executed in some order may lead to the production of the top element. In the next section, we will discuss the criteria that make one design preferable over another.

Design Criteria

WHEN DESIGNING PRODUCT-BASED WORKFLOWS, a large variety of criteria may be taken into account (for an overview, see [28]). Clearly, it is impossible to translate all these criteria into one design approach. Therefore, we are considering for our paper the following three important design criteria: (1) *quality,* (2) *costs,* and (3) *time.* We focus on these three criteria because we have encountered them in several redesign projects. Cost is often a convenient view on the internal efficiency of a company, whereas time is a way to make the external demands of clients on a company operational. Very often, a trade-off has to be made between doing things at low cost (e.g., by postponing activities that may become superfluous) or in a rapid fashion (e.g., by doing possibly superfluous activities simultaneously). But even when cost is the prevalent factor and time is not, there may be requirements on an organization that it cannot drop its service between a certain quality threshold. This may be because of government regulations or expected dents in the corporate image. Different redesign criteria or other definitions of the quality, costs, and time criteria than the ones that follow may be more suitable in an actual redesign encounter.

Costs and time will be defined in this paper according to the functions *cst* and *flow.* Quality is defined as the probability that an acceptable 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 subgraph 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 $(d,\{\varnothing\})$ 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 $d \in S$:

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

where

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

The quality of a plan is the probability that an acceptable value of the top element can be determined assuming that all production rules *only* referring to elements in S are executed. The exact formulation of (1) ensures that all possible scenario's of obtaining an acceptable result are added in the overall quality of the plan. Note that for any production rule $(p, cs) \in F$ 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 (1), $q_a = 1 - (1 - prob(a,\{d\}) \cdot q_d \cdot \delta(d))$ with $q_d = 1 - (1 - prob(d, \{\varnothing\}) \cdot q_\varnothing \cdot d(\varnothing)) = 1$. So, $p\_quality(S_1) = q_a = 0.1$. Similarly, for plan S_2, $p\_quality(S_2) = 1$ and for plan S_3, $p\_quality(S_3) = 0$.

Note that the introduced notion is a rather narrow view on quality. For example, it disregards the distinction between producing an exactly correct result and an acceptable result. In line with the definition of the *prob* function of the product/data model, our quality notion focuses on producing acceptable results. However, from a quality viewpoint it may be of interest to define a quality design criterion that expresses which accuracy of the results may be expected. As we have indicated in the introduction of this section, the presented criteria are open for adaptation in actual PBWD encounters.

Costs of a Plan

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

$$p\_csts(S) = \sum_{(p,cs)\in F} cst(p,cs) \cdot \delta(p) \cdot \prod_{e\in cs} \delta(e). \quad (2)$$

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 $\{p\} \cup cs$ 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 $(d, \{\varnothing\})$. So, according to formula (2), $p\_csts(S_1) = cst(a, \{d\}) \cdot d(a) \cdot d(d) + cst(d, \{\varnothing\}) \cdot d(d) \cdot d(\varnothing) = 20$ (see Table 1). Similarly, $p\_csts(S_2) = 405$ and $p\_csts(S_3) = 60$.

In practice, the *expected* cost of a plan may be of much more interest to the company that issues the design than the maximum cost. However, the attractiveness of the presented cost criterion of a plan is the simplicity of its definition and its independence of the way the production rules are structured in the final workflow design. Therefore, this criterion should be seen as a first cost indication for a selected set of production rules. It may be wise to refine this notion in later steps of the design or to define a different cost notion from the start (e.g., focusing on the minimal cost). We will indicate further in this section at what stage of the design a more refined cost notion in our opinion is recommendable.

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:

$$p\_flow(S) = \sum_{(p,cs)\in F} flow(p,cs)\cdot \delta(p)\cdot \prod_{e\in cs} \delta(e). \tag{3}$$

Formula (3) expresses the sum of the flow times of individual production rules that can actually be applied because their input data elements are also part of the plan. Obviously, by executing some of the production rules of the plan in parallel the *actual* 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: $(d,\{\emptyset\})$, $(a,\{d\})$, $(f,\{\emptyset\})$, $(a,\{f\})$, $(e,\{\emptyset\})$, $(c,\{e,f\})$, $(b,\{\emptyset\})$, $(a,\{b,c\})$. Then the average worst case $flow\_time(S_4) = flow(a, \{b, c\}) \cdot \delta(a) \cdot \delta(b) \cdot \delta(c) + flow(a, \{f\}) \cdot \delta(a) \cdot \delta(f) + flow(a, \{d\}) \cdot \delta(a) \cdot \delta(d) + flow(b, \{\emptyset\}) \cdot \delta(b) \cdot \delta(\emptyset) + flow(c, \{e,f\}) \cdot \delta(c) \cdot \delta(e) \cdot \delta(f) + flow(f, \{\emptyset\}) \cdot \delta(f) \cdot \delta(\emptyset) + flow(e, \{\emptyset\}) \cdot \delta(e) \cdot \delta(\emptyset) + flow(d, \{\emptyset\}) \cdot \delta(d) \cdot \delta(\emptyset) = 76$ time units. Now suppose that the production rule $(a, \{d\})$ leads to a value for a, then the $flow\_time(S_4) = flow(a, \{d\}) \cdot \delta(a) \cdot \delta(d) + flow(d, \{\emptyset\}) \cdot \delta(d) \cdot \delta(\emptyset) = 17$ 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.

Given a plan S, it is easy to calculate its quality $p\_quality(S)$, the associated maximum costs $p\_costs(S)$, and the worst-case flow time $p\_flow(S)$. Note that a plan is not a workflow model: it is merely a subset of data elements. In general, given target values for each of these design criteria it is very complex to find the optimal solution. In this paper, we present an approach to deal with this multicriteria optimization problem in a heuristic sense. It should be stressed, however, that this approach does not necessarily render the optimal solution. Given sufficient time and a limited product/data model, a brute-force generation of all alternative workflow designs may be preferable. In our proposed heuristic, the notions of a plan and the criteria $p\_quality(S)$ and $p\_costs(S)$ play a central role. The heuristic uses the fact that $p\_quality(S)$ and $p\_costs(S)$ allow for the definition of a *cost optimal plan* (COP) 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

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

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 COP.

A COP gives the least costly subset of data elements that needs to be calculated to obtain a given quality level. Looking at the underlying multicriteria optimization problem of finding an optimal workflow design on basis of a product/data model, we have in truth used the notion of the COP to recast the quality criterion as a constraint. In our experience, several companies will find this acceptable as they will focus primarily on time and cost—although they may feel for legal reasons or for reasons of decency obliged to maintain a minimal level of quality. Note that the usefulness of a COP increases when the number of data elements composing the product/data model becomes high. Then, the COP helps to make a first rough elimination of data elements before the design process starts. This heuristic can be implemented in a toolbox (see below).

Note that the costs associated to such a plan are the maximal costs, that is, the costs that are made if all corresponding production rules need to be calculated. However, a knockout 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 then be based on refined notions of the time criterion or the cost criterion mentioned in the previous section, that is, focusing on the expected flow time and cost rather than their maximums. Such criteria should obviously take the exact ordering of the production rules into account. For the ordering, there are two extreme approaches:

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 that has the best quality/cost ratio, that is, 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 expected 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 maxi-

mize parallelism. For the depth-first approach these activities need to be ordered sequentially.

We wish to conclude this section with two observations. The first is that the presented criteria and their exact definitions are *not* fundamental for the PBWD method. What is fundamental is that these criteria are specified at all and actually used to guide the design. Specific demands of a redesign project may deliver specific design criteria. Second, the approach we presented is not guaranteed to deliver the optimal result with respect to all design criteria we introduced. Rather, we sketched a possible yet general approach to heuristically deal with the complex multi-criteria optimization problem at hand. In the eighth section, we will show a contrasting approach, which explores the complete search space for an optimal design in a practical setting.

An ExSpect Toolbox for Cost Optimal Plan, ExSpect/COP

TO SUPPORT THE NOTIONS OF A *plan* and a *COP,* we have developed a toolbox using ExSpect (see [10, 11]). Before we present the functionality of the toolbox, we briefly discuss the background of ExSpect. ExSpect has been developed in the Department of Mathematics and Computing Science of Eindhoven University of Technology. In 1995 the development of ExSpect moved to the consultancy firm Deloitte & Touche where it is used as the standard modeling and analysis tool. ExSpect is based on high-level Petri nets (see, e.g., [15]). It offers a complete functional programming language, and supports complex simulation studies. The ExSpect language and tool can be used in two ways. One way is to combine predefined building blocks (processors and subsystems) to specify a large system. However, not all systems can be built in this way. It may be necessary to adapt building blocks or even to create them from scratch using the ExSpect functional language. More details about ExSpect can be found in van der Aalst et al. [6].

In order to develop the ExSpect/COP toolbox, we have used the functional part of ExSpect as a high-level programming language. This choice has been made because of the advantages of this functional language over ordinary programming languages.

ExSpect/COP consists of an ExSpect model that returns the COP for a given product/data model and a given quality level. It is composed of a single system where channels, stores, and processors have been created and installed. A channel is a place in the Petri nets terminology. A store is a special kind of place where information can be saved. A processor is a transition in the Petri nets terminology. It is the active component of the specification where functions may be called and conditions defined. For this tool several functions have been defined in order to derive plans, quality, cost, and flow time of plans. Functions also allow the search of a COP.

Figure 3 shows a screenshot of ExSpect/COP. The product/data model is modeled defining two tables: *prt* and *det.* Table *det* (Data ElemenT) returns for each data element the production rule that generates it (*p, cs*), the cost, the probability, and the flow time of generating it from (*p, cs*). Table *prt* (Production Rule Table) lists for each production rule (*p, cs*) the *pre* data elements needed (that is, all elements of *cs*). Both tables are stored in *Store_prt* and *Store_ det.*

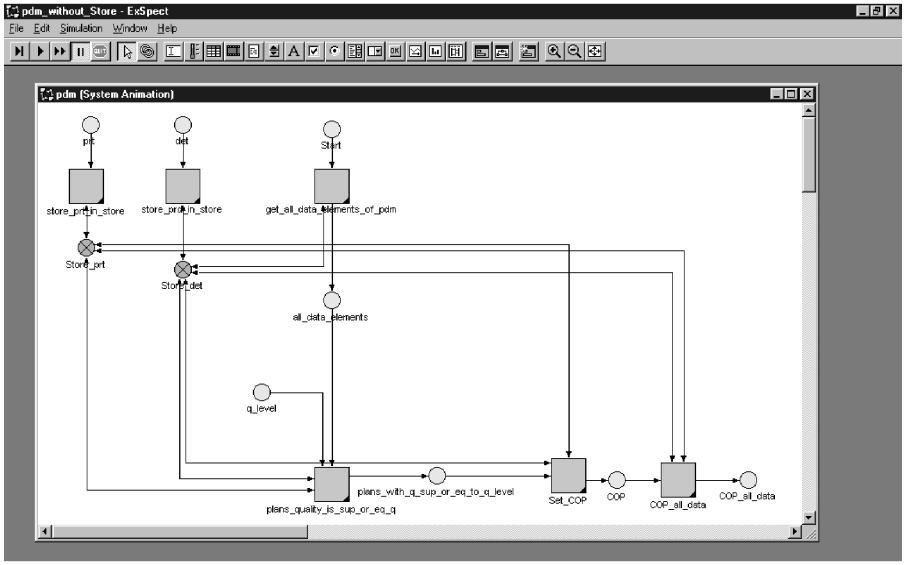

Figure 3. A Screenshot of ExSpect/COP

The first important step in executing ExSpect/COP is to obtain all relevant possible plans that can be generated from the product/data model. A natural way of doing that would be to compute all possible plans that can be derived from the product data model (which is a set of data elements) and then to pick up the plans that meet the user's required quality level. In general, this is not feasible since the number of subsets of possible plans increases exponentially with the set of data elements. So an algorithm has been implemented to reduce the search space for fitting plans. Since we are interested in high-quality level plans, the algorithm is based on the property that if a sub-plan has a lower quality than what is desirable then any subset of this sub-plan will only have a lower or equal quality. Thus, the search for sub-plans should start with the biggest sub-plan, that is, the plan containing all data elements. This reduces considerably the search for sub-plans. To implement this algorithm, we first obtain all the data elements of the product data model (pdm) by firing processor *get_all_data_elements_of_pdm*. In this processor, a function uses tables *prt* and *det* to derive the data elements. Since each data element can be generated from various production rules, it may appear several times as an element of table *det*. Next, plans that meet the desirable quality level only are derived from the set of all data elements by firing processor *plans_quality_is_sup_or_eq_q*.

The second step in executing ExSpect/COP is to identify the COP among all the plans that meet the quality level criteria. This is determined by applying in processor *Set_COP* the COP heuristic described in the sixth section.

Results related to the COP (included data elements, cost of the plan, quality of the plan, and flow time of the plan) are obtained after firing of processor *COP_all_data*.

Figure 4 shows the input data dashboard available in the simulation mode. The dashboard window is used to monitor and edit the contents of the channels during the

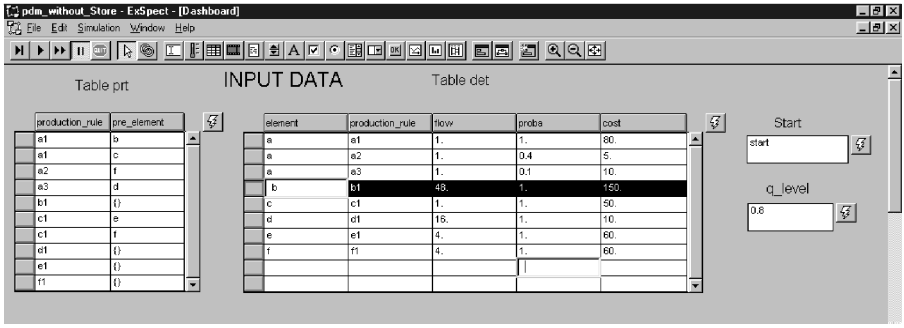

Figure 4. Input Data Dashboard of the ExSpect/COP Toolbox

execution of the ExSpect model. The user is asked to build both tables *det* and *prt,* to set the quality level and to start the simulation. Figure 4 shows the dashboard filled with input data for the helicopter pilot example.

Figure 5 shows the dashboard containing the simulation results for the helicopter pilot example when the quality level equals 0.8. The dashboard gives the results of the execution consisting in the contents of each channel: channel *plans_quality_is_sup_or_eq_q* shows the plans that satisfy the quality level criterion and channel *COP_all_data* shows the COP with details of cost, quality, and flow time.

In the case of the helicopter pilot, there are two plans with a quality level of one. The plan containing all the data elements {*a,b,c,d,e,f*} and the plan {*a,b,c,e,f*}. The COP in this case is the latter plan {*a,b,c,e,f*}, with a flow of 59 and a cost of 450.

If we assume a lower quality level, say 0.8, then we still obtain the two above-mentioned plans. We need to apply a quality level of 0.4 to obtain a different configuration containing 16 plans. The COPs are in this case {*a,c,f*} and {*a,f*}, with a quality of 0.4, a cost of 65, and a flow of five. In this case, the cost is obviously lower than 450 (obtained with a quality level of 0.8), but, on the other hand, there is only a 40 percent chance to make a decision about the suitability for candidates to become a helicopter pilot (data element *a*).

A Case Study: The Social Insurance Company

AT THIS POINT WE HAVE DESCRIBED the first two phases of PBWD, namely the scope and analysis phases. The design and the evaluation phases are not considered in depth here. We do present a case study that illustrates how the complete method works. We aim at demonstrating, using this case, that the design phase is dependent on the strategy the company would like to adopt in its relationship with customers and with respect to other strategic issues such as cost and time. We argue that the relative importance of the criteria the company wishes to set also determines the product-driven redesign rules that should be applied in the design phase. The latter two considerations are, for example, not taken into account in the design method developed in Herrmann and Walter [17].

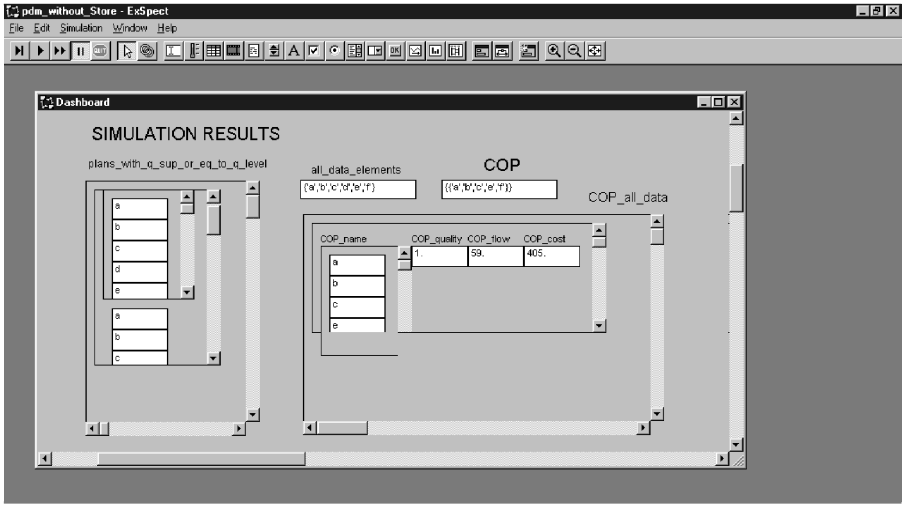

Figure 5. Simulation Results Dashboard for the Helicopter Pilot Example

Introduction and Scope Phase

The PBWD method was applied in 1999 within the setting of a Dutch social insurance company (SIC). The information product under consideration during this project was the decision if a claimant is entitled to pay-related unemployment benefits when he or she claims to have become unemployed. The SIC had unsuccessfully tried to redesign the process in a more conventional approach during that year using brainstorming techniques and green field approaches. The resulting process design could not convince the people on the work floor or the information technology (IT) department that it was a workable improvement of the existing process. At the time, the pressure on the SIC was immense as the Dutch market for this type of product would be liberated at the beginning of 2000, allowing commercial insurance companies to offer the same type of service to the Dutch government. As the end of 1999 was nearing, the management of the SIC became receptive for a new redesign approach, in this case PBWD.

The reference office for this project was one medium-sized office out of the 20 offices in the Netherlands working on this decision. This office on average handles a little less than 10,000 claims a year. Three design directives were defined by the SIC for the final design. First and mainly to *decrease* the expected average effort in terms of *human labor hours*. The reason is: an expected gain in efficiency and a minimization of cost. Second, to *apply the case worker principle:* for each particular case, the same resource executes all steps within the process. The motivation is: higher quality of work, as the case worker knows the case in detail. Finally, to minimize the number of contacts with customers. The motivation is: an improvement in the service quality as perceived by the applicant.

In a period of three months, a new workflow design was derived for the described product, taking the former directives into account. Information related to the product

in question, the claim decision, was derived from three sources: the Dutch Unemployment Law, which contains the regulations regarding the unemployment benefits decision; the SIC handbooks, which maintain operational interpretations of the law; and a detailed SIC administration of causes for denying unemployment benefits to individual cases, as well as other statistical figures on its operations.

Analysis Phase

In the analysis phase a study of the product specification was made. It led to its decomposition into data elements and their logical dependencies (production rules) as defined in the third section. The analysis phase yielded a number of data elements, which were numbered as i1, i2, ..., i51.

Data Elements

The informal description of the meaning of each of the data elements is given in Table 2. As can be seen from this table, i18 is the top element "claimant is entitled to (pay-related) unemployment benefits." Data elements i1 to i10 are "time periods," for example, i1 is the period in which a claimant receives illness benefits (if at all). It is prohibited under Dutch law to receive unemployment benefits during such a period.

The Production Rules

Table 3 lists the production rules needed for this case. The exact specifications of the content of the production rules are not presented here because of its sheer size (30 pages) [27]. To give the reader some intuition on these rules, it can be stated here that the decision to grant benefits to someone who has become unemployed (i18) depends on three major requirements, which are:

1. The claimant should be insured against becoming unemployed (i9).
2. The claimant must satisfy a "refer requirement" (i11).
3. The claimant must satisfy a "labor history requirement" (i15).

The mentioned requirements themselves depend on various other data elements. For example, the labor history requirement (i15) can be satisfied by either having become unemployed immediately after a period of protracted disability (i17) or by having worked during four out of five of the years immediately preceding the unemployment (i16). The exact logic and constraints to count these years are expressed by other production rules and data elements of the product/data model. Figure 6 shows the entire product data model.

To help implementing the first design directive, for each production rule it was decided whether it could be formally specified, thus enabling an automation of the production rule. Of all rules, 26 production rules were formally specifiable and only six production rules could not be specified or could not be specified completely in the form of an algorithm. The latter involved the production of data elements i11, i16,

Table 2. Meaning of Data Elements

Data element	Description
i1	Period in which claimant receives illness benefits
i2	Period in which claimant receives excessive combined social benefits
i3	Period claimant lives/resides outside the Netherlands
i4	Period in which claimant does not rightfully live in the Netherlands
i5	Period in which claimant is detained/imprisoned
i6	Period in which the claimant is 65 years or older
i7	Period in which the claimant has legal scruples against insurance
i8	Period in which claimant enjoys holiday
i9	Period in which claimant is an employee
i10	Period in which claimant is unemployed
i11	Claimant satisfies refer requirement
i13	Date from which claimant lost the right for payment
i14	Data from which the claimant is available to accept labor
i15	Claimant satisfies labor history requirement
i16	Claimant satisfies four out of five years requirement
i17	Claim is directly following labor disablement benefits
i18	Claimant is entitled to (pay-related) unemployment benefits
i21	Birth date of the claimant
i23	Claimant's holiday administration
i24	Registration of unemployment insurance
i25	Registration of social benefits
i27	Claimant's unemployment is caused by strike/work stoppage
i28	Period in which applicant receives reintegration benefits
i29	Refer period for claimant
i30	First day of unemployment of claimant
i31	Number of weeks claimant worked in refer period
i32	First week of unemployment of claimant
i33	Registration of housing
i34	Average number of labor hours per week of claimant
i35	First day of labor history for applicant
i36	Day status survey of claimant's labor history
i37	Loss pattern of labor hours of claimant
i38	Care data on claimant
i39	Employment function of which the claimant has become unemployed
i40	Employment functions that have been followed up by the employment function of which the claimant has become unemployed
i41	Earlier employment functions of the claimant
i42	Approved labor courses for unemployed
i43	Common first labor day for claimant
i44	List of claimant's yearly worked days
i45	Register of convictions
i47	Education/courses of claimant that precede or follow on the loss of labor hours
i48	Weeks in refer period already taken into account
i49	Labor pattern of claimant
i50	Register of special classes of employment functions
i51	Claimant has taken care of underage children

Table 3. SIC Production Rules

Production rule	Automatic?	Constraint	Cost	Probability
(i11, {i31})	Partly	i31 defined	0.6	0.85
(i13, {∅})	No	True	0.08	1.0
(i14, {∅})	No	True	0.08	1.0
(i15, {i16})	Yes	i16 = true	0	0.997
(i15, {i17})	Yes	i17 = true	0	0.003
(i17, {i25, i30})	Partly	True	6.1	1.0
(i16, {i25, i30, i35, i36, i44})	Partly	True	5.61	1.0
(i18, {i1})	Yes	i37 in i1	0	0.009
(i18, {i2})	Yes	i37 in i2	0	0.013
(i18, {i8})	Yes	i37 in i8	0	0.016
(i18, {i9})	Yes	i9 = false	0	0.002
(i18, {i10})	Yes	i10 not defined	0	0.068
(i18, {i11})	Yes	i11 = false	0	0.079
(i18, {i15})	Yes	i15 = false	0	0.21
(i23, {∅})	No	True	0.67	1.0
(i27, {∅})	No	True	0.08	1.0
(i34, {i36, i37, i41})	Partly	True	4.2	1.0
(i36, {∅})	No	True	1.0	1.0
(i37, {∅})	No	True	1.67	1.0
(i39, {∅})	No	True	0.17	1.0
(i40, {i39, i41}), (i42, {i47})	Partly	True	0.3	1.0
(i43, {i39, i49})	Partly	True	0.6	1.0
(i47, {∅})	No	True	0.33	1.0

Notes: The following production rules are automatic, their constraints evaluate to true, there is no cost in executing them and their success probability equals one: (i1, {i25, i37}), (i2, {i25, i37}), (i3, {i33, i37}), (i4, {i33, i37}), (i5, {i37, i45}), (i6, {i21, i37}), (i7, {i24, i37}), (i8, {i23, i37}), (i9, {i24, i39}), (i10, {i13, i14, i34, i37, i42}), (i15, {i16, i17}), (i17, {i25, i30}), (i18, {i9, i11, i15}), (i21, {∅}), (i24, {∅}), (i25, {∅}), (i28, {i25, i37}), (i29, {i25, i30, i35, i36}), (i30, {i32, i37, i43}), (i32, {i1, i2, i3, i4, i5, i6, i7, i8, i10, i27, i28}), (i33, {∅}), (i35, {∅}), (i41, {∅}), (i44, {∅}), (i45, {∅}), (i48, {∅}), (i49, {∅}), (i31, {i29, i40, i48}).

i34, i40, i42, and i43. To make the difference clear between (partly) manual and automatic production rules, (partly) manual production rules are represented in the product/data model like dashed lines (see Figure 6).

In Table 3, four columns are depicted in addition to that of the production rule: "automatic?" "constraint," "cost," and "probability." The column "automatic?" is defined in order to take into account the first design directive. Columns "constraint," "cost," and "probability" are directly derived from the formal product model, as defined in the third section. We will discuss each of them now.

- The column "automatic?": for a production rule for a *node* element it indicates whether it can be formally specified. For a production rule for a *leaf* element it indicates whether it is directly available and accessible to the SIC. For both

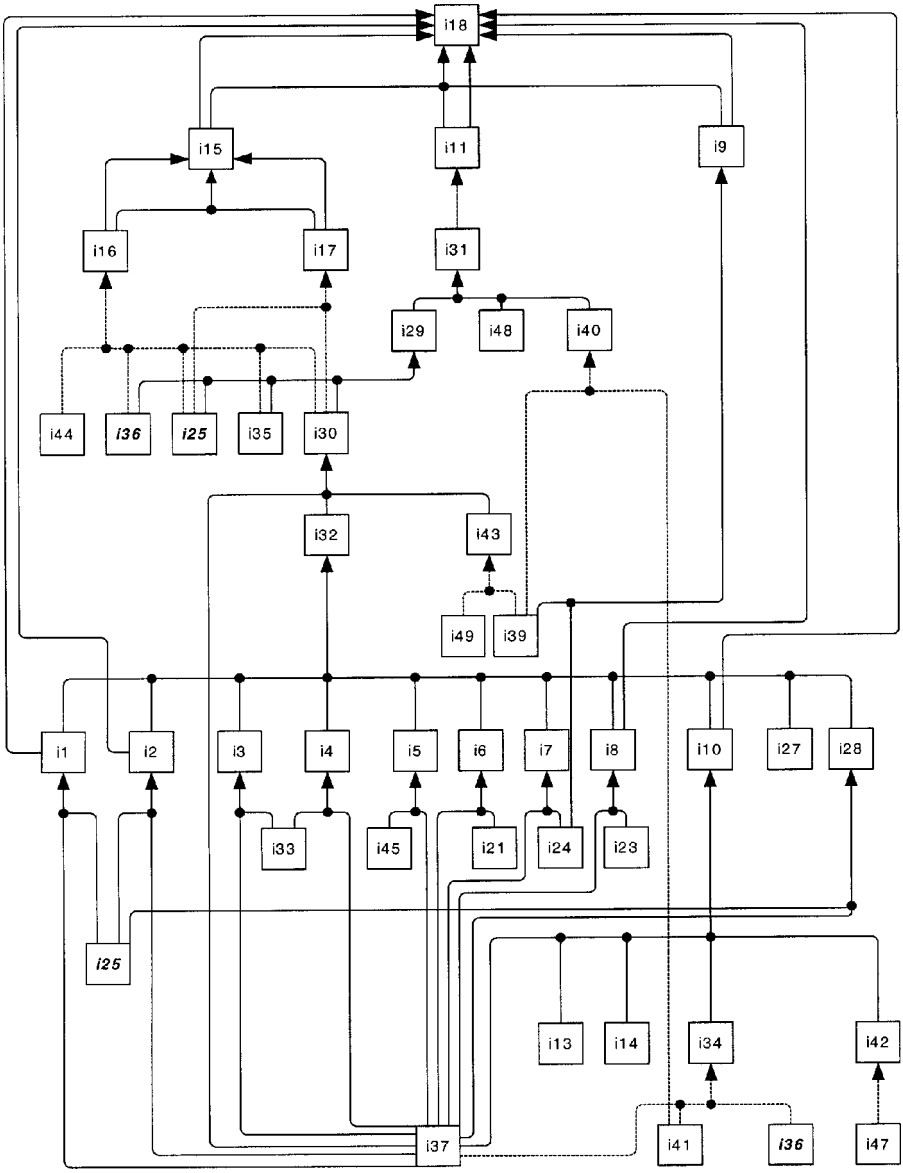

Figure 6. The SIC Product/Data Model

types of rules, a positive answer to this question implies that it can be automatically made available to the process. Hence, no labor cost is involved. For information rules for node elements, it may be the case that it is not completely formally specifiable. Partial handwork is still required. It is indicated by the value "partly" in the "automatic?" column.

- The column "constraint": the constraints were derived from the law and handbooks and also from the SIC experts opinions and precedents.

- The column "probability": due to the aggregated administration of historical cases the SIC maintained, a quantitative survey could be easily executed after the probabilities under which the production results produce the desired result. Although these probabilities are probably not completely independent of each other, there were no records of the dependencies among them.
- The column "cost": cost is expressed in terms of the average time in minutes an SIC clerk has to spend on it. For the execution of production rules that could be specified formally no cost figures were assumed to be imposed.

Concerning production rules that obtain values for the 18 leaf nodes, it turned out that 10 of the leaf data elements are directly and permanently available to the SIC itself (i21, i24, i25, i33, i35, i41, i44, i45, i48, and i49). This was either because the SIC maintains the data itself or because it had direct access to data of third parties. No production rules are required for these data elements and no cost for obtaining them has been applied. For each of the other eight leaf data elements (i13, i14, i23, i27, i36, i37, i39, and i47), exactly one production rule was specified, with a cost figure used throughout the project. The source of the information is in all cases the claimant itself. For the seven partly manual production rules (i11, i16, i17, i34, i40, i42, and i43), the total cost was determined on basis of the weighted cost for manual and automated cost. For example, the production rule for i11 must be executed manually in 15 percent of the cases; in 85 percent of the cases there is no labor cost. As the average cost for manual execution of this production rule is four labor minutes, the weighted average is 0.15 * 4 = 0.6 minutes.

The Product/Data Model

The relations between the data elements (production rules) are given in the product/data model of Figure 6. Note that a crossing of arrows that is not covered by a black dot has no semantics. Most data elements are represented precisely once in the figure. There are two exceptions: i25 and i36. These are both depicted *twice* to prevent too much entanglement of arrows. This is indicated by the bold and italic form of their identifiers.

As explained in the third section, knockout production rules are rules that execution may lead to the determination of a value for the top element after which the processing can end. So, from Figure 6 follows that eight knockout production rules are in effect. We will informally discuss these knockouts here to illustrate the model.

A straightforward way of decision-making is to check whether the claimant is insured against becoming unemployed (i9), whether the claimant satisfies a "refer requirement" (i11), *and* whether the claimant satisfies a so-called "labor history requirement" (i15). If all these conditions apply, the claimant will receive his or her periodical unemployment benefits. Either unsatisfactory outcome of one of these three conditions will disqualify the claim. The latter three production rules are represented by the single arrows leading from respectively i9, i11, and i15 to i18. There are four more conditions that may also stop the processing if their outcome is unsatisfactory. These three conditions directly depend on the values of respectively i1, i2, i8, and

Table 4. Cost Optimal Plans for the SIC Case

Minimal quality level required	Plan	Quality	Cost
1	∅	—	—
0.9	PLAN 1 = ALL DATA ELEMENTS	0.901	21.78
0.8	PLAN 1	0.901	21.78
	PLAN 2 = PLAN 1 \ {i17}	0.899	15.69
0.3	PLAN 1	0.901	21.78
	PLAN 2	0.899	15.69
	PLAN 3 = PLAN 1 \ {i9}	0.339	21.78
	PLAN 4 = PLAN 1 \ {i17, i9}	0.338	15.69

i10. For example, if the claimant is unemployed while he or she is on a holiday, the claim will not be rewarded. This can be deduced from the value of data element i8.

In the sequel we refer with "manual production rules" to production rules with a positive labor cost, and with "automatic production rules" to all other production rules.

Cost Optimal Plan

Before the design phase starts, the COP was derived using the heuristic in the sixth section. The aim of deriving such a plan is to minimize the costs for treating claim applications and also to facilitate the exploration and design of the workflow. The importance of the COP becomes higher as the number of data elements increases.

We have used the ExSpect/COP toolbox to determine the COP in the case of SIC. Several alternatives have been considered for quality levels. The results are shown in Table 4.

It was agreed to consider as acceptable a 0.8 quality level. As there are two plans with the same cost ad quality, the smallest was chosen to simplify the design. That led to plan 2 as a final COP. The subgraph of the original product/data model on basis of this plan is shown in Figure 7.

Design Phase

The design of the workflow on basis of the COP was driven by the earlier-stated design objectives. We discuss the results for our design approach:

- Minimize the cost: implies automating automatic production rules and using a depth-first strategy in which knockouts are ordered such that the expected cost of executing the process is minimal (see [3, 24]).
- Case worker principle: all work on one specific case is assigned to the same employee. (This is not reflected in the structure of the workflow design, but by the allocation rules used in practice.)

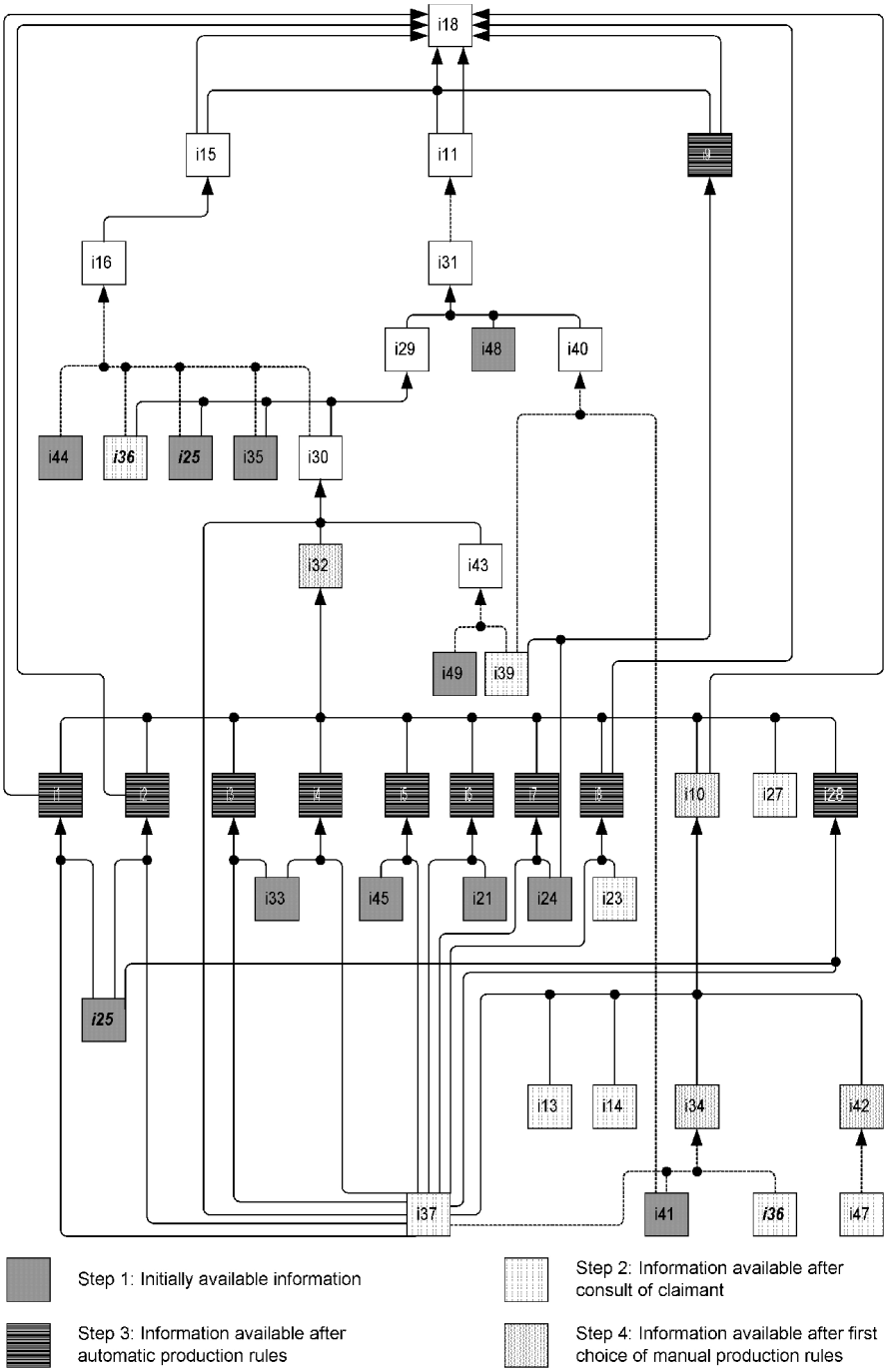

Figure 7. Cost-Optimal Plan of the SIC Product/Data Model

- Minimize the number of contacts with the claimant: as soon as *one* data element is required from the claimant, *all* data elements of the claimant are gathered.

We will approach the design by considering one imaginary case. This is valid as all cases will be treated equally. The design of the workflow takes place in five steps. For the first four steps, the available data elements are depicted according to the legend of Figure 7. All data elements of Figure 7 are available after the fifth step.

Step 1: Initially Available Information

As a start situation, the SIC holds the information i21, i24, i25, i33, i35, i41, i44, i45, i48, and i49. (See Figure 7.)

Step 2: Consult of Claimant

No automatic production rules can be performed on basis of the available information. Recall that all leaf data elements not readily available to the SIC have to be provided by the claimant. On basis of the third design objective, all these eight leaf production rules are executed. Having done that, the process execution up to that point has an average cost of 4.08 minutes (= 0.08 + 0.08 + 0.67 + 0.08 + 1 + 1.67 + 0.17 + 0.33) and the following information is available: i13, i14, i21, i23, i24, i25, i27, i33, i35, i36, i37, i39, i41, i44, i45, i47, i48, and i49. The additional data elements at this point are depicted in Figure 7.

Step 3: Automatic Production Rules

On basis of the available information, we first look for carrying out automatic production rules, as they can be applied without any cost. These are: (i1, {i25, i37}), (i2, {i25, i37}), (i3, {i33, i37}), (i4, {i33, i37}), (i5, {i37, i45}), (i6, {i21, i37}), (i7, {i24, i37}), (i8, {i23, i37}), (i9, {i24, i39}), and (i28, {i25, i37}).

The total available information is now: i1, i2, i3, i4, i5, i6, i7, i8, i9, i13, i14, i21, i23, i24, i25, i27, i28, i33, i35, i36, i37, i39, i41, i44, i45, i47, i48, and i49. The additional data elements at this point are depicted in Figure 7.

Step 4: First Choice of Manual Production Rules

Already, there is a probability of 0.04 (= 0.009 + 0.013 + 0.016 + 0.002) that the processing may end by an additional execution of one of the knockouts (i18, {i1}), (i18, {i2}), (i18, {i8}), or (i18, {i9}) in case either i1, i2, i8, or i9 are not satisfied (see Table 3). So, for 4 percent of the cases an average cost of only 4.08 minutes may be expected.

In the more general case that there is no knockout, processing must proceed. By now, there is no other option than to execute a manual production rule. We recall that these are (i11, {i31}), (i16, {i25, i30, i35, i36, i44}), (i17, {i25, i30}), (i34, {i36, i37, i41}), (i40, {i39, i41}), (i42, {i47}), and (i43, {i39, i49}).

Would these production rules be totally independent of each other, 6! = 720 different orderings should be considered for the rest of the design. However, we can limit the number of alternatives by inspecting the dependencies of the product/data model. In general, the optimal choice for a production rule is the one that increases the chance for a knockout at the lowest possible cost. For any of the remaining knockout possibilities, production rules (i34, {i36, i37, i41}) and (i42, {i47}) are required. In executing them there is no preference in their ordering: individual executions of these rules will not lead to any probability for a knockout. So, the following step in the process is to execute (i34, {i36, i37, i41}) and (i42, {i47}) in arbitrary order, followed by (i10, {i13, i14, i34, i37, i42}), which has no cost. There is a probability of 0.068 that the process will then end by applying (i18, {i10}). The process so far has an average cost of 8.58 minutes. On average, already 11 percent of the cases is knocked out.

If the execution of (i34, {i36, i37, i41}), (i42, {i47}), and (i10, {i13, i14, i34, i37, i42}) did not facilitate a knockout through (i10, {i18}), the automatic production rule (i32, {i1, i2, i3, i4, i5, i6, i7, i8, i10, i27, i28}) may be executed. Supposing that a value for i18 could not be obtained earlier, this makes the following information available: i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i13, i14, i21, i23, i24, i25, i27, i28, i32, i33, i34, i35, i36, i37, i39, i41, i44, i45, i47, i48, and i49. The data elements additionally available at this point are depicted in Figure 7.

Step 5: Final Choice of Manual Production Rules

Again, a manual production rule has to be chosen to be applied next. All remaining knockouts rely on the result of (i43, {i39, i49}), so this is our obvious next choice. It will facilitate the automatic execution of (i30, {i32, i37, i43}) followed by the execution of automatic production rule (i29, {i25, i30, i35, i36}).

Because i17 is no part of the COP, there is no alternative than to choose another manual production rule. We recall that the only remaining manual production rules are (i11, {i31}), (i16, {i25, i30, i35, i36, i44}), and (i40, {i39, i41}). Inspecting their dependencies, it is obvious that (i40, {i39, i41}) must precede (i11, {i31}). What is more, the scenario of subsequent executions of (i40, {i39, i41}), (i16, {i25, i30, i35, i36, i44}), and (i11, {i31}) is not a smart one. After all, if (i16, {i25, i30, i35, i36, i44}) is executed, a knockout may follow making the prior effort to execute (i40, {i39, i41}) superfluous. The only sensible scenario is to order the remaining manual production rules are:

- scenario 1: (i40, {i39, i41}), (i11, {i31}), (i16, {i25, i30, i35, i36, i44}),
- scenario 2: (i16, {i25, i30, i35, i36, i44}), (i40, {i39, i41}), (i11, {i31}).

With respect to scenario 1, it starts with the subsequent execution of (i40, {i39, i41}) and (i31, {i29, i40, i48}). With a probability of 0.85, production rule (i11, {i31}) can be successfully applied. If so, either knockout may follow by the single execution of (i18, {i11}) or by the thread of executions of (i16, {i25, i30, i35, i36, i44}), (i15, {i16}), and (i18, {i9, i11, i15}). Note that in the latter case, (i18, {i9, i11, i15}) is preferable over (i18, {i15}) because it has no cost and a success probability

of one. Now suppose after execution of (i40, {i39, i41}) and (i31, {i29, i40, i48}) that (i11, {i31}) could *not* be applied ($p = 0.15$). The only chance of obtaining a value for i18 is then that subsequent execution of (i16, {i25, i30, i35, i36, i44}), (i15, {i16}), and (i18, {i15}) is successful. The total cost for this scenario is 6.04 time units and it has a success probability of 0.88.

With respect to scenario 2, this scenario will start with the execution of (i16, {i25, i30, i35, i36, i44}), followed—if possible ($p = 0.997$)—by the automatic production rule (i15, {i16}). With a probability of 0.21, the knockout (i18, {i15}) can then take place. If either of the production rules (i15, {i16}) or (i18, {i15}) failed, then the subsequent execution of the production rules (i40, {i39, i41}), (i31, {i29, i40, i48}), and (i11, {i31}) should preferably take place. In addition, if (i15, {i16}) was not successful, then it may be followed by (i18, {i11}) with a slight success probability (0.079); otherwise, it is preferable to execute (i18, {i9, i11, i15}). The total cost of this scenario is 6.25 minutes and it has exactly the same probability to deliver a top value (0.88).

As follows from this comparison, the cost figures of these scenario's are almost similar, but the preferable alternative is scenario 1. Note that the evaluation of these scenario's is simplified because of the elimination of i17. Otherwise, the probability that i15 could have been determined on basis of i17 should have been taken into account.

The Final Design

This concludes the design of the workflow. The execution sequences of the workflow model are compactly represented with the workflow net in Figure 8. Petri nets are used for representing the workflow net (see, e.g., [1, 2]). Alongside or inside most tasks, an ordered enumeration of production rules is listed. The exceptions are the three tasks labeled with "skip," which have no production rules and are only added for logistic reasons. For the other tasks, the production rules are executed in the order of enumeration when the corresponding task is executed. The order of some of these rules is arbitrarily fixed. This is of no concern for the performance of the workflow as the case worker principle is applied, that is, all tasks for the same case are performed by the same worker. Each state in which different alternative routings can be followed is depicted as an OR-split. In italics, the condition is expressed for the corresponding routing. These conditions are derived from the constraints in Table 3.

Quality of the Design

For the sake of validation, a prototype was developed of an integrated information system that both supports the logistic routing of the workflow and the execution of its content, that is, the production rules. This prototype was also developed with ExSpect because of the relative ease to reuse the involved production logic. The subject of building such a prototype on basis of a formal product/data model is treated in Crom and Reijers [8].

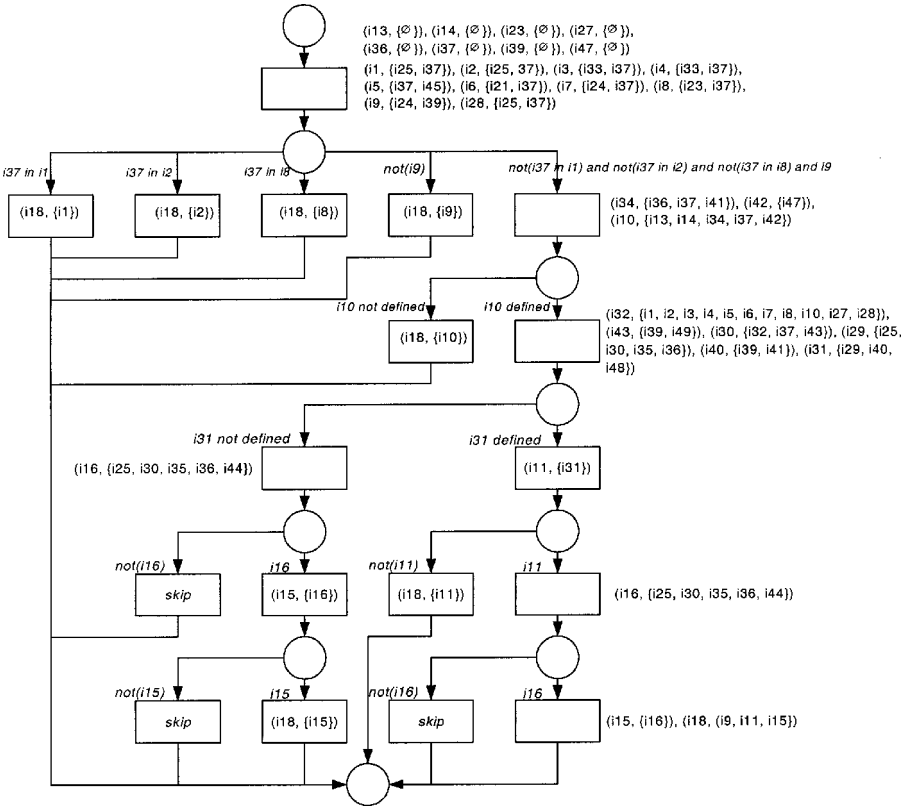

Figure 8. Final Workflow Design

It is important to distinguish at this point the two levels of use of this tool: the *meta-modeling* (modeling of the product data model and derivation of the COP: ExSpect/COP toolbox, see Figure 3) and the *actual modeling* level (prototype execution of the workflow). These levels are shown in Figure 9.

To the end user of the prototype, four different, predefined cases were presented that referred to real applications for an unemployment allowance and included real data. When using the prototype to enact the handling of such a case, tasks with automatic production rules were executed by the prototype itself. Tasks with manual production rules were to be performed by the end user of the prototype. SIC professionals used it in workshops held during the last weeks of 1999. Aside from some minor remarks, the prototyped design was accepted as reflecting a way of working that was sufficient and acceptable to determine the right for an unemployment allowance.

Evaluation of the validated workflow design pointed out that all design objectives were met by it. In particular, cost was drastically minimized. The average execution cost of the new workflow design for a single case was 14.34 minutes. This is a 73 percent reduction of the original average service time of 53.8 minutes of the real workflow. It means about 1,000 working days saved for the SIC company each year. This reduction was mainly due to the high level of automation within the design. To

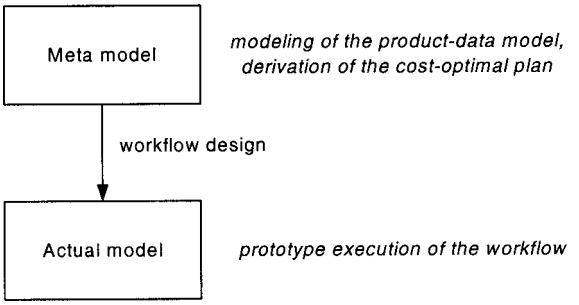

Figure 9. The Different Levels of Use of the ExSpect Tool

a lesser degree, the specific ordering of knockouts contributed to this decrease. In comparison with the existing workflow in operation, about three-fourths of its operations was automated. As a side effect, it was observed that for 10 percent of all cases, no human intervention at all would be required to determine the right for a claim. In this situation, we can speak of straight-through-processing or unattended workflow [21]. The effort in applying the PBWD method consisted in about 300 working days. This clearly indicates that the redesign effort resulted in a significantly better workflow even considering implementation costs.

No fair comparison can be made with the results of a participative redesign approach of the case study process, which takes the existing process as a starting point. The results of earlier redesign attempts within the SIC were not available to the authors. However, the deficiencies we attributed to prevailing design practice in the introduction—lack of design detail and difficulty to leave the original design behind—are no characteristics of the new design.

Lessons Learned

This case description shows that in specific circumstances exploring the complete search space for the most favorable workflow design is feasible. The COP did not significantly limit the size of the data elements to be considered in this case. In general, product/data models may become much larger, so that the role of the COP becomes more important. For example, the case described in Reijers [26] incorporates a product/data model of some 600 data elements. In this paper, we do not consider more general ways of deriving workflow designs within the bounds of a COP in detail, as it is a subject on its own. The general outline of a general design for a breadth-first workflow model can be sketched as follows. On basis of a COP and the product/data model, all execution sequences of production rules can be derived that have a positive probability to deliver a value for the top element. The model that pursues the simultaneous execution of all these execution sequences will lead to the quickest possible result and is therefore the optimal breadth-first model. All interleavings of this model can be used to determine a breadth-first workflow model that satisfies the targets for the used design criteria.

An important lesson learned from the case study about the applicability of PBWD is the importance of existing product specifications. PBWD is restricted to fields where a clear concept of the products to be delivered exists, as specified in handbooks, production procedures, customer brochures, and so on. This means that PBWD is more likely to be applied in legislative settings or within companies that already have some tradition on administrative recording, such as banks and insurance companies. Participative approaches do not suffer from this restriction. Nonetheless, it seems common sense that any company should decide first *what* to do, before it can consider *how* to do it best.

Conclusion

IN THIS PAPER, WE PRESENTED A NEW, analytical 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 possible quantification of design criteria have been given. A heuristic approach toward product-driven workflow design has been presented, as well as a case description that includes an actual workflow design.

By now, the consultancy firm Deloitte & Touche applied the method presented in this paper four times in client engagements in 1999, 2000, and 2001 (including the case study described in the eighth section). Business processes of a large Dutch bank and a national social security agency have been redesigned. In each occasion, at least a 50 percent cost reduction and a 30 percent flow time reduction has been achieved, while being accepted by the end users as valid and workable (see, e.g., [26]).

The issue whether PBWD as a method will be adopted by the business community is to some extent open. We already indicated that the ability to collect the data for the product/data model (see the third section) and the existence of product specifications (see "Lessons Learned" in the eighth section) are important conditions for its application. We feel that considering the risks involved for a company to adopt an innovative new approach, the number of engagements so far is hopeful. In our opinion, the success of the SIC company was not an incident, but the result of a rational, quantitative way of looking at process redesign. In the end, what will convince general management to try PBWD is *not* its technical merit but the method's practical success in similar settings. Until a substantial track record has been built up, it will primarily be great pressure—such as the situation of the SIC in the case study—that will move companies from the beaten path.

A practical boost for the ease with which PBWD can be applied would come from the development of tools to model product/data models and derive workflow models. A first step has been reached by the development of the ExSpect/COP toolbox that allows the modeling of the product/data model and the evaluation of the COP. Further research can lead to the incorporation of other relevant design criteria, relevant definitions, techniques, and algorithms.

We have demonstrated through the case study that the design method cannot be independent of the strategies companies may wish to adopt. We have described infor-

mally in which way a design method can be applied when defining design criteria and their relative importance. Future research will deal with the formal description of the design method and the generation of corresponding workflows.

Our workflow design approach so far has abstracted from the resource dimension. In particular, a workflow design generated with PBWD does not prescribe to which people work that is generated by executing the workflow should be allocated to. Both structural and dynamic demands on work distribution may be considered to be integrated in the design method (e.g., [4, 20]).

Finally, in the third section we compared the classical BOM with our product/data model for workflow processes. Current ERP systems allow for *generic/variant* BOMs [12, 16]. In this way, it is possible to deal with large product families having millions of variants, van Veen and Wortman [30]. As is shown in van der Aalst [1], 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. We stated that using cardinalities to represent that exactly the same piece of information is required multiple times does not make much sense (see the third section). However, in some cases, multiple *instances* of the same type of data element are required, for example, multiple eyewitness reports for a car damage claim. The product/data model presented in the second section does not take these cardinalities into account.

References

1. Aalst, W.M.P., van der. Generic workflow models: How to handle dynamic change and capture management information. In M. Lenzerini and U. Dayal (eds.), *Proceedings of the Fourth IFCIS International Conference on Cooperative Information Systems (CoopIS'99)*. Los Alamitos, CA: IEEE Computer Society Press, 1999, pp. 115–126.

2. Aalst, W.M.P., van der. On the automatic generation of workflow processes based on product structures. *Computers in Industry, 39,* 2 (1999), 97–111.

3. Aalst, W.M.P., van der. Reengineering knock-out processes. *Decision Support Systems, 30,* 4 (2000), 451–468.

4. Aalst, W.M.P., van der, and van Hee, K.M. *Workflow Management: Models, Methods, and Systems*. Cambridge: MIT Press, 2002.

5. Aalst, W.M.P., van der; Reijers, H.A.; and Limam, S. Product-based workflow design. In W. Shen, Z. Lin, J.P. Barthes, and M. Kamel (eds.), *Proceedings of the Sixth International Conference on CSCW in Design*. Ottawa: NRC Research Press, 2001, pp. 397–402.

6. Aalst, W.M.P., van der; de Crom, P.J.N.; Goverde, R.R.H.M.J.; van Hee, K.M.; Hofman, W.J; Reijers, H.A.; and van der Toorn, R.A. ExSpect 6.4: An executable specification tool for hierarchical colored Petri nets. In M. Nielsen and D. Simpson (eds.), *Application and Theory of Petri Nets 2000 (Lecture Notes in Computer Science 1825)*. Berlin: Springer Verlag, 2000, pp. 455–464.

7. Aldowaisan, T.A., and Gaafar, L.K. Business process redesign: An approach for process mapping. *Omega, 27,* 5 (1999), 515–524.

8. Crom, P.J.N., de, and Reijers, H.A. Using prototyping in a product-driven design of business processes. In A. D'Atri, A. Solvberg, and L. Willcocks (eds.), *Proceedings of the Open Enterprise Solutions: Systems, Experiences, and Organizations Conference*. Rome: Luiss Edizioni, 2001, pp. 41–47.

9. Davenport, T.H., and Short, J.E. The new industrial engineering: Information technology and business process redesign. *Sloan Management Review, 31,* 4 (1990), 11–27, 1990.

10. Deloitte & Touche Management and ICT Consultants. *ExSpect 6.2 User Manual*. Amsterdam: Bakkenist, 1998.

11. Deloitte & Touche Management and ICT Consultants. ExSpect. Amsterdam, 2001 (available at www.exspect.com).

12. Erens, F.; MacKay, A.; and Sulonen, R. Product modelling using multiple levels of abstraction—Instances as types. *Computers in Industry, 24,* 1 (1994), 17–28.

13. Hammer, M. Reengineering work: Don't automate, obliterate. *Harvard Business Review, 68,* 4 (July–August 1990), 104–112.

14. Hammer, M., and Champy, J. *Reengineering the Corporation: A Manifesto for Business Revolution.* New York: Harper Business, 1993.

15. Hee, K.M., van. *Information System Engineering: A Formal Approach.* Cambridge: Cambridge University Press, 1994.

16. Hegge, H.M.H. *Intelligent Product Family Descriptions for Business Applications.* Ph.D. dissertation, Eindhoven University of Technology, Eindhoven, 1995.

17. Herrmann, Th., and Walter, T. The relevance of showcases for the participative improvement of business processes and workflow management. In R.H. Chatfield, S. Kuhn, and M. Muller (eds.), *Proceedings of the Participatory Design Conference (PDI '98).* Palo Alto: CPSR, 1998, pp. 117–127.

18. Hofacker, I., and Vetschera, R. Algorithmical approaches to business process design. *Computers & Operations Research, 28,* 13 (2001), 1253–1275.

19. Hupp, T.; Polak, G.; and Westgaard, O. *Designing Work Groups, Jobs, and Work Flow.* San Francisco: Jossey-Bass, 1995.

20. Kumar, A; van der Aalst, W.M.P.; and Verbeek H.M.W. Dynamic work distribution in workflow management systems: How to balance quality and performance? *Journal of Management Information Systems, 18,* 3 (Winter 2001–2002), 157–193.

21. MacSweeney, G. Traveling the long road to end-to-end processing. *Insurance & Technology, 26,* 10 (2001), 30–34.

22. O'Neill, P., and Sohal, A.S. Business process redesign: A review of recent literature. *Technovation, 19,* 9 (1999), 571–581.

23. Orlicky, A. Structuring the bill of materials for MRP. *Production and Inventory Management, 13,* 4 (December 1972), 19–42.

24. Orman, L.V. A model management approach to business process redesign. *Journal of Management Information Systems, 15,* 1 (Summer 1998): 187–212.

25. Peppard, J., and Rowland, P. *The Essence of Business Process Re-Engineering.* Upper Saddle River, NJ: Prentice Hall, 1995.

26. Reijers, H.A. Product-based design of business processes applied within the financial services. *Journal of Research and Practice in Information Technology, 34,* 2 (2002) 34–46.

27. Reijers, H.A., and Goverde, R.R.H.M.J. Smooth cases: An application of product-based design [in Dutch]. Deloitte & Touche, Amsterdam, 1999.

28. Sharp, A., and McDermott, P. *Workflow Modeling: Tools for Process Improvement and Application Development.* Boston: Artech House, 2001.

29. Sherwoord-Smith, M. People centered process re-engineering: An evaluation perspective to office systems re-design. In B.C. Glasson, I. Hawryszkiewycz, A. Underwood, and R.A. Weber (eds.), *Business Process Re-Engineering: Information Systems Opportunities and Challenges. Proceedings of the IFIP TC8 Open Conference on Business Re-Engineering: Information Systems Opportunities and Challenges.* Amsterdam: Elsevier, 1994, pp. 535–544.

30. Veen, E.A., van, and Wortmann, J.C. Generative bill of material processing systems. *Production Planning and Control, 3,* 3 (1992), 314–326.