Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2001), *15*, 411–424. Printed in the USA.
Copyright © 2001 Cambridge University Press 0890-0604/01 $12.50

Process configuration: Combining the principles of product configuration and process planning

KARSTEN SCHIERHOLT*
Swiss Federal Institute of Technology (ETH) Zürich, Logistics and Information Management, Zürich, Switzerland

(Received February 1, 2000; Accepted May 7, 2001)

Abstract

Product configuration is the process of generating a product variant from a previously defined product family model and additional product specifications for this variant. The process of finding and sequencing the relevant operations for manufacturing this product is called process planning. This article combines the two principles in a new concept of process configuration that solves the process planning task using product configuration methods. The second section develops characteristics for two process configuration concepts, the interactive process configuration and the automation-based process configuration. Following an overview of the implementation of a process configuration system, the results of a case study in the aluminum rolling industry are presented. The main benefits of the process configuration concept are observed in a reduced knowledge-maintenance effort and in increased problem-solving speed.

Keywords: Generative Constraint Satisfaction Problem; Multiple-Variant Products; Process Configuration; Process Planning; Product Configuration

1. PROCESS CONFIGURATION AS AN ENHANCED PROCESS PLANNING CONCEPT

1.1. Multiple-variant products

Manufacturing companies have in recent decades continually diversified their products and thus responded to increasing competition by more and more respecting the individuality of customer needs. Products that are very similar in their general structure but differ in the details of each customer-specific variant are grouped to the more general constructs of multiple-variant products (Schwarze, 1996) or product families (Erens, 1996). While the number of variants in a product family grows over time, the number of times that a certain variant is built declines. Since the generation of each new variant generally requires some design work, the total amount of design work within the life cycle of a product family increases as well. This design work very often consists in a repeated routine task, such as the calculation of parameter values, the completion of parameter lists, or the verification of consistency of customer requirements.

1.2. Product configuration and process planning as two principles for generating production data

A product configurator can perform the design task necessary to produce a new product variant. Mittal and Frayman (1989) define the configuration task in general as a design activity with the key feature that the designed artifact is assembled from a predefined set of components that can only be connected in certain ways. Both the components and the possible relations between them are known at the start of a configuration process. A product configurator is a tool that supports the user during the configuration process or that performs the configuration process automatically. The use of a configurator not only improves the variant generation process but also contributes to knowledge preservation and knowledge consolidation within a company (Schwarze, 1996).

Configuration and design tasks (Brown & Chandrasekaran, 1989) are generally classified as:

- *Routine tasks:* well-defined problems where solution approaches are known, but problems cannot be solved by simple straightforward algorithms,

*Currently at the SAP AG, Walldorf, Germany.
Reprint requests to: Karsten Schierholt, Swiss Federal Institute of Technology (ETH) Zürich, Logistics and Information Management, Zürichbergstrasse 18, CH 8028 Zürich, Switzerland. E-mail: Karsten.Schierholt@ethz.ch

https://doi.org/10.1017/S0890060401155046 Published online by Cambridge University Press

- *Innovative tasks:* the existing knowledge must be extended during the solving process, or
- *Creative tasks:* weakly defined goals and solution procedures; new solution components need to be developed or introduced to find a solution.

Today, configuration is applied mainly to the generation of product variants, particularly product variants of assembled products. The manufacturing processes of these products are so closely related to the selected components that their determination, based on the component structure of these products, is hardly a problem. For other kinds of products, this is not necessarily valid. As soon as not the assembly but rather the treatment of one work piece is the predominant subject of planning, the determination of necessary manufacturing steps is regarded as a planning problem, more specifically, as a process planning problem, rather than as a configuration problem.

Process planning is defined as the task of finding the relevant processes for manufacturing a product, sequencing these processes, and defining the complete set of parameters for each process. Process planning is therefore the task of precisely specifying how to manufacture a particular product (Schlenoff et al., 1996). Within operations management, process planning is part of operations planning.

The variety of process planning tasks in practice ranges from filling certain parameters into an existing fixed structure of operations to the generation of new process plans for each new variant of a product family. Computer-aided process planning (CAPP) systems are currently used only in a small area of process planning. A 1992 study "raises the assumption that IT-support for NC-programming is more widespread than for process planning. Secondly, the automation of IT-supported process planning seams to be on a very low level. Planning tasks more difficult than the administration of plans are with few exceptions supported only by specialised process planning systems" (p. 113) (Hamelmann, 1996). Since few real-world process planning tasks are of the routine kind, enhanced support of planners in their work bears great potential. This support can take effect in the automated solving of subtasks of the planning problem, in structuring and presenting knowledge for easier maintenance, or in giving decision support to the human planner during problem solving.

1.3. Process configuration as a combination of both principles

There are many similarities between product configuration and process planning, such as the goals of automation of routine tasks and standardized storing of knowledge or the construction of a structure out of components. A main difference is that the sequence of the components and the order dependencies of components are neglected aspects in product configuration. In manufacturing applications that deal mainly with the treatment of one work piece, the structure of process plans within a certain product family is highly similar. Two forms of similarity can be distinguished:

- Manufacturing process plans vary in their structure, that is, in the set of manufacturing processes and their sequence for manufacturing the product.
- Process plans of the same structure can vary in the manufacturing process parameters.

Using this similarity, it is not farfetched to use the strengths of the configuration approach for transferring the principles used in product configuration to the structurally similar problem of process planning, which in this context is called *process configuration*. Single manufacturing processes form the basic components of the problem. As in product configuration tasks, new types of components (i.e., processes) will not be created during the configuration process.

While a maximum bill of material together with associated production rules defines the knowledge about a product family in a product configuration problem, the process configuration approach uses a *plan skeleton* for describing the knowledge of the process plan family, that is, of all process plans related to product variants of that product family.

Section 2 reviews environment characteristics for process configuration problems and identifies application areas. A detailed description of the structure of plan skeletons is given in Section 3, while Section 4 reports on the use of process configuration at a manufacturer of aluminum sheets and coils.

2. CHARACTERISTICS OF PROCESS CONFIGURATION PROBLEMS

2.1. Environment characteristics for the process configuration task

Process configuration is a task not always suited to receive great attention during order processing. Often, the generation of process plans is trivial, but certain characteristics of a product can make it more important to deal with the configuration of process plans. A morphologic scheme is used to identify these relevant areas.

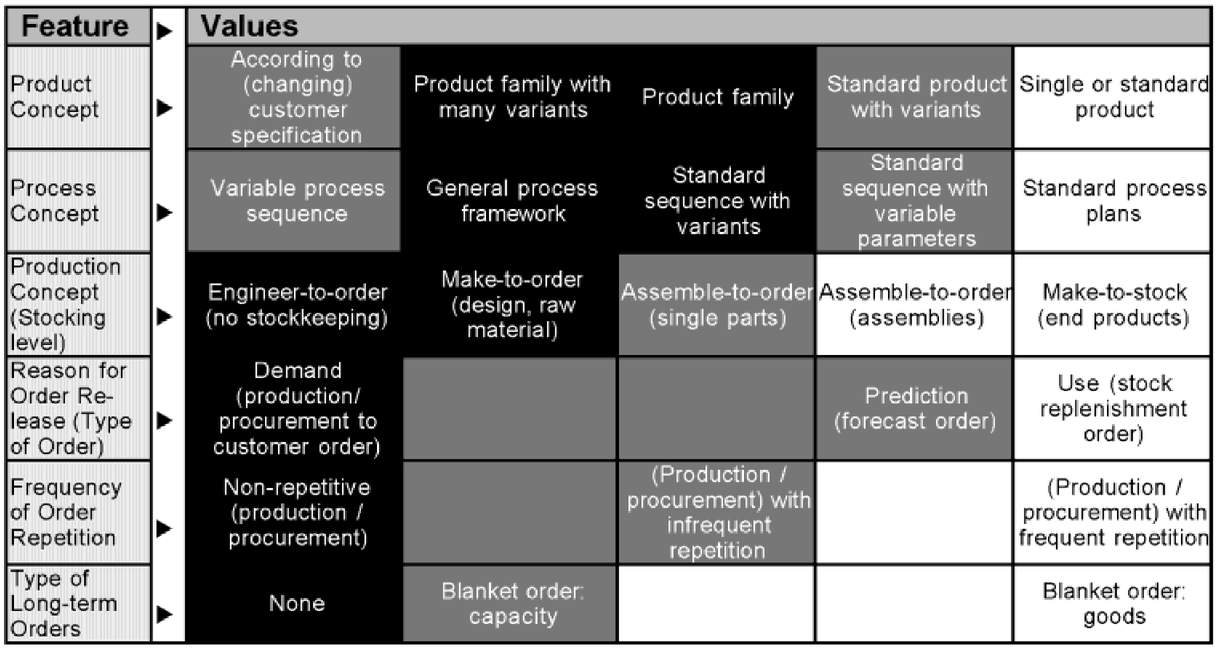
Morphologic schemes are sets of features with each feature having certain possible values that allow analysis and evaluation of all possible solutions to a particular problem in the given feature dimensions. The evaluation of an object with respect to the given features leads to a pattern, which can be analyzed for finding interrelations between these features. More important, typical patterns can be associated with specific assignments on how to deal with such groups of objects. In the area of planning and control, several morphologies have been developed to find suitable reference business processes and planning and control methods for products and product families with certain feature values. Using a slightly extended morphology based on Schönsleben (2000), a brief characteristic of those products and product families is given for which process configuration seems applicable. One feature, the *process concept*, is added. The five original features are explained only briefly; further explanations regarding feature values are given by Schönsleben (2000).

- The *product concept* determines the strategy for developing the product and offering it to the customer. The product concept selects the degree of variant orientation.
- The *stocking level* defines that level of the value-added chain above which a product (component) can be produced within the time to delivery, or in accordance with demand. For goods below and at the stocking level, no exact demand is known. Demand forecast is required. Each stocking level is closely associated with a *production concept.*
- The *reason for order release* is the origin of demand. The *type of order* indicates the origin of demand that resulted in the order.
- The *frequency of order repetition* states how often, within a sufficiently long time period, a production or procurement order for the same product is placed.
- The *type of long-term orders* describes the manner in which long-term planning is conducted in the logistics network.

The new feature, *process concept*, defines the strategy for developing the manufacturing process plans. The values are defined very similarly to the values of the feature product concept mentioned above:

- *Standard process plans:* The processes necessary to manufacture the product are always the same, always appear in the same sequence, and have fixed process parameters. Such process plans are usually used in the production of standard products with no or only a few variants.
- *Standard process sequence with variable parameters:* The processes necessary to manufacture the product are always the same and always appear in the same sequence, but might have variable process parameters depending on the specified product variant.
- *Standard process sequence with variants:* A standard process sequence for manufacturing product variants of a product family exists. With respect to the specified variant, some processes of the standard sequence might be skipped or added or alternative processes chosen. There is thus some variance in the process plan structure.
- *General process framework:* A general process framework exists for all product variants. The framework is filled with manufacturing processes from a predefined set. The sequence within the framework might vary.
- *Variable process sequence:* No predefined sequence of processes exists. The set of manufacturing processes used might be extended on demand.

Using these features, relevant areas for the use of process configuration are identified in Figure 1. Black fields suggest full applicability, while shaded fields define feature values for which process configuration is only partially applicable. It is not a coincidence that production concepts suitable for process configuration are very similar to those concepts that are generally increasingly suitable for variant-oriented concepts (Schönsleben, 2000). The individuality



Feature	►	Values				
Product Concept	►	According to (changing) customer specification	Product family with many variants	Product family	Standard product with variants	Single or standard product
Process Concept	►	Variable process sequence	General process framework	Standard sequence with variants	Standard sequence with variable parameters	Standard process plans
Production Concept (Stocking level)	►	Engineer-to-order (no stockkeeping)	Make-to-order (design, raw material)	Assemble-to-order (single parts)	Assemble-to-order (assemblies)	Make-to-stock (end products)
Reason for Order Release (Type of Order)	►	Demand (production/ procurement to customer order)	▨	▨	Prediction (forecast order)	Use (stock replenishment order)
Frequency of Order Repetition	►	Non-repetitive (production / procurement)	▨	(Production / procurement) with infrequent repetition		(Production / procurement) with frequent repetition
Type of Long-term Orders	►	None	Blanket order: capacity			Blanket order: goods

Fig. 1. Values of product and production features for process configuration.

of product variants and their manufacturing processes is a major reason for using process configuration concepts: The more uniquely products are manufactured, the better the fit of process configuration as a method for coping with the increased complexity of the process planning task.

At first sight, process configuration looks useful for manufacturers of multiple-variant products that also require variance in the manufacturing process. Still, the process configuration task in itself varies greatly in complexity, growing from a standard sequence of manufacturing processes with variable parameters to a completely variable process structure. Within this range, the requirements of the process configuration task vary as much.

A second morphologic scheme is now presented to further detail the characterization of the process configuration task for a given product family with process-related features of process configuration. In addition to the previously introduced feature process concept, four other features are added to the morphologic scheme. Their values are explained briefly:

- *Alternative process plans:* On a scale ranging from *no alternatives*, *few alternatives*, and *some alternatives* to *many alternatives*, this feature describes how many technically valid alternative process plans exist to manufacture one specified product variant. The existence of alternatives calls for methods that rate these alternatives for selecting an optimal or near optimal process plan.
- The *complexity of process configuration knowledge* characterizes the knowledge necessary to create the process plan from the product data.
    - *Simple complexity* expresses the fact that common IF–THEN rules or similarly defined constraints are sufficient to express all knowledge needed to define the existence of a process. Rules and perhaps formulas for calculating process parameters can be evaluated by using only customer order data.
    - *Fair complexity* is given when, in addition, interdependencies between processes occur that are defined by consistency rules.
    - Once rules and formulas depend not only on the product specification but also on the data generated for other processes, process configuration knowledge becomes *highly complex*. Interdependencies between processes might be cyclic. Methods for resolving possible conflicts must be available (Alder, 1991).
    - *High complexity with incomplete knowledge* exists in cases where the product family is not completely defined for all its possible product variants. In this case, methods allowing manual interaction are absolutely required.
- The *configuration task classification* distinguishes between *routine configuration*, *innovative configuration*, and *creative configuration* according to the definitions mentioned earlier.
- The amount of user influence during the process configuration task is described according to the feature of *degree of automation for plan generation*. Possible values of this feature are:
    - *Fully manual:* Plans are built manually in a generative way.
    - *Case-based with manual modifications:* The process configuration system supports the planner by finding relevant cases from a case base, which are then adapted manually to fit the product requirements.
    - *Case-based or skeleton-based with supported modifications:* The process configuration system supports the planner by finding relevant cases or plan skeletons, which are adapted and detailed in an interactive way. The system performs tasks such as the calculation of parameters and consistency checks.
    - *Automated generation with possible interaction:* Process plans are generally generated automatically. In difficult cases, where the process configuration system cannot find a solution, instruments for guiding the search by the user can be used.
    - *Fully automated:* All plans are generated automatically. Except for changing the underlying process configuration knowledge base, there are no supported interactions.

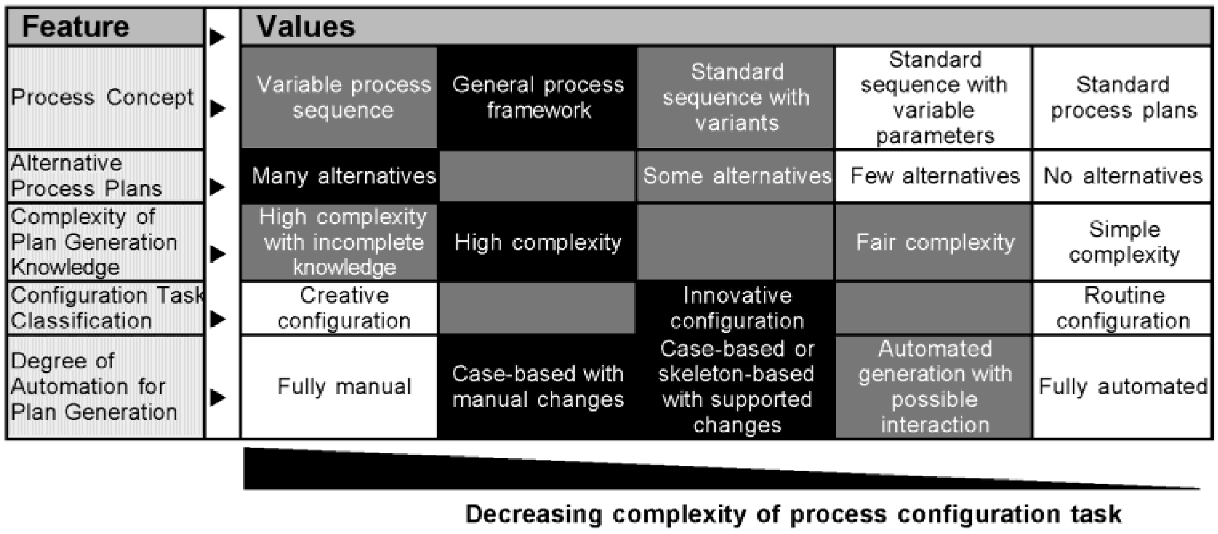
Using this morphologic scheme, the following two subsections identify and describe two typical scenarios for applying process configuration. One is based predominantly on user interaction, while the other is automation oriented.

2.2. Interactive process configuration

The scenario of interactive process configuration aims to define a decision-making process for configuration tasks in which user input data and data generated by the process configuration system are integrated. Figure 2 shows typical values of features where this scenario is applicable.

Typical areas of application for interactive process configuration are found in production schemes where the complete knowledge necessary for defining process plans cannot be made available to the planning system. This can be due to reasons such as:

- *Lack of structured knowledge:* The planning task requires not only technical knowledge but also knowledge about the environment that is not structured enough to be defined in term of rules that can be evaluated by a system. An example of this kind of knowledge is the handling of exceptions.
- *Fast changing knowledge:* Some areas of the knowledge base are so dynamic that the maintenance of the respective structured knowledge requires an effort much



Feature	Values				
Process Concept	Variable process sequence	General process framework	Standard sequence with variants	Standard sequence with variable parameters	Standard process plans
Alternative Process Plans	Many alternatives		Some alternatives	Few alternatives	No alternatives
Complexity of Plan Generation Knowledge	High complexity with incomplete knowledge	High complexity		Fair complexity	Simple complexity
Configuration Task Classification	Creative configuration		Innovative configuration		Routine configuration
Degree of Automation for Plan Generation	Fully manual	Case-based with manual changes	Case-based or skeleton-based with supported changes	Automated generation with possible interaction	Fully automated

Decreasing complexity of process configuration task

Fig. 2. Features and their values in an interactive process configuration scenario.

greater than the benefit. This is valid also for rarely used knowledge.

In the cases mentioned above, it is impossible or at least not feasible to keep the knowledge base up-to-date and complete at all times. Here the incompleteness of the knowledge base should be accepted and the system-supported process configuration task designed interactively.

Process configuration patterns that rate even higher on the complexity scale, that is, appearing further to the left in the morphologic scheme in Figure 2, seem hardly suited for support by a process configuration system. The individuality of the performed task and the knowledge used present a situation where the initial effort for developing a general process framework does not pay off. In that case, a completely manual process configuration for each manufactured product is more appropriate.

Interactive process configuration depends strongly on communication between the user and the process configuration system. Two aspects are important for a successful system:

- A suitable user interface that supports the user in the tasks to be performed, and
- A level of communication between user and system that is easy enough to be used intuitively but at the same time expressive enough to describe even complex knowledge structures. This knowledge structure level used for presentation is not necessarily the same level used for reasoning by the system. A mapping can take place in between.

The implementation of an interactive process configuration also requires highly skilled planners with profound knowledge of their domain. They are only marginally supported in their decision making by the system. On the other hand, the maintenance of the process configuration knowledge is quite easy, since only a small and little interdependent knowledge base remains to be maintained.

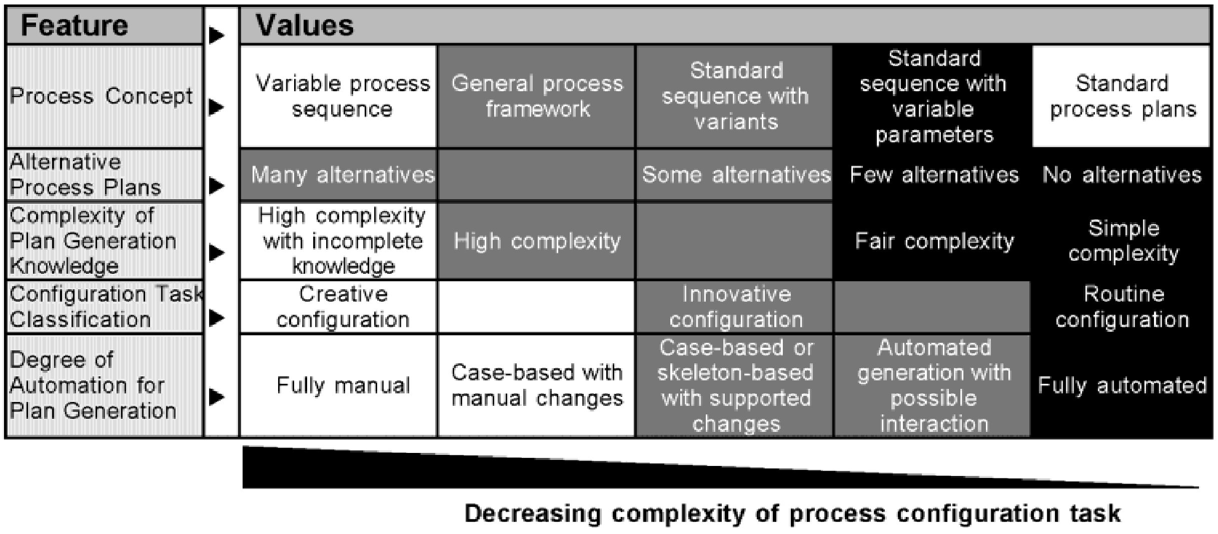
2.3. Automation-based process configuration

In contrast to the interactive process configuration scenario, which is highly dependent on manual interference during the process configuration task, the scenario of automation-based process configuration assumes automated process plan generation of most standard product variants but also allows manual interference in cases of exceptions or special orders. Figure 3 shows typical values of features of the process configuration morphology where such a scenario is applicable.

The scenario of automation-based process configuration is found in applications where rules, constraints, and formulas are sufficient to express the technical process configuration knowledge. Here process configuration knowledge of only fair complexity typically remains stable for longer periods than configuration knowledge of higher complexity. The product family definition is rarely extended due to product changes or additional customer requests. Process structures within a product family remain similar over the life cycle of that product family.

Even though the process configuration task in this scenario is designed to be performed in an automated way, an interface for handling exceptions and for changing configured process plans is inevitable. Including manufacturing, which uses the results of the process configuration, and system development, Figure 4 shows four interest groups that each place different requirements on an automation-based process configuration system (Schierholt, 1998):

- *Manufacturing:* Most important is the correctness of process plans. The plan should include all process in-



Feature	Values				
Process Concept	Variable process sequence	General process framework	Standard sequence with variants	Standard sequence with variable parameters	Standard process plans
Alternative Process Plans	Many alternatives		Some alternatives	Few alternatives	No alternatives
Complexity of Plan Generation Knowledge	High complexity with incomplete knowledge	High complexity		Fair complexity	Simple complexity
Configuration Task Classification	Creative configuration		Innovative configuration		Routine configuration
Degree of Automation for Plan Generation	Fully manual	Case-based with manual changes	Case-based or skeleton-based with supported changes	Automated generation with possible interaction	Fully automated

Fig. 3. Features and their values in an automation-based process configuration scenario.

formation as well as possible references to manufacturing instructions that must be obeyed. Determination of alternative machines and process tolerances help operators to interpret their action potential.

- *Planning:* A requirement important to human process planners is transparency of the reasoning steps taken by a computer-aided process planning system in developing the process plan. This is essential for eventual manual additions or changes to the plan. In addition, the planning system should be flexible enough to allow planning system-human planner interaction during the planning task execution. Interaction, on the other hand, requires a reasonably fast system reaction time.
- *Knowledge engineering:* Knowledge engineers are looking for a powerful form of knowledge structuring that allows them to define general principles on a very abstract level but also enables a simple representation of specific exceptions to these principles. An important aspect in this context is the definition of a comprehensive structure of knowledge. Dependencies within the knowledge structure should be as visible as possible. This helps to determine the side effects of any changes in the knowledge base and thus reduces test effort.
- *System development:* The requirements of system developers are somewhat similar to those of knowledge engineers. On top of that, the reasoning tools available in the system should be both powerful enough for

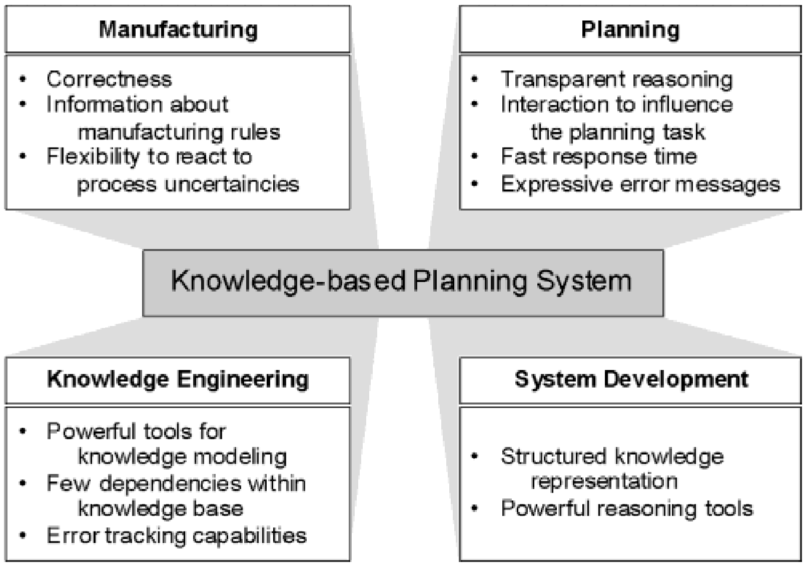

Manufacturing
• Correctness
• Information about manufacturing rules
• Flexibility to react to process uncertaincies
Planning
• Transparent reasoning
• Interaction to influence the planning task
• Fast response time
• Expressive error messages
Knowledge-based Planning System
Knowledge Engineering
• Powerful tools for knowledge modeling
• Few dependencies within knowledge base
• Error tracking capabilities
System Development
• Structured knowledge representation
• Powerful reasoning tools

Fig. 4. Key requirements placed on an automation-based process configuration system by different user groups.

achieving good performance of the planning task and expressive enough to define efficient problem solving strategies.

Again, even a mostly automated process configuration system relies strongly on the interfaces offered to the different kinds of users. The following section suggests a visually based representation language for knowledge needed to perform the process configuration task.

3. IMPLEMENTATION OF PROCESS CONFIGURATION

3.1. Knowledge representation for automation-based process configuration

Product configuration systems usually use a generic product structure for representing the knowledge about a multiple variant product. In addition to a common product structure, a generic product structure consists not only of the components of one product variant, but also includes all possible alternative components within the given product family. For each alternative component there is a rule that states the conditions under which this component is part of the product structure of a specific product variant. Other rules may calculate parameters for a component or check compatibility among components. The concept of generic product structures is explained in further detail in Erens (1996) or Schönsleben (1985).

A direct conversion of generic product structures to generic process structures for process configuration is not possible. Rules for defining the existence of processes, their parameters, and compatibilities among them are still required. In contrast to product configuration, which includes the problem of component or part selection (Darr & Birmingham, 2000) and component arrangement, process configuration has to consider an additional dimension: time, or the sequence of processes.

Generic product structures use hierarchical maximum bills of material with associated production rules for defining product family knowledge. To describe generic process structures with knowledge about sequences of processes, directed graphs are more appropriate. These graphs are called *plan skeletons*. Each node in a plan skeleton represents a process. The edges define possible predecessor or successor relationships in a process plan. Every path from a defined starting process to a defined final process is a possible process plan. The bases of a plan skeleton are process type components that are hierarchically structured in a process-type hierarchy. Associated with these process types are calculation functions that define input–output relations of the manufacturing processes. A process can appear as mandatory (displayed darker) or optional (lighter) in a plan skeleton. When mandatory, they have to be part of every process plan variant; optional process have associated existence rules that determine whether this process is needed in a specific context or not. Insert processes are processes that are modeled outside the graph structure, because their positions in the process plan variant cannot be determined in advance. They will be determined during problem solving through evaluating insertion decision tables. An example of a plan skeleton of the aluminum sheet and coil manufacturer application is given in Figure 5. Figure 6 shows the ramification decision table for the warm rolling node, and Figure 7 shows a possible insertion decision table.

Different categories of process-planning knowledge are modeled in plan skeletons and the associated process type hierarchy. A formal definition is given in Schierholt (2000):

- The *graph structure* provides a generic model of process plan variants in a product family.

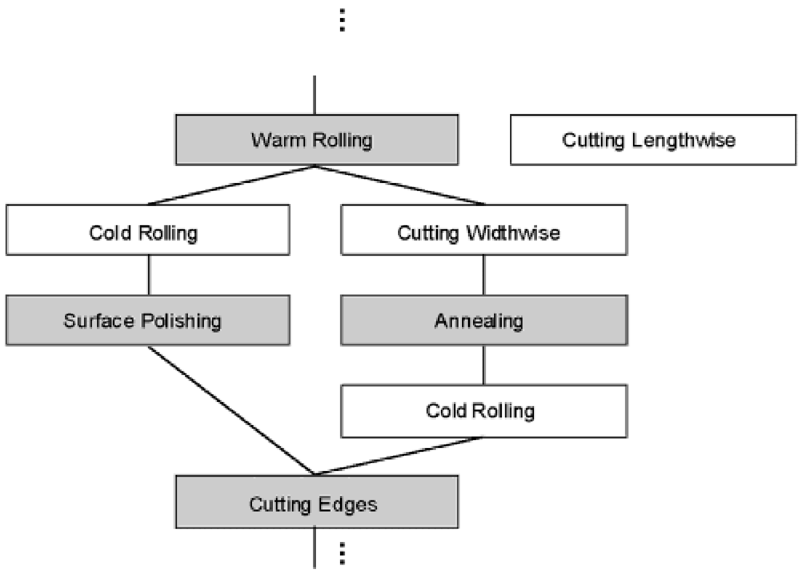
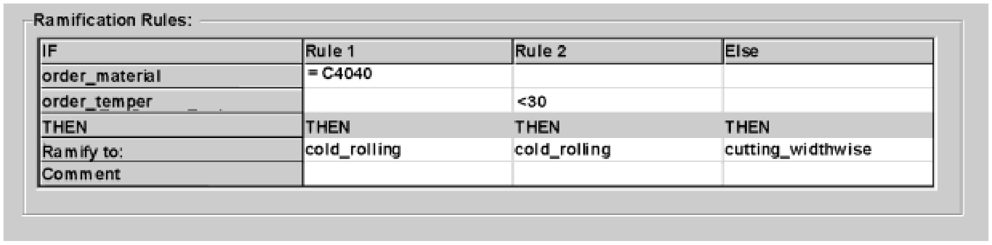


Fig. 5. Example plan skeleton.



Ramification Rules:

IF	Rule 1	Rule 2	Else
order_material	= C4040		
order_temper		<30	
THEN	THEN	THEN	THEN
Ramify to:	cold_rolling	cold_rolling	cutting_widthwise
Comment			

Fig. 6. Ramification decision table for the "Warm Rolling" node.

- A *set of process types* used in one or more plan skeletons with process parameters defined for each of the process types provides the elements of the graph.
- *Calculation functions* define relations among process parameters in or between processes of certain process types.
- The *process type hierarchy* orders the process types hierarchically. It allows abstraction in modeling process parameters and calculation function definitions.
- *Ramification rules* specify the path to be taken in the graph structure for a specific process plan variant.
- *Existence rules* determine the existence of optional processes in a path.
- *Insert processes* with insertion rules determine eventual dynamic extensions to the graph structures in process plan variants.

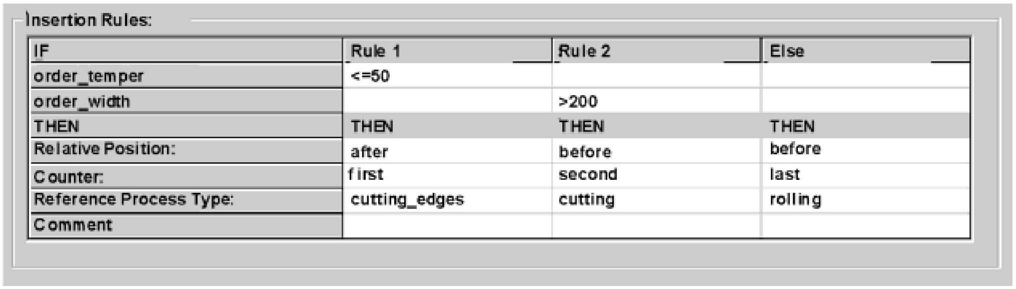
When related to the general classes of configuration knowledge, all of these categories belong to the class of problem-independent knowledge (Klein et al., 1994). The knowledge about process types and their associated parameters as well as the hierarchical order or these process types belong to the subclass of component knowledge. All other knowledge categories belong to constraint knowledge and (in-)compatibility rules.

Not all of the seven knowledge categories are needed for modeling process configuration knowledge with plan skeletons. Process types with their parameters, calculation functions, the graph structure, and ramification rules are sufficient for that purpose. When using only these four knowledge categories, however, a lack of comprehensibility that is needed for knowledge maintenance is obvious.

The three additional knowledge categories, the hierarchy of process types, optional processes with existence rules, and insert processes, only ease the modeling and the visualization of plan skeletons. They do not allow modeling of new classes of knowledge, such as functional knowledge or fuzzy knowledge. Plan skeletons that use the additional knowledge categories can always be transferred into those that do not use them—with all the disadvantages with respect to clarity and readability of the plan skeletons.

It is thus clear that, to a certain extent, process-planning knowledge within one class of knowledge can be modeled in various ways; that is, the same facts of knowledge can be modeled using different categories of knowledge. The knowledge engineer thus has some degrees of freedom in modeling, and individual modeling styles are possible. The list of possible knowledge categories used for modeling plan skeletons also has no clear limits. New categories might enable the modeling of plan skeletons with even better comprehensibility.

Plan skeletons seem to be expressive enough for representing even complex process structures of product families with many variants. Most rules relating to sequencing knowledge are encapsulated in the graph itself. Plan skeletons also display an easy way of understanding process plans while at the same time guaranteeing a minimum of structured knowledge definition that can be processed by an automated system. Plan skeletons are thus well suited to serve as a mediator language between the user and the system.



Insertion Rules:

IF	Rule 1	Rule 2	Else
order_temper	<=50		
order_width		>200	
THEN	THEN	THEN	THEN
Relative Position:	after	before	before
Counter:	first	second	last
Reference Process Type:	cutting_edges	cutting	rolling
Comment			

Fig. 7. Insertion decision table for the "Cutting Lengthwise" node.

3.2. Solving process configuration tasks as constraint satisfaction problems

Constraint satisfaction techniques form the foundation of the configuration approach described in Mittal and Frayman (1989), which focuses on the assembly of systems by connecting components. However, the basic finite and discrete constraint approach is limited in expressiveness and does not provide hooks for representing important features of configuration problems. These features are the organization in terms of components and the dynamics of the problem—that is, the fact that the number of components in a solution may change during problem solving.

Mittal and Falkenhainer (1990) proposed dynamic constraint satisfaction problems (dynamic CSP) to avoid the latter problem. Dynamic CSPs extend a finite CSP to allow constraints on both values of a variable and its relevance to a solution. In a dynamic CSP, not all variables have to be assigned to solve the problem. This is addressed by the introduction of a second type of constraint. While compatibility constraints represent the constraints known from the finite CSP, activity constraints require a variable to be active or not active based on other variables' activity and value assignments. The introduction of activity constraints does not resolve the problem of limited components, since all possible components still need to be defined in the problem specification. In configuration problems, defining the maximal set of possible components may be impossible (Mailharro, 1998). At the same time, neither CSPs nor dynamic CSPs allow for efficient support of component hierarchies.

Component-oriented configuration approaches are an extension to the modeling approaches based on CSPs or dynamic CSPs. Component-oriented configuration introduces component types as the central object, where each type determines the structure and constraints of its instances, the components. Component types are typically organized in object-oriented inheritance hierarchies and thus allow reasoning on an abstract level. The number of component instances is not limited (Sabin & Weigel, 1998).

Configuration problems are represented as generative CSPs, in which components and their attributes can be generated as needed. New instances of a component type also trigger the generation of attribute variables and constraints that are defined on this component type. The new component may in addition be restricted by previously existent constraints. Connections between components, described in the form of ports, are represented using sets with nonfinite domains to allow as many components as necessary to connect. At the same time, a generative CSP framework supports resource-balancing constraints, where resource demands and supplies are defined through component attributes (Mailharro, 1998; Stumptner et al., 1998).

For problem solving, plan skeletons have to be compiled into a generative CSP, the *process configuration constraint model*, and then solved using standard CSP algorithms with application-specific search heuristics. The definition of the process configuration constraint model is performed in three phases. The first phase is model independent; that is, the plan skeleton chosen as the basis of the problem instance is of no relevance. During the second phase, constraints are generated that are related to the plan skeleton, but still independent of the problem instance. Only in the third phase is problem-specific data used for specifying further constraints.

The *first phase* starts with defining a component type hierarchy on the basis of the process type hierarchy used in modeling plan skeletons. After that, port variables are added to the root component type and thus inherited by all component types on lower levels. These ports, the direct and global predecessor and successor sets of a component in a process plan variant, are the main variables that are constrained for finding a valid sequence of process components for a given problem. In a solution, at most one component may be connected to the direct predecessor and successor port, as this is necessary in order to guarantee that the process sequence is unambiguous. Attributes, their possible value domains, and calculation functions that define relations between input and output parameters of a component are defined next in the constraint model. These definitions may be specific to only parts of the component type hierarchy, such as, for example, only the rolling processes. The necessary information is taken from the process type hierarchy in the plan skeleton model.

The *second phase* constrains the model further by using input from a specific plan skeleton. Except for the direct predecessor port of the component type representing the source node of the plan skeleton and the direct successor port of the component type representing the destination node of the plan skeleton, exactly one component must connect to all direct successor and predecessor ports in a valid solution. No component may connect to the other two ports. After that, the number of possible components of a specific type is restricted to the count of appearances of the related process type in the plan skeleton. If a process type is modeled as absolutely mandatory in a plan skeleton—that is, it is part of every process plan variant developed from such a plan skeleton—minimum bounds are defined as well. The structure of the plan skeleton is then redefined by constraints on component types that restrict the possible component types that may connect to the direct predecessor and successor ports. At the same time, it must be assured that mandatory parts of the plan skeletons cannot be skipped in a solution due to the use of insert processes that until now have unrestricted predecessor and successor sets. To avoid this, a further constraint requires that for every component type in the plan skeleton, at least one of the possible direct successors must also be in the global successor set. Such a constraint ensures that all mandatory process on a path are included in a process plan variant.

In the last step of the second phase, the information given in the existence, the ramification, and the insertion decision tables is also added to the process configuration constraint model. Rules in a decision table are transformed into conjunctive statements that are then defined as constraints. Since decision tables are defined as first-hit decision tables with rules being evaluated from left to right, the action part of the second rule may only be activated if the condition of the first rule is false, and the condition of the second rule is true. These statements are easily defined as logical constraints equal in all kinds of decision tables. The action part of a rule, on the other hand, is different between the decision tables. While in existence decision tables a component of a specific type is required or rejected to be part of the process plan variant solution, the action part of ramification decision tables triggers constraints that reduce the set of possible successor component types at a dividing node in the plan skeleton. In insertion decision tables there are three inputs given in a rule conclusion (see also Fig. 7). Each rule leads to two constraints. One defines the type of the direct predecessor or successor component of the inserted process in the process plan variant. The other one defines the exact reference component by defining its count (from the top to the bottom of the process plan variant). With this choice it is possible to require a process to be inserted after any rolling process, but specifically, for example, after the third rolling process in the process plan variant, regardless of which of the rolling processes in the plan skeleton are finally part of the solution.

In the *third phase* of model definition, the problem-specific parts are added to the constraint model. These parts are basically the order parameters for the specific product variant to be manufactured, which also form the input special to this problem instance.

A formal definition of all constraints in these three phases is given in Schierholt (2000), which gives the details of how the example given in Section 3.1 is modeled in a process configuration constraint model. The definition of the process configuration constraint model was implemented using optimization programming libraries by ILOG, the ILOG Solver and ILOG Configurator. These tools eased the model definition and search strategy formulation by providing data structures with embedded constraint propagation algorithms and with predefined basic search strategies that were adapted to the specific needs of the process configuration task.

3.3. Solving the process configuration constraint model

The process configuration constraint model as defined in three phases in the previous subsection is solved in two steps. The first step performs the initial propagation of the constraint model. Components that are absolutely necessary to solve the model are generated, and their type, port, and attribute variables constrained to ranges where no constraints are violated. This step is performed completely by a constraint propagation algorithm without further guidance. The result of this first step is a reduced, but still large, search space through which the solving process is guided during the second solution step.

If during the first step no solution is generated by initial propagation, the remaining search space has to be traversed further. This is done by selecting values from the valid intervals of variables that are not yet bound to one value. The consequences of such a selection are propagated through the constraint network. If a contradiction is detected, that is, the constraint satisfaction failed, the system will backtrack to the last choice point, and the next value of the unbound variable is chosen. Otherwise, a selection is made at the next choice point. This procedure is continued until a valid solution to the problem is found, that is, all variables are bound, or until the search space is completely traversed. After finding one solution, more solutions can be produced when backtracking and continuing search with other variable choices at choice points.

It is obvious that the sequence of unbound variables used as choice points in this second step and the way of selecting the values of the unbound variables are of great importance. An inadequate solving strategy will lead to a large number of backtracking operations that were preceded by wasted propagation effort and cost much solving time. Rather than attempting to find an optimal or near-to-optimal solution right away, one valid solution should be searched, which is then improved during the continuing solving process.

A solution to the process configuration problem is found when a consistent plan has been generated, that is, the components generated during search can be assigned a type, can be ordered in a sequence while no constraints (especially input–output relationships between succeeding processes) are violated. The predefined search strategy builds the process plan from both ends and adds new process components until the gaps in input–output restrictions between processes are filled.

The following example will find a process plan variant for the plan skeleton shown in Figures 5 to 7 with the following order data:

- order_material=C6031
- order_temper=60
- order_width=300

The “Warm Rolling” and “Cutting Edges” processes must be part of the process plan variant, since they are defined as mandatory, and there are no alternatives routes in the plan skeleton. With the given order data, the first two rules in the ramification decision table of the warm rolling node, shown in Figure 6, fail. The Else-rule will fire. As a result, processes on the left path are excluded from the successor set of the warm rolling component, and the right path is chosen. An “Annealing” component will be generated, since it is modeled as mandatory on this path.

Assuming that the existence decision table of the insert node “Cutting Lengthwise” determined the need for such a component in the process plan variant, this component will be added to the process plan variant according to the second rule. This is the first rule to fire when examining the decision table from left to right. The rule requires the insertion before the second "Cutting" process. At this point, only one "Cutting" process, the "Cutting Edges" component, exists in the solution. For a valid position to exist, a second "Cutting" component has to be generated. The only choice in this case is the instantiation of the "Cutting Widthwise" process at the predefined position. The "Cutting Lengthwise" component is then inserted directly before the "Cutting Edges" component.

The result of the solving process is shown in Figure 8. The processes outlined in bold borders are existing components in the solution.

4. PROCESS CONFIGURATION IN PRACTICE

4.1 Application context of the case study

This section describes the industrial setting of a case study for validating the process configuration concepts. The prototype implementation of the process configuration system and the case study evaluation were conducted at different rolling plants of an aluminum manufacturer. Production in the case study is based on a make-to-order concept. Each order is treated as a new product variant. A new process plan is thus generated for each order. Most orders differ in some order parameter detail, making reuse of a previously generated process plan impossible. About 10,000 different process plans presently exist, most of which were only manufactured once and are not expected to be manufactured a second time.

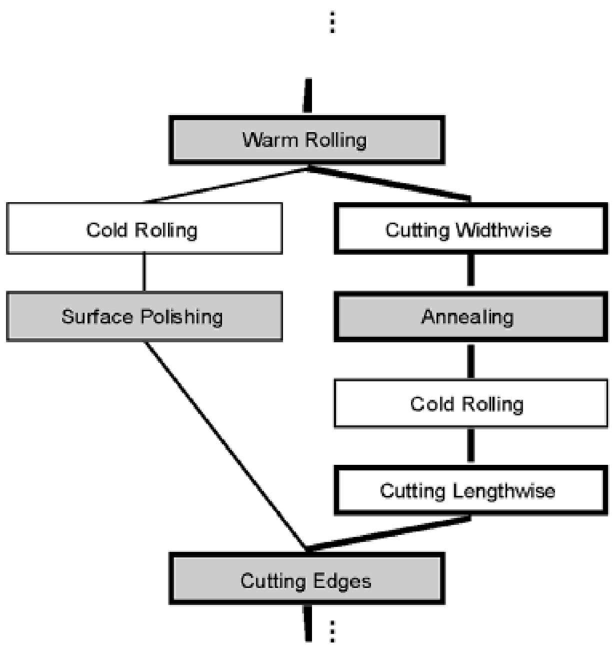


Fig. 8. Process plan variant solution of the example.

The company's main customers commonly place repetitive orders with equal order parameters. The vast majority of orders, however, are nonrepetitive. The focus of the process planning task is thus on supporting process plan generation for these orders. This task is automated to the greatest degree possible.

About 70 different manufacturing process types are available for the production of all types of rolls and sheets. Rolling processes, cutting processes, and heat treatment processes are relevant to all product groups. Specific treatment processes, such as washing or lacquering processes, are used only in certain product groups. A process plan consists of about 10 to 25 processes.

The existing process planning system was installed in the mid-1990s. It builds on a rule-based planning approach and is built on top of an expert system shell. Since its implementation, some deficiencies in the knowledge management of the process planning knowledge had become evident. The process configuration system is seen as a basis for new system generation that improves these shortcomings.

The rule base contains around 2,000 to 2,500 rules. These rules can be divided into three categories. Between 35 and 40% of the rules are existence rules that define the conditions under which a process is needed in a process plan. Another 35 to 40% of the rules are sequencing rules that determine the position of a process in the process plan relative to other processes and attempt to determine a predecessor or successor. The remaining rules are manufacturing instructions.

The planning systems works in a cyclical approach. In each cycle, processes of the process plan variant are selected, ordered, and then their process parameters calculated with the newest information gathered in the previous cycles. The planning system finishes when plan parameters stop changing between planning cycles. While the process plan structure is usually generated during the first cycles, a number of additional cycles are needed to optimize the process parameters. The planning task commonly requires between 5 and 15 planning cycles. The optimization part takes about 4 to 6 cycles.

Besides the attempt to reduce planning time, difficulties in knowledge management provided the main impetus to a rethinking of the selected approach. Maintaining the knowledge base is an extremely complex task. Rules, especially sequencing rules, are highly interrelated among each other such that changes in one rule may lead to unforeseen and unintended side effects at other places. Profound knowledge of the rule base is necessary to perform modifications.

Planners themselves often do not have sufficient knowledge of the rule-base structures to perform larger modifications. They usually maintain existence rules and manufacturing instructions. Sequence rules and calculation functions are, on the other hand, maintained by the knowledge engineers who designed the knowledge base. The original goal—to have the planners maintain all changing knowledge—could not be achieved by the planning system.

4.2. Implementation concept

Figure 9 shows an overview of the implementation concept for a process configuration system. The planner uses the plan skeleton editor to design, edit, and maintain the plan skeletons. The process configurator retrieves the product specification of a specific customer order from the company ERP software system and selects an appropriate plan skeleton for this order. Using the knowledge defined in the plan skeleton, it configures the process plan. The resulting plan is passed back to the ERP software system as input for further order processing tasks.

While the plan skeleton editor is a modeling tool for plan skeletons, the plan modification editor is a tool for specific process plan variants. The plan modification editor can be used for:

- visualization of a generated process plan variant,
- manually generating a (initial) process plan variant for a given customer order, based on a predefined plan skeleton, or
- modification of incompletely or unsuccessfully generated process plan variants.

The plan modification editor uses the same interface and a visual language very similar to the one in the plan skeleton editor. In contrast to the plan skeleton editor, it allows explicit manipulation of process selection and process parameter selection in a specific process plan variant. If desired, the changed process plan variant can be passed back to the process configurator for completion and a plausibility check. By using the plan modification editor, the level of interactivity during the execution of the process configuration task can be set on a problem-to-problem basis. A completely automated process configuration is possible as well as a completely manual generation of the process plan variant.

Figure 10 shows the workings of the process configurator in greater detail. The planner, who is the planning domain expert, defines a plan skeleton by developing a maximum process plan and identifying additional production and consistency rules that are associated to each of the manufacturing processes in the maximum process plan. One plan skeleton usually defines process configuration knowledge for one or part of one product family.

Whenever the process plan for a customer order is to be configured, the process configurator selects one plan skeleton from the plan skeleton library based on certain order parameters, such as the product family or possibly other key parameters. This plan skeleton is compiled into the process configuration constraint model. Taking this model together with the customer order data, the process configurator will attempt to find a process plan variant for the desired product specification in the model that defines a correct and technically feasible solution. If the product specification does not allow the correct configuration of a process plan, possible sources of errors are returned as feedback so that the user can identify mismatches easily. The planner can apply changes in the plan modification editor or restart the process configurator with changed input.

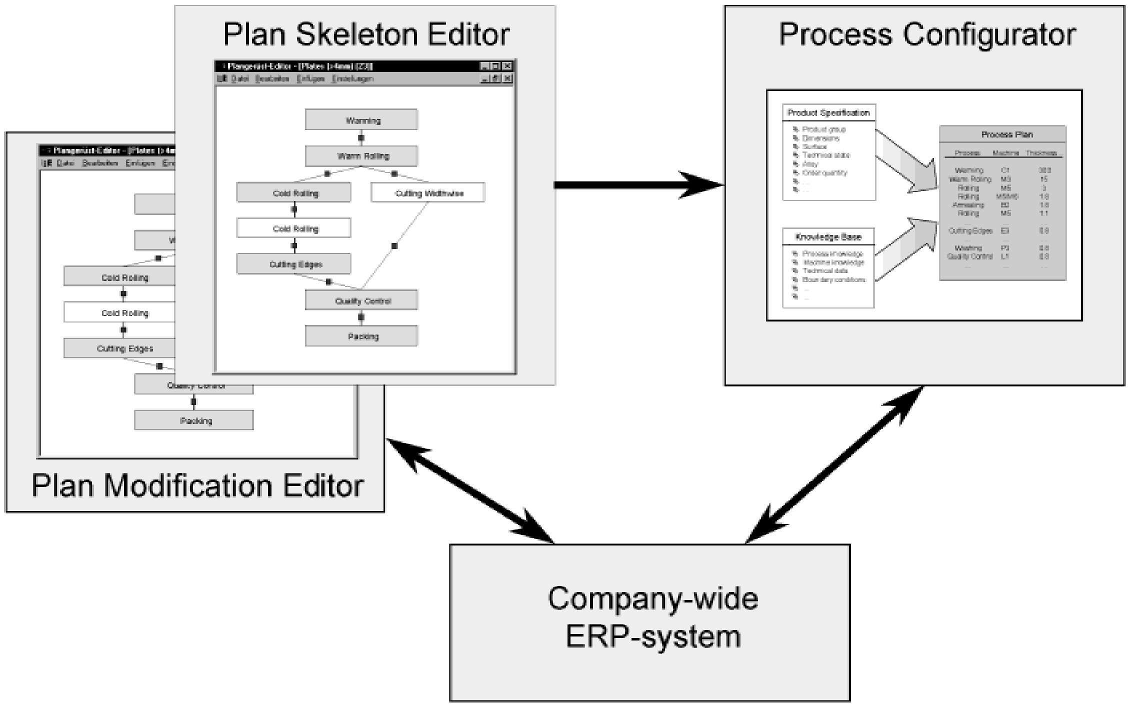


Fig. 9. Overview of the implementation concept for a process configuration system.

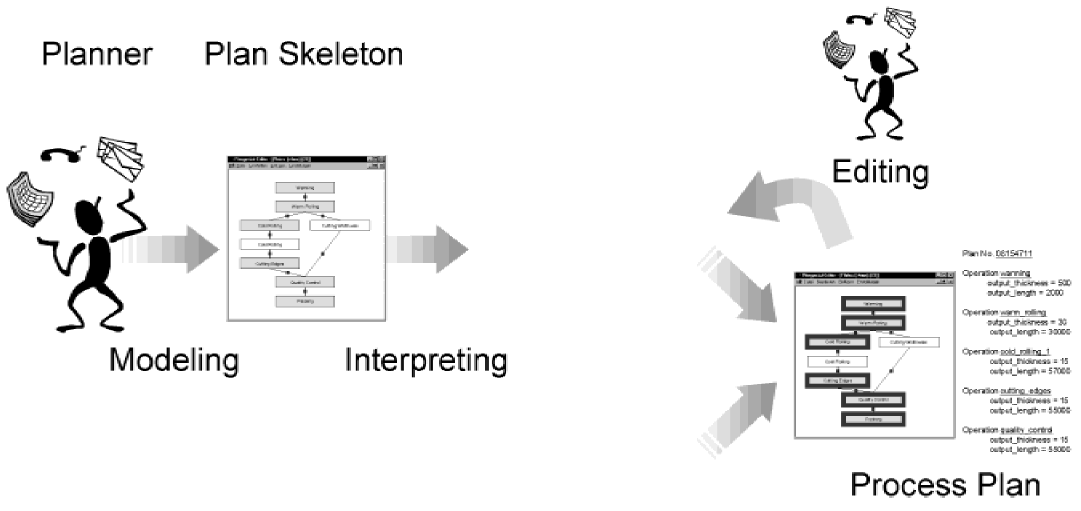


Fig. 10. The workings of the process configurator.

4.3. Case study results

The main goal in introducing the process configuration approach at the aluminum manufacturing company was to improve knowledge maintenance of process plan generation knowledge. Process plan quality improvement and process planning performance improvement were secondary aims. As the knowledge being modeled with the process configuration approach remains equal to the previous situation and also the plan optimization functions continue to be implemented in the expert system, there is no change in the quality of the process plans.

The sequencing task is solved correctly using the process configurator. While calculation times are higher than in the original expert system-based sequencing execution, they still remain in an absolutely acceptable single-digit second range. Increased calculation times are more than compensated for by the reduction of planning cycles needed to find a stable solution. Test runs showed that the number of planning cycles is reduced in the range between 15% and 50%, with an average reduction of 30%.

Much more important than performance improvements are simplifications and complexity reductions in knowledge modeling. These improve not only the first-time generation of plan skeletons, but also maintenance tasks. In the original expert system, the maintenance of calculation parameter tables and the maintenance of existence rules for process types were performed by the planners. Sequence rules were already too complex and most often had to be updated by knowledge engineers. Maintenance of parameter calculation functions as well as the introduction of new process types and their integration in the rule network were also tasks executed by the knowledge engineer.

Those responsibilities change with the introduction of the process configuration approach. Planners are now mostly able to perform sequence rule maintenance in the plan skeleton editor due to the improved modeling capabilities. The clarity of knowledge modeling was improved by the separation of sequence rules with respect to the related product groups, the implicit graphical representation of a large number of the sequence rules, and by relating the remaining rules to nodes in the plan skeleton. Certain syntax checks as part of the plan skeleton editor further reduce the risk of modeling errors and costly error search.

5. CONCLUSION

Process configuration was presented as a concept for solving the process planning task using principles known from the product configuration concept. Two main concepts for process configuration systems were presented: interactive process configuration and automation-based process configuration. Further explanations were given on how these concepts can be implemented in planning systems, and the concepts were implemented in a case study.

Users gave positive feedback on the use of visual means for representing process structures. Knowledge engineering in particular is greatly improved by making interdependencies between processes instantly visible. The use of similar interfaces for knowledge maintenance as well as for the visualization and manipulation of configured process plan variants is seen as a step forward as compared to the rule-based knowledge maintenance used previously. Tests in the integrated test case environment reveal improvements in plan generation time as well as in knowledge maintenance effort.

ACKNOWLEDGMENTS

The Project ProCONFIG (Process Configuration for Multiple-Variant Products: Work Plan Generation for Sheet and Coil Material Production) denotes the major contribution of the Swiss partners to the international IMS-Project GNOSIS—Knowledge Systematization: Configuration Systems for Design and Manufacturing. The Swiss Commission for Technology and Innovation supports the project within the IMS program through Fund No. KTI 3528.1.

REFERENCES

Alder, H. (1991). *Verteiltes Planen mittels selbstorganisierender Objektnetzwerke*. Dissertation, ETH Zürich.

Brown, D., & Chandrasekaran, B. (1989). *Design problem solving—Knowledge structures and control strategies*. London: Pitman.

Darr, T.P., & Birmingham, W.P. (2000). Part-selection triptych: A representation, problem properties and problem definition, and problem-solving method. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing 14(1)*, 39–51.

Erens, F.J. (1996). *The synthesis of variety*. Dissertation, Eindhoven University of Technology.

Hamelmann, S. (1996). *Systementwicklung zur Automatisierung der Arbeitsplanung*. Düsseldorf, Germany: VDI Verlag.

Klein, R., Buchheit, M., & Nutt, W. (1994). Configuration as model construction: The constructive problem solving approach. In: *Artificial Intelligence in Design '94*, (Gero, J.S. & Sudweeks, F., Eds.), pp. 201–218. Dordrecht, Netherlands: Kluwer.

Mailharro, D. (1998). A classification and constraint-based framework for configuration. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing 12(4)*, 383–397.

Mittal, S., & Falkenhainer, B. (1990). Dynamic constraint satisfaction problems. *Proc. Eighth Nat. Conf. on Artificial Intelligence*, pp. 25–32. Menlo Park, CA: AAAI Press.

Mittal, S., & Frayman, F. (1989). Towards a generic model of configuration tasks. *Proc. Eleventh Int. Joint Conf. on Artificial Intelligence*, pp. 1395–1401. Los Altos, CA: Morgan Kaufmann.

Sabin, D., & Weigel, R. (1998). Product configuration frameworks—A survey. *IEEE Intelligent Systems 13(4)*, 42–49.

Schierholt, K. (1998). Knowledge management aspects in process configuration for multiple-variant products. *Proc. First Int. Workshop on Intelligent Manufacturing Systems*, pp. 417–428. Lausanne, Switzerland: EPFL.

Schierholt, K. (2000). *Process configuration—Mastering knowledge-intensive planning tasks*. Dissertation, ETH Zürich.

Schlenoff, C., Knutilla, A., & Ray, S. (1996). *Unified process specification language: Requirements for modeling process*. Gaithersburg, TN: National Institute for Standards and Technology.

Schönsleben, P. (1985). *Flexibilität in der computergestützten Produktionsplanung und-Steuerung mit dem Computer* (1st ed.). Munich, Germany: CW-Publikationen.

Schönsleben, P. (2000). *Integral logistics management—Planning & control of comprehensive business processes*. Boca Raton, FL: St. Lucie Press.

Schwarze, S. (1996). *Configuration of multiple-variant products*. Zürich, Switzerland: VDF.

Stumptner, M., Friedrich, G.E., & Haselböck, A. (1998). Generative constraint-based configuration of large technical systems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 12(4)*, 307–320.

Karsten Schierholt was a researcher in the group of Logistics and Information Management at the Swiss Federal Institute of Technology (ETH) Zurich, where he also received his Ph.D. in Industrial Engineering and Management. His research interests are in the areas of information system support for intra- and interenterprise logistics processes, especially product and process configuration systems. He is currently at the SAP AG, Walldorf, Germany.