

Knowledge-Based Configuration – Survey and Future Directions –

Andreas Günter

Dep. Computer Science
University of Hamburg
guenter@informatik.uni-hamburg.de

Christian Kühn

DaimlerChrysler AG
FT2/EK, Stuttgart
christian.kuehn@daimlerchrysler.com

Abstract

The configuration of technical systems is one of the most successful application areas of knowledge-based systems. This article overviews and evaluates the developed representation technologies and problem-solving methods, successful application fields and software-tools for configuration of the last few years. Current research themes and perspectives for the application of knowledge-based systems are presented as an outlook for the future.

1 Introduction

The configuration of technical systems is one of the most successful application areas of knowledge-based systems. Configuration and construction are used in the following as a synonym. Thus, it must be noted that most construction tasks from the areas principle-, variant- and adaptive construction can be equated with configuration. An in-depth discussion about this can be found in [Günter 93b]. Following a general analysis of configuration problems, four central aspects (with respect to knowledge types) concerned with configuration were identified. Configuration tasks have the following characteristics:

- A set of **objects** in the application domain and their properties (parameters).
- A set of **relations** between the domain objects. Taxonomical and compositional relations are of particular importance for configuration.
- A **task specification** (configuration objectives) that specifies the demands a created configuration has to accomplish.
- **Control knowledge** about the configuration process.

Firstly, a few concepts are explained in order to further present and isolate the theme. Thus the concepts of design and configuration are initially differentiated. In the United States “design” is often discussed in publications when in fact according to our viewpoint, they are actually configuration problems. In [Tong et al. 92] design is defined as follows:

“What is design? Design is the process of constructing a description of an artifact that satisfies a (possibly informal) functional specification, meets certain performance criteria and resource limitations, is realizable in a given target technology, and satisfies criteria such as simplicity, testability, manufacturability, reusability etc.; the design process itself may also be subject to certain restrictions such as time, human power, cost etc.”

In addition to the general definition in [Tong et.al. 92] under *engineering design*, the project of "*physical artifacts or physical processes of various kinds*" is identified. A central aspect in this context is the creation of a structure from a functional requirement description. Design tasks are often identified as *ill-structured problems*. According to [Dörner 95] this means that no known direct mapping of functions on components exists. An additional problem often arises in the fact that this mapping is not clear, rather one component fulfils more functions or more components together provide one functionality (cp. [Ulrich et al. 92]).

Configuration is often described as routine design. This is based especially on the classification of synthesis tasks by Brown and Chandrasekaran [Brown et al. 89]. The following comments refer to this classification of synthesis tasks into three classes. Coming from the general model, the so-called routine design (design class 3) is described as a problem, where the specifications of objects, their properties and compositional structures are already given, and the discovery of a solution is based on a likewise known strategy. The task then is to assemble a configuration, which fulfils the task specification, from the known objects and according to a given strategy. An example here is the offer creation for technical systems on the basis of a complete description of components, their properties, a compositional structure and combination restrictions.

Since the named limitations are not given in the domain, then innovative design (cp. design class 2) or even further reaching creative design (cp. design class 1) can be spoken about. As a result the transitions between the different classes are smooth. In so far as a clear allocation is seldom possible, several graduations can be introduced between the routine design and the creative design. For this reason a range is assumed from routine design to creative design. An example of creative design is the artistic design of furniture.

In the following sections we discuss knowledge-based configuration methods, typical application areas, and tools for configuration. Current research themes bring this report to a close.

2 Knowledge-Based Configuration Methods

The first and for a long time most well-known knowledge-based configuration system was the R1/XCON [McDermott 82]. XCON was a rule-based system for the configuration of DEC computers. It was classed for a long time as a very successful system, however, this changed in the course of time mainly due to maintenance problems concerning the large knowledge base of XCON (see section about rule-based systems).

```
ASSIGN-POWER-SUPPLY-1

IF:
THE MOST CURRENT ACTIVE CONTEXT IS ASSIGNING A POWER SUPPLY
AND AN SBI MODULE OF ANY TYPE HAS BEEN PUT IN A CABINET
AND THE POSITION IT OCCUPIES IN THE CABINET IS KNOWN
AND THERE IS SPACE IN THE CABINET FOR A POWER SUPPLY AND THERE IS
NO AVAILABLE POWER SUPPLY
AND THE VOLTAGE AND FREQUENCY OF THE COMPONENTS IS KNOWN

THEN:
FIND A POWER SUPPLY OF THAT VOLTAGE AND FREQUENCY
AND ADD IT TO THE ORDER
```

Figure 1: A typical rule in XCON

Firstly, the most important concepts in the field of knowledge-based configuration, which are already applied, will be presented:

- Rule-based systems
- Concept hierarchies
- Structure-based approach
- Constraint-based systems
- Resource-based approach
- Case-based configuration
- Backtracking and variants of backtracking

2.1 Rule-Based Systems

The development of expert systems was occasionally strongly influenced by the rule-based paradigm. The restriction on the rule-based paradigm for expert systems was criticized on many occasions [Günter 91]. The critics referred to the following aspects: serious problems concerning knowledge acquisition, consistency checking, and in particular maintenance as well as lacking modularity and adaptability. Furthermore, it has led to the need to revise decisions concerning problems and the only unfavorable opportunities for integration of user instructions and case-based approaches.

Above deficits appeared when rules were applied as the sole formalism to represent knowledge. Those opposing our opinion concerning rules have protected their possibilities by representing and evaluating heuristic relations which is where the strengths of rules lie. The main problem is that there was an overloaded use of rules in the course of expert system research and they were utilized as uniform knowledge representation formalisms.

Consequently, rules can be used for the representation of local connections in the configuration process and also for the specification of cycles. However, the combination of both types of knowledge often leads to the mentioned problems. Diverse tools are available for rule-based systems. But configuration tasks can hardly be realized adequately with these tools alone. Using tools, only a small part of the necessary knowledge can be represented, consequently they should be used only as a method among others in a configuration framework.

2.2 Concept Hierarchies

There is a need for the representation of object knowledge in almost all configuration systems. An object-oriented concept-based representation is commonly chosen as a representation form. Such an object-centralized representation makes the bundled specification of properties and their potential value ranges possible. Thus so-called facets for the specification of application knowledge are used (for example, default values). In an experiment about methods used in configuration expert systems, Linnemann came to the conclusion that frame-like representation technologies for object presentation are used in 9 out of 10 systems [Linnemann 94].

In a predominant number of configuration systems, object descriptions are classified in *taxonomical hierarchies* (is-a-relation). Using this method of abstraction, knowledge can be structured, generic descriptions of objects can be formed (which correspond to object classes), and knowledge can be represented more efficiently and with less data redundancy using inheritance mechanisms.

With the configuration process, the compositional structure of an object according to the has-parts-relation is of great importance. It is still only this structure which is included in the solution, it links together an aggregate and its components. The idea of skeleton plans (s. [Friedland 79]) is often used in technical applications of knowledge-based systems, for example in the project ARC-TEC [Richter et al. 91]. This technique is based on compositional hierarchies which are represented here in the form of skeleton plans. The skeleton plans allow compositional relations to be represented, for example the has-parts-relations in the PLAKON-System [Cunis et al. 91] or in KONWERK [Günter 95a]. The so-called structure-based approach which is based on explicit representation of component structure is explained in one of the following sections.

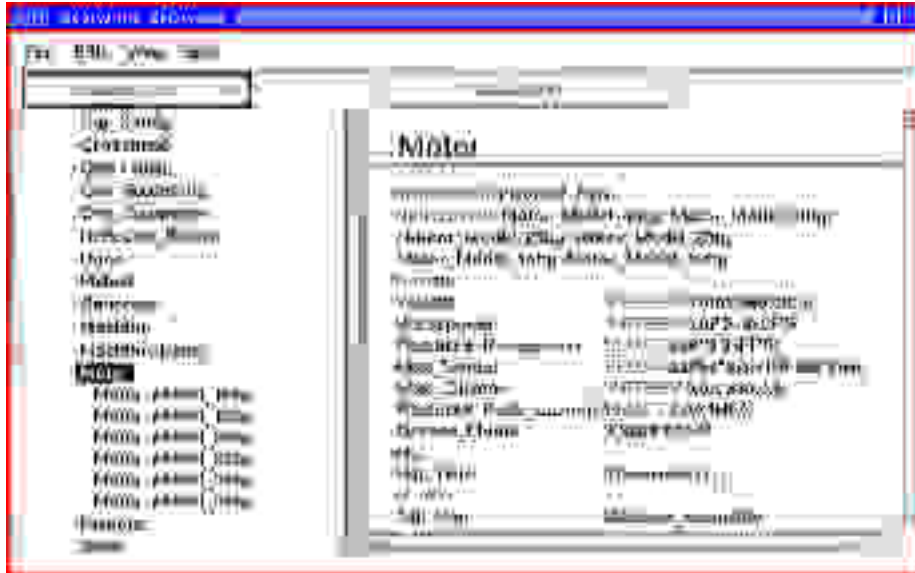


Figure 2: Example of a concept definition

2.3 Structure-Based Approach

In [Günter 91] different so-called control types are introduced. One of these is the *structure-based approach*. The compositional, hierarchical structure of an object serves thus as a guideline for the control of problem solution. Through this, a top-down-concept can be realized which orientates itself on the component structure. The control type „structure-based“ can be found in many concepts and systems (also in planning systems):

- Specification of a task in the form of AND/OR-trees.
- Skeleton construction (according to [Puppe 90], derived from Friedland's skeleton plans).
- Hierarchical structured rule contexts (for example, XCON [McDermott 82]).
- Configuration tools PLAKON and KONWERK.
- Direct or indirect in a predominant number of application systems.

Therefore, structural information has two meanings (s. [Günter 91]):

- The structure of a solution is predefined by the conceptual hierarchy, it consists of hierarchically ordered components.
- Tasks can be divided into subtasks according to their structure.

The skeleton plans from Friedland [Friedland 79] are often chosen in technical domains as a starting point for the strategy (s. [Richter et al. 94; Richter et al. 91]),

they allow a top-down-approach which is orientated to the component structure. The *concept hierarchy orientated control* of PLAKON [Günter 91] and KONWERK [Günter 95a] is based on the specification of component structure in the form of has-parts-relations in the concept hierarchy. A structure-based approach is supported (partly implicitly) in a predominant number of commercial tools for offer creation and configuration support. This is adequate and also often familiar to the user since it can be mapped on a product model relatively easily.

2.4 Constraint-Based Systems

Restrictions between objects can be represented and evaluated with the help of constraints. Using a constraint, a relationship between objects and their properties can be specified in a knowledge base and can be evaluated by constraint propagation.

Numerous constraint systems and also commercially available tools have been developed in the meantime. These constraint systems are however concerned with the aspect of the so-called constraint satisfaction, which is the efficient evaluation of a given constraint network with a given value allocation. In contrast to other application areas of constraint systems, the following demands arise from configuration tasks:

- Constraint networks need to be accumulated incrementally in line with the further development of partial solutions. At a given time, the complete number of the constraints to be considered is not known (except in the final solution).
- Constraints must be formulated in such a manner that they either require, forbid or limit the number of new objects.
- Symbolic, functional and numerical constraints need to be available and processed in a complete system.
- Often a complete evaluation of constraint-relations is not possible due to the magnitude of the search space. Therefore, only the upper or lower limits can be propagated but not each individual value because of complexity reasons concerning the range of parameter (property) values.



Figure 3: Example of a simple constraint net

Because of these properties, further reaching technologies are currently being developed (or have already been) for the usage of constraints in configuration systems (cf. [Cunis et al. 91; Fleischanderl et al. 98; GÜSGEN et al. 92]). Also the „Dynamic CSP“ approach (see also [Stumptner 97]) includes a few of the specified

enhancements, in particular the dynamic structure of the constraint network. A commercial tool which fulfils these high requirements is not yet known, however in universities diverse prototypes have been successfully realized and used.

2.5 Resource-Based Approach

Resource-based configuration is based on the following elementary principle: components of a technical system are used because they offer a service which is required by the system as a whole or because other components of the system need this service (cp. [Böhm et al. 96; Emde et al. 97; Heinrich 93; Heinrich et al. 93; Jüngst et al. 98; Neumann et al. 89]). Interfaces between components are specified by the exchanged *resources*. The same is true for the relation between system and surroundings. To these resources belong technical resources such as power consumption and memory capacity in addition to mercantilistic resources such as price and maintenance effort. Components make a number of resources available and also consume resources themselves. A task specification exists in that the concrete surroundings of the chosen system are resource-based specified as components. In an interactive configuration process, the resource deficits are recognized and are thus equated by the instantiation of components (balancing methods).

2.6 Case-Based Technologies

Case-based problem-solving methods are therefore identified, that knowledge about already solved tasks is saved and is used for the solution of new tasks. The methods for selecting case knowledge from a case base can be divided into two groups, those which carry out a *preprocessing* of the case base and those which do not but work on an unstructured number of cases. Using the case knowledge which is used to support configuration, it must not always be concerned with a *complete* case. The takeover of parts is also possible. The reason being is the assumption that similar tasks lead to similar solutions. An alternative basic approach is the comparison of current tasks to already known problem solutions in the application area [Pfitzner 93].

Fundamentally two types of approaches to determine how case knowledge can be selected are:

- Retrieval and, when suitable, adaptation of the solution (transformational analogy) or
- Selection of case knowledge in individual steps. This case knowledge can also act as control knowledge (derivational analogy).

Pfitzner [Pfitzner 93] differentiates respectively between different transfer modes. Retrieval and adaptation of case knowledge however always remain separate procedures. A range of systems and applications have already been case-based realized (cp. [Bartsch-Spörl 87; Hua et al. 93; Olivier et al. 96; Paulokat et al. 93; Weiß 92]).

However, case-based technologies also have a few disadvantages:

- They typically provide *conservative*, that is no – or only few – innovative solution suggestions.
- No – or only narrow – causal explanations exist for the suggested solutions.
- Case-based methods often lead to relatively *bad* solutions

2.7 Resolving of Configuration Conflicts

In general it is true that heuristical decisions are necessary for many applications due to complexity and partly due to absent knowledge (cf. [Günter 93a]). These decisions imply among other things that:

- an optimal solution is not necessarily found and
- the decisions prove to be disadvantageous during the course of further problem-solving or can even be incompatible with the rest of the solution parts.

In this situation, mechanisms for further work are necessary. Independent of the reason there are two alternative methods for conflicts:

- *Revision of decisions*

The revision of configuration steps is also known as *backtracking*. Here one or more of the previously met decisions are taken back and are performed again with another value.

- *Repair of a partial solution*

Instead of revising decisions, it is sometimes also possible to “repair” a partial solution using a direct modification and/or additional domain objects and thus to retransform it into a consistent state. Using such a repair method, decisions are changed likewise, however no reference to configuration history and the previous successful steps is given.

The following variations concerning the retraction of configuration decisions have been developed (see [Ginsberg 93; Günter 93a]):

- Chronological backtracking
- Backtracking over more levels
- Dependency-directed backtracking
- Knowledge-based backtracking
- Interactive backtracking
- Backtracking with data adoption

3 Applications and Transfer of Technology

Configuration with knowledge-based techniques is a successful application field for methods of the artificial intelligence area (AI). There are two application scenarios where configuration methods are principally employed (see [Brinkop et.al. 94; Haag 98; Tank 96]):

- development and construction as well as
- product configuration and feasibility analysis (offer creation and sales)

Knowledge-based configuration systems have been developed among other things for: computer systems, elevators, drive controllers, rolling mills, industrial stirring devices, passenger vehicles, image processing systems, machine parts, communication systems, pumps, trucks, switchgears, conveyor systems, SPS controls, lighting systems, manufacturing cells, aircraft vehicle cabins, liquid crystals etc.

General objectives for the development of application systems are:

- reduction costs for development and making sales offers,
- lowering of error rate,
- faster offer creation,
- successful duplication of solved applications,
- support for the quick change of products and product generations,
- general backup of the company's individual knowledge (i.e. documentation) and
- standardization of solutions.

There is no patent recipe for the transfer of research results in applications, this is valid for configuration tasks as well. The development of an integrated software system with a knowledge-based configuration system as an essential component requires the application of „classic“ software engineering principles, which we will not go into here. In addition, individual process models exist for knowledge-based systems, which are mainly based on the two fundamentally different statements of prototype-based and model-based methods. A multitude of process models for the development of expert systems try to combine both approaches.

The following steps are not necessary in every application area, need not be arranged in a strict sequence, and may include feedback. Nevertheless this appears to us as a meaningful structuring method (see [Günter et al. 98]):

- Analysis of the application problem
- Definition of a pilot study or example
- Knowledge acquisition
- Feasibility investigation
- Economical analysis
- Tool selection
- Demonstration prototype
- Specification and realization
- Step-by-step implementation
- Integration

Due to our experiences we would like to comment as follows on the above points:

- *Quick analysis*
Already in the first discussion an initial classification of the problem must be made and statements to the solution possibilities must be given. At this point it is clear to both sides that nothing obligatory has to be said, the first impression and reaction to it is important. This requires experience in the solution of configuration problems and of the analysis of applications. Computer scientists have frequently the task of solving „other people’s“ problems. This demands communication abilities and a service awareness. Both hardly play a role in the university education of computer scientists.
- *Technical possibilities*
Not everything which is technically possible should necessarily be realized. What is realized should orientate itself around economic considerations, the business environment and the integration strategies.
- *Tools vs. individual software*
Individual software is cost-intensive, both in development and maintenance. If possible, software tools should be used. This is not even a new discovery but it is hardly ever considered.
- *Meaning of (university) prototypes*
We have realized same applications with KONWERK [Günter 95b], while it was clear that this would find no entrance in the application system. Nevertheless such a prototype represents a „convincing“ help in supporting arguments. This is also valid for our opinion when the prototype is based on a university tool. Software tools are a decision aid for the transfer of technology (cp. [Neumann 95]).
- *Early inclusion of the user*
What do users expect in the future? A question which is often asked too late. An early discussion with the user improves the system and in particular the acceptance of it.
- *Software-ergonomical considerations*
The usability instead of the functionality is in the foreground of ergonomics. Also through the inclusion of the user in the early phases of system development, the requirements can be in principle better fulfilled.
- *Integration*
Configuration systems are used in a context. A link to data bases, CAS systems or internet/intranet is often necessary and must be already included in the run up to system development.

4 Software Tools

Computer support using knowledge-based configuration can be identified for various tasks, especially for product development, offer creation and technical feasibility analysis.

There are several available tools on the market for supporting the configuration of variety-rich products for offer creation (e.g. CAS, Cosmos, et-epos, SalesPlus, SCE, Secon, Sellor, SC-Config). On the other hand only university tools have been developed for complex tasks, which lie in the field of supporting engineering tasks.

The tasks which can be carried out using available CAS tools range from price calculations (including the evaluation of rules for discounts and mark-ups), consistency checking, generating bills of materials, and the creation of written offers on the basis of predefined text modules. Here the following technologies are used: object-oriented representation of configuration objects with properties and relations (partly in connection with a database), rules, and user-driven configuration. The application of technologies such as constraints, heuristics, resource-based methods, case-based methods, modeling of complex requirements, and functional modeling, however is only now realized in individual applications, or rather supported through prototype software tools.

A university tool system for configuration is KONWERK. A fundamental assumption connected with the development of KONWERK was that the efficient, economic employment of knowledge-based methods can only be successful if a domain independent tool can be made available and can support this problem-specific aspects of configuration problems. KONWERK (cp. [Günter 95b]) was founded as a problem class specific framework system in which configuration methods were integrated and made available.

A central requirement for tool systems is that suitable concepts for different domains must be given and that also no unnecessary complex mechanisms are made available (i.e. all necessary mechanisms, but only these). Our solution provides here a sub-section in problem-solving modules which are built on each other. KONWERK was realized as a domain-independent module-based construction set which makes modules available for the following aspects:

- Representation of domain objects and their properties,
- Representation and processing of relations and restrictions,
- Formulization of objectives specifications (task formulation) and
- Control of the configuration process.

Therefore both standard mechanisms (basic modules) as well as conceptual further-reaching methods in the form of enhancement modules are available in a complete framework system.

Structuring in the problem-solving modules of KONWERK lies on a technical level, that is, not on a level of human solving. The structuring as a module-based construction set allows a flexible combination of modules for the applications. The user-necessary functionality is made available through the “configuration” of the modules to an application system.

KONWERK allows several applications from different domains to be realized:

- Selection and arrangement of friction bearings
- Configuration and dimensioning of jointed shafts
- Design of liquid crystals
- Configuration of passenger cabins in vehicle aircrafts
- Configuration of hydro-geological models
- Configuration of elevators (VT-domain)
- Modeling of bills of materials in automobile production
- Arrangement and dimensioning of drive control systems

Due to the modeling of real domains, it has been shown that the concepts are also suitable under application-based aspects. There the requirements were based both on KONWERK and on the known deficits of preceding prototype production with help of PLAKON [Cunis et.al. 91]. The heterogeneity of the considered applications, which among other things include several of the specified aspects, ensure the usability of KONWERK for a wide range of application problems. With the university tool KONWERK applications can be adequately realized for the prototyp (cp. [Neumann 95; Richter 96]).

5 Current Research Themes

In the presentation of current research themes, the following methods and application problems will be considered:

- Model-based configuration and simulation
- Truth maintenance systems (TMS) and assumption-based TMS
- Description logics
- Extensions of constraint techniques
- Functional requirements and specification through sketching
- Explorative configuration
- Integration in a product model and workflow
- Configuration of software and embedded systems

5.1 Model-Based Approaches

The concept “model-based” is used here in analogy to model-based diagnosis. Such methods are based on a qualitative representation of technical laws, general principles, and knowledge about functions of technical devices and processes (cp. [Früchtenicht 88; Günter et.al. 97; Stein 95; Stein et al 96]). The aim is, on the basis of this general knowledge, to design (new) technical systems. This includes also tasks from creative design. An important argument for model-based methods is that technical systems are often so complex that they offer the possibility to work with a simplified (qualitative) model.

5.2 Truth Maintenance Systems (TMS) and Assumption-Based TMS

The employment of *Truth Maintenance Systems* (cp. [Doyle 79]) represents a promising possibility in improving construction systems. There are several systems in which the employment of TMS has been discussed and partly implemented (cp. [Cunis et al. 91; Paulokat 92; Schädler 95]). The motivations for application TMS are to support knowledge-based backtracking (possibilities to data adoption) and to improve explanations concerning design decisions. The most important functionality of TMS is that in the case of revised decisions, all the dependent consequent decisions are identified and are revised where appropriate.

One possibility of configuration strategy can be developing and managing all partial solutions during problem-solving. Several concepts exist here for configuration which are based on the application of ATMS (*Assumption-Based Truth Maintenance Systems*, introduced by [DeKleer 86]) (see [Bräuer 91; Haag 95; Hein 91; Paulokat 92; Schirp 96]). Inconsistent part solutions are thus accordingly marked and are no longer considered later on. This method manages a breadth-first search. For configuration it has to be stated that none of these systems have so far left the prototype stage. Indeed relatively simple configuration tasks can be solved using this method.

5.3 Description Logics

Taxonomical hierarchies in the AI have already been investigated in-depth and have culminated in the conception of terminological systems as an application of description logics. Description logics are based on formal semantics which allow certain consistence predictions on a knowledge base and include also interesting inference procedures (e.g. classification). Description logics are expressive, include fundamentally investigated inferences, are formally founded, and form a quasi-standard in the theoretical AI; these arguments highlight that they are to be included also in configuration systems. Up to now this has not been possible without problems especially because configuration-specific requirements have only been partially considered (see [Baader et.al. 96; Baader et.al. 98, McGuinness et.al. 98; Wright et.al. 93]).

5.4 Extension of Constraint Techniques

In order to enhance the constraint techniques research work is being done at this time on different problems. Lots of the work deal with the enhancement of the Constraint Satisfaction Problems (CSPs) to allow the possibility to dynamically enlarge these amounts, not only for fixed, enumerated sets of variables.

An extension of the CSPs are the dynamic CSPs (DCSPs), which differentiate between meta-level constraints and compatibility constraints. While compatibility constraints have no influence on the activities of the variables, variable activation can be directly achieved or rather prevented by the meta-level constraints. An introduction to DCSP is given in [Stumptner 97], for example.

A further topic that raises a lot of questions is the constraint relaxation, i.e. the fading out of constraints in over-constrained conflict situations. For this, domain-dependent knowledge about the relaxability of the constraints is necessary. The revision of constraints often results in too many constraints (in the worst case all) having to be withdrawn. An idea exists that the same restriction of one domain is modeled through several constraints of different relaxability (as constraint bundles), so that the conflict can be solved locally through the successive cancelling of individual constraints from the bundle (cp. [Cunis et al. 91]).

A further statement which also exists in connection with constraint relaxation is the introduction of fuzzy constraints, in that the value combinations, which are permitted by a constraint, understand the parameters as a quantity, to which a membership measurement can be defined. In [Schumann et.al. 95] an evaluation of fuzzy constraint relations exists through a projection on fixed constraints.

Further research work deals with the efficient evaluation of constraints. This is for example possible through the definition of domain-specific evaluation strategies, when for instance, there is a given sequence of constraints or of variable mapping (cp. [Buchheit et.al. 94; Stumptner 97]). An efficiency improvement of the constraint propagation can be achieved by the parallelization and distribution over several workstations. The difficulty here lies in the high degree of dependencies in constraint networks and therefore finding the most independent sub-network (see [Hotz 96]). A further unsolved problem are relation with temporal restrictions, which for example are necessary for the configuration of reactive systems like control systems.

5.5 Functional Requirements and Sketches

Up to now the specification of a task formulation has taken place in configuration systems, mostly using the direct selection of the solution features. An important and interesting addition are the opportunities of giving functional requirements to configuration using optimization objectives. This means that the necessary functionalities are formulated as part of the task formulation and a configuration system maps this on the objects and parameters which are necessary for it. So-called sketch functions are a further addition. Using a sketch function a user should be in the position to produce a task formulation as a sketch. The sketch is therefore based on application-specific by which graphically describe a task formulation. An example is the application-specific user-interface of the system art deco [Stein 95]. Both requirements are particularly valid for the employment of configuration technologies in internet-based applications. An internet configuration system must be in the position to make expert knowledge transparent to the user and represent it graphically. The users will give here only under-specified requirements and therefore focus very strongly on pure functional and price aspects. It is necessary to map functional requirements on the concrete solution level.

5.6 Explorative Configuration

An explorative strategy is necessary in many complex application domains. Here the following two points are of particular importance (see [Navinchandra 91]):

- the *redesign* of partial solutions and
- exploration into “new” areas.

In simple design tasks, the changing of partial solutions is avoided where possible using skilful strategies. In opposition to this, the alteration of a partial solution within design tasks is a natural procedure also for the constructor. A central requirement for design and construction tasks is consequently the opportunity of (interactively) changing the partial solutions. A possible procedure is the application of so-called repair knowledge (see [Günter 93a; Kühn 95; Marcus et.al. 88]). This was implemented, for example, into the system VT for configuration of elevators. One important feature is the direct modification (redesign) of partial solutions without reference to “development history” (which is always considered using backtracking).

5.7 From Configuration to Design

Whereas for the routine configuration field, a multitude of successful technologies have already been developed and partly implemented, there is a need for research work on the automation of designs in ill-structured domains¹. For such domains, which hardly allow used structures to be recognized for configuration, the methods of routine configuration are no longer sufficient [Dörner 95]. Instead new technologies are required, which belong to areas of innovative and creative design (cp. design classes 2 and 1 in [Brown et.al. 89]). Special mechanisms are required, which can handle incomplete, uncertain, and fuzzy knowledge.

We are, however, of the opinion, that also a complex design process can be partly supported by the integration of configuration methods, so that the human designer is relieved of routine application work. Such an approach is followed for example by [Faltings et.al. 93] for the design of mechanical systems. For design support, Faltings and Sun have used technologies from qualitative physics.

The previously mentioned approaches – model-based techniques, functional modeling, and explorative configuration – principally also belong to the field of knowledge-based design. These techniques should not be considered as competitive techniques, but rather as methods that give the ability to supplement one another.

¹ Strictly speaking ill-structured domains are thought of as domains, for which only ill-structured models are known [Dörner 95].

5.8 Integration in a Product Model and Workflow

The integration of different methods is especially important for the successful transfer of technology. Although great success has been achieved in the research field of knowledge-based systems, the transfer of such success depends heavily on the practical integration of combined management with other technologies. A constructive cooperation is especially necessary with technologies from the areas of databases, graphical engineering systems (CAD), software ergonomics, distributed system architectures, optimization, simulation of systems, as well as validation and verification. A further level of integration is found in the application field, where different disciplines like electrical engineering, mechanical engineering, and economics in addition to information technology often take part.

Furthermore, integration plays an important role in management cycles. Due to the fact that it does not always suffice to deal with the configuration tasks individually, it is often necessary to view the complete product process chain because certain processes and strategies have effects on other parts of the process chain. The advantages of a continuous configuration concept which is embedded in the workflow of a company begin with improved documentation and end with clear improvements of compatibility and tuning between individual steps in the process chain.

Consequently, continuity is required also for the creation and maintenance of a knowledge base. This should be integrated together with component libraries in a complete product data model.

A further, more important aspect is the operative introduction of knowledge-based configuration technologies. Here the strategies are to be worked out and the acceptance of introduced systems is to be considered (cp. [Günter et.al. 98]). In a VDI-guideline, the three “dimensions” of integration are considered: technical, organizational, and social integration [VDI 93].

5.9 Configuration of Software and Embedded Systems

The configuration of software or rather of embedded systems can prove to be a very complex task, in that it concerns realizing functional and dynamic concept aims in a software-based system. Thus requirements of the produced system include efficiency, correctness and reliability. These requirements are particularly important in safety-critical applications where embedded systems are frequently used. The employment of software can lead to the ability to set parameters to a higher grade as well as more frequent modification and adjustment. Software in an embedded system has to interact directly with electronic components (sensors and actuators) and – in case of mechatronic systems – indirectly with mechanical components. Thus embedded systems often require the production of software for hardware platforms which do not yet exist.

The knowledge-based configuration of software does not deal in this context with the free, creative development of software systems or rather products, but rather deals with choosing and assembling software components, parameter setting and allocate

suitable hardware. Thus a complete method is to be developed which takes into account the following sub-problems:

- How can software suitable be modeled for configuration. That is, which structure criterions exist, how can software components be adequately modeled with their interfaces and what are the most important concept properties for configuration? How can the dependency between software and hardware structures be modeled?
- In which way can requirements on system behavior be formally specified (safety requirements, functional and dynamic requirements)?
- How can heuristical, domain-dependent knowledge for drawing conclusions from specification features to solution features be adequately modeled, especially on software structures and properties?
- Integration of validation and verification mechanisms in the configuration process: Which technologies are suitable for guaranteeing safety requirements, how can a suitable cooperation between synthesis steps and analytical validation and verification methods take place?

A possible application example is the configuration of vehicle electronic systems consisting of control units which are networked with each other using a CAN-Bus system as well as with sensors and actuators. The tendency to realize even more functionality through software leads to higher requirements in the development of electronic systems. Thus special control units employed for various different functions (for example, like control units for engine, gears, break etc.) could be replaced in the future by universal control units which are independently programmable, can carry out more sub-functionalities at the same time and later allow themselves to be modified (redesign).

6. Summary

Knowledge-based configuration and construction methods have been developed in the last few years “for the market”, included in applications, and implemented into software tools. Their main advantages are improved maintainability, shorter development time, and problem functionality.

Although several software-tools (computer-aided selling) are available on the market for the production of offers, procedures for the technical configuration of complex products (both in development as well as for technical feasibility analysis and product adjustment) are not yet a component of marketable tools. Concerned tools have been provided up to now almost exclusively by university tools.

References

- Baader, F., Sattler, U. *Description Logics with Symbolic Number Restrictions*. in: Proc. 12th European Conf. on Artificial Intelligence, Budapest 1996, p. 283-287
- Baader, F., Sattler, U. *Description Logics with Concrete Domains and Aggregation*. in: Proc. 13th European Conf. on Artificial Intelligence, Brighton, UK, 1998, S 336-340
- Bäckström, C., Jonsson, P. *Planning with Abstraction Hierarchies can be Exponentially Less Efficient*. in: Proc. 14th Int. Joint Conf. on Artificial Intelligence, Montréal, 1995, p.1599-1605
- Bartsch-Spörl, B. *Ansätze zur Behandlung von fallorientiertem Erfahrungswissen in Expertensystemen*. in: KI 4/87, Oldenbourg-Verlag, p.32-36, 1987
- Böhm, A., Uellner, S. *Kundenspezifische Konfiguration von Telekommunikationssystemen*. in: J. Sauer, A. Günter, J. Hertzberg (Hrsg) 10. Workshop Planen und Konfigurieren, infix Verlag, p. 149-159, 1996
- Brinkop, A., Laudwein, N., Maassen, R. *Routine Design for Mechanical Engineering*. in: 6th Innovative Application of AI Conference (IAAI), p. 3-13, Seattle, 1994
- Brown, D.C., Chandrasekaran, B. *Design Problem Solving* Pitman, Research Notes in AI, 1989
- Buchheit, M., Klein, R., Nutt, W. *Configuration as Model Construction: The Constructive Problem Solving Approach*. in: J. Gero & F. Sudweeks (Hrsg.), Artificial Intelligence in Design, Kluwer Academic Press, 1994
- Cunis, R., Günter, A., Strecker, H. (Hrsg.) *Das PLAKON Buch - Ein Expertensystemkern für Planungs- und Konfigurationaufgaben in technischen Domänen*. Springer, 1991
- DeKleer, J. *An Assumption-Based TMS*. in: Artificial Intelligence Journal, Vol. 32, p. 231-272, 1986
- Dörner, H. *Konfigurieren in schwach strukturierten Domänen*. in: [Günter 95b], p. 33-38
- Doyle, J.A. *A Truth Maintenance System*. in: Artificial Intelligence Journal, Vol. 12, p. 231-272, 1979
- Emde, W. et al. *Interactive Configuration in KIKon*. in: P. Mertens, H. Voss (Hrsg.) Wissensbasierte Systeme '97, infix Verlag, p. 79-92, 1997
- Faltings, B., Sun, K. *Computer-Aided Creative Mechanism Design*. in: Proc. IJCAI-93, p. 1451-1457, 1993
- Fleischanderl, G. et al. *Configuring Large Systems using Generative Constraint Satisfaction*. in: IEEE Intelligent Systems Jul/Aug 98, p. 59-68, 1998
- Friedland, P.E. *Knowledge-Based Experiment Design in Molecular Genetics*. Report STAN-CS-79-771, Stanford University, 1979
- Früchtenicht, H.-W. (Hrsg.) *Technische Expertensysteme: Wissensrepräsentation und Schlußfolgerungsverfahren*. Oldenbourg, 1988
- Ginsberg, M.L. *Dynamic Backtracking*. in: Journal of Artificial Intelligence Research, Vol. 1, p. 25-46, 1993
- Güsgen, H.-W., Hertzberg, J. *A Perspective of Constraint-Based Reasoning*. Springer, 1992
- Günter, A. *Flexible Kontrolle in Expertensystemen für Planungs- und Konfigurationaufgaben in technischen Domänen* Dissertation, Universität Hamburg, 1991
- Günter, A. *Verfahren zur Auflösung von Konfigurationskonflikten in Expertensystemen*. in: KI 1/93, p. 16-22, 1993a
- Günter, A. *Modelle beim Konfigurieren*. in: O. Herzog et al. (Hrsg.) KI-93, Springer Verlag, p. 169-176, 1993b

- Günter, A. *KONWERK - ein modulares Konfigurierungswerkzeug*. in: F. Maurer & M. M. Richter (Hrsg.) *Expertensysteme 95*, Kaiserslautern, infix Verlag, p. 1-18, 1995a
- Günter, A. (Hrsg.) *Wissensbasiertes Konfigurieren – Ergebnisse aus dem Projekt PROKON*. infix Verlag, St. Augustin, 1995b
- Günter, A., Kühn, C. *Einsatz der Simulation zur Unterstützung der Konfigurierung von technischen Systemen*. in: Mertens, Voss (Hrsg.) *Expertensysteme '97*, infix Verlag, p. 93-106, 1997
- Günter, A., Kühn, C. *Erfahrungen beim Transfer von Konfigurierungsmethoden*. in: J. Sauer & B. Stein (Hrsg.) *12. Workshop Planen und Konfigurieren*, Paderborn, Bericht Universität-GH Paderborn, p. 59-64, 1998
- Haag, A. *The ATMS* – An Assumption Based Problem Solving Architecture Utilizing Specialization Relations*. Dissertation, Universität Kaiserslautern, 1995
- Haag, A. *Sales Configuration in Business Processes*. in : IEEE Intelligent Systems Jul/Aug 98, p. 78-85, 1998
- Hein, M. *Effizientes Lösen von Konfigurierungsaufgaben*. Dissertation, TU Berlin, 1991.
- Heinrich, M. *Ressourcenorientiertes Konfigurieren*. *Künstliche Intelligenz*, 7 (1), p.11-15, 1993
- Heinrich, M., Jüngst, E.-W. *Konfigurieren technischer Einrichtungen ausgehend von den Komponenten des technischen Prozesses: Prinzip und erste Erfahrungen*. in: F. Puppe, A. Günter (Hrsg.), *Expertensysteme 93*, p. 98-111, Springer Verlag, 1993.
- Hotz, L. *Überlegungen zur parallelen Verarbeitung in Konfigurierungssystemen*. in: J. Sauer, A. Günter, J. Hertzberg (Hrsg.) *Beiträge zum 10. Workshop „Planen und Konfigurieren*, Bonn, Infix, p. 172-178, 1996
- Hua, K., Smith, I., Faltings, B. *Integrated Case-Based Building Design*. in: Proc. EWCBR '93, Springer, p. 436 - 445, 1993
- Jüngst, W.E., Heinrich, M. *Using Resource Balacing to Configure Modular Systems*. in: IEEE Intelligent Systems Jul/Aug 98, p. 50-58, 1998
- Kühn, C. *Konfigurierung von Fahrstühlen mit VT/KONWERK*. in: [Günter 95b], p. 327 - 336
- Kühn, C. *Konzeption und Implementation von Modulen zur Integration funktionaler Modelle und Simulationsunterstützung in ein Konfigurierungssystem*. Diplomarbeit Fachbereich Informatik, Universität Hamburg, 1997
- Linnemann, B. *Technologietransfer von KI-Methoden in Konfigurierungs-Expertensysteme*. Diplomarbeit, Hamburg, 1994
- Marcus, S., Stout, J., McDermott, J. *VT: An Expert Elevator Designer That Uses Knowledge-Based Backtracking*. in: AI Magazine, Vol. Spring 88, p. 95-112, 1988
- McDermott, J. *RI: A Rule-Based Configurer of Computer Systems*. in: Artificial Intelligence, Vol. 19, (1), p. 39 - 88, 1982
- McGuinness, D.L., Wright, J.R. *An Industrial-Strength Description Logic-Based Configurator Platform*. in: IEEE Intelligent Systems Jul/Aug 98, p. 69-77, 1998
- Navinchandra, D. *Exploration and Innovation in Design* Springer Verlag, 1991.
- Neumann, B., Weiner, J. *Anlagenkonzept und Bilanzverarbeitung*. in: 3. Workshop Planen und Konfigurieren, Berlin, 1989 Arbeitspapiere GMD
- Neumann, B. *KONWERK – Technologietransfer mit einem Werkzeug*. in: [Günter 95b]
- Olivier, P., Nakata, K., Landon, M., McManus, A. *Analogical Representations for Mechanism Synthesis*. in: Proc. 12th European Conf. on Artificial Intelligence, Budapest 1996, p. 506-510

- Paulokat, J., Weiß, S. *Fallauswahl und fallbasierte Steuerung bei der nichtlinearen hierarchischen Planung*. in: Beiträge zum 7. Workshop Planen und Konfigurieren, Hamburg, GMD-Arbeitspapier Nr. 723, p. 109 - 120, 1993
- Paulokat, J. (Hrsg.) *(A)TMS in Expert Systems*. SEKI-Report Universität Kaiserslautern, 1992
- Pfützner, K. *Fallbasierte Konfigurierung technischer Systeme*. in: Künstliche Intelligenz (1), p. 24-30, 1993
- Puppe, F. *Problemlösungsmethoden in Expertensystemen*. Springer Verlag, 1990
- Richter, M.M., B. Bachmann, A. Bernardi *ARC-TEC - ein Beitrag zur wissensbasierten Unterstützung der industriellen Praxis*. in: Künstliche Intelligenz, 8 (2), p. 52-56, 1994.
- Richter, M.M., A. Bernardi, C. Klauck, R. Legleitner *Akquisition und Repräsentation von technischem Wissen für Planungsaufgaben im Bereich der Fertigungstechnik*. Research Report, RR-91-23, DFKI, 1991
- Richter, M.M. *Neue Aufgaben beim Planen und Konfigurieren*. in: J. Sauer, A. Günter, J. Hertzberg (Hrsg) 10. Workshop Planen und Konfigurieren, infix Verlag, p. 3-15, 1996
- Schädler, K. *Ansätze zur Abhängigkeitsverwaltung in KONWERK*. in: [Günter 95b], p. 217-228, 1995
- Schirp, W. *Einsatz expliziter Begründungen in einem Assistenzsystem für Konfigurierungsaufgaben*. in: J. Sauer, A. Günter, J. Hertzberg (Hrsg) 10. Workshop Planen und Konfigurieren, infix Verlag, p. 75-86, 1996
- Schumann, O. Schumann, S. *Modellierung von Unschärfe in KONWERK*. in: [Günter 95b], p. 97-117
- Stein, B. *Functional Models in Configuration Systems*. Dissertation, GH-Universität Paderborn, 1995
- Stein, B., Curatolo, D. *Model Formulation and Configuration of Technical Systems*. in: J. Sauer, A. Günter, J. Hertzberg (Hrsg) 10. Workshop Planen und Konfigurieren, infix Verlag, p. 56-70, 1996
- Stumptner, M. *An overview of knowledge-based configuration*. in: AI Com 10 (2), p. 111-126, 1997
- Tank, W. *Ein produktorientierter Technologietransfer für wissensbasiertes Konfigurieren*. in: J. Sauer, A. Günter, J. Hertzberg (Hrsg) 10. Workshop Planen und Konfigurieren, infix Verlag, p. 46-54, 1996
- Tong, C., Sriram, D. (Hrsg.) *Design Representation and Models of Routine Design*. Academic Press, 1992
- Ulrich, K.T., Seering, W.P. *Function Sharing in Mechanical Design*. in: C. Tang & D. Sriram (Hrsg.) *Artificial Intelligence in Engineering Design*, Vol. II, p. 185-214, Academic Press, 1992
- VDI *Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte*. VDI-Handbuch Konstruktion, Beuth Verlag, Berlin, 1993
- Weiß, S. *Fallbasiertes Schließen in Deutschland – Eine Übersicht*. in: KI 4/92, FBO-Verlag Baden-Baden, p.46-51, 1992
- Wright, J.R. et al. *A Knowledge-based Configurator that Supports Sales, Engineering and Manufacturing at AT&T Network Systems*. in: AI Magazine Vol. 14, No. 3, p. 67-88, 1993