

EngCon

A Flexible Domain-Independent Configuration Engine

Oliver Hollmann, Thomas Wagner¹ and Andreas Guenter²

Abstract. The knowledge-based configuration engine *EngCon* is presented which is developed in a cooperation of TZI Bremen and LENZE GmbH & Co KG Hameln, Germany.

We discuss the system architecture and the configuration techniques used in *EngCon*. In our demonstration we try to focus on the differences to the LISP prototype *KONWERK* which was developed at the University of Hamburg, Germany.

1 INTRODUCTION

The configuration of products with a wealth of variants has enormously gained in relevance over the last years. Customers increasingly expect consideration of their individual requirements on a product. There was a paradigm switch from *mass production* to *mass customization*. Intelligent configuration and design tools support sales managers, engineers, and customers to layout and configure individual, innovative, and complex products.

The implementation of a knowledge-based configuration engine in a company leads to more transparency and efficiency in the usage of expert knowledge. The specific configuration knowledge of a domain expert can be used by colleagues or customers directly.

In the automobile industry, for example, a customer can configure a car directly on the internet page of a company³ ⁴ and can order it without any personal contact to a dealer. An overview of online configurators is given in [Boehm et al., 1999].

In this paper we discuss the domain-independent, knowledge-based configurator *EngCon*. The main idea of the tool is methodically based on the LISP prototype *KONWERK* (see [Guenter, 1995]). But in detail there are some differences and extensions.

The next section provides a short introduction to knowledge-based configuration. Section 3 shows the configuration engine *EngCon*. We discuss the knowledge representation, problem-solving methods, architecture and system integration with CRM- and ERP-systems. Finally, a short summary and some future directions of *EngCon* are given in section 4.

2 KNOWLEDGE-BASED CONFIGURATION

Solving a configuration task means selecting, parametering, and composing a system from single components to a valid solution according to all requirements and constraints [Sabin, 1998] [Guenter, 1995]. A configuration task consists of a problem

representation and algorithms generating a solution. There are several problem-solving methods for configuration problems (e.g. logic-based, structure-oriented, resource-oriented, associative and function-oriented) which were discussed in [Brinkop, 1999] and [Guenter, 1995].

The domain-independent configuration tool *EngCon* [Arlt et al., 1999] works with the structure-oriented configuration method. The tool is successfully used for the complex product configuration of electronic drives. *EngCon* is implemented in Java and has an ontology for representing domain knowledge. Domain concepts can be defined with parameters and relations in a concept hierarchy. Modeling descriptions for taxonomic (is-a), partonomic (has-part) and user-defined relations can be used. The action of the configuration engine can be controlled with declarative control knowledge. Requirements and restrictions between configuration objects are represented with constraints.

3 ENGCN TECHNOLOGIES

The domain-independent, knowledge-based configurator *EngCon* is successfully used for the configuration of electronic drive systems. The structure-oriented configuration method is well suitable for technical domains with a wealth of variants and relations between the components. The user can develop step by step the solution guided by the configurator (see figure 1). In this section we discuss the technologies of *EngCon* in detail.

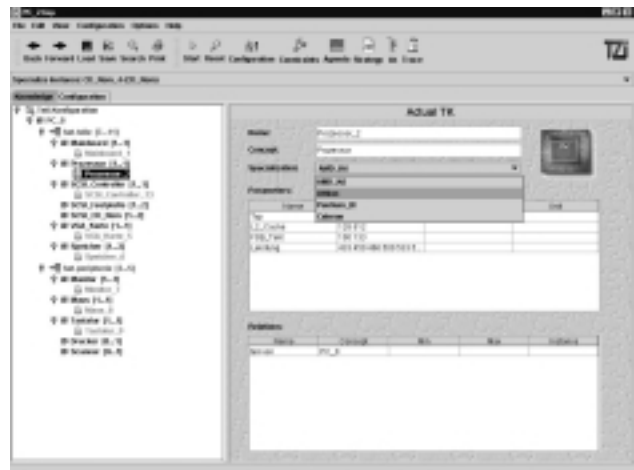


Figure 1. EngCon System

¹ TZI, Center for Computing Technologies, University of Bremen, D-28359 Bremen, Germany, email: {oho,twagner}@tzi.de

² HITeC e.V., University of Hamburg, D-22527 Hamburg, Germany, email: guenter@informatik.uni-hamburg.de

³ <http://www.opel.de/webkauf>

⁴ <http://www.bmw.de/carconfigurator>

3.1 Knowledge Representation

The domain knowledge is represented in an object-oriented concept-hierarchy. Domain objects are defined with parameters and relations. *EngCon* provides the *taxonomic* (is-a) and *partonomic* relation (has-part).

```
(def-do
  :name <name>
  :super-concept <name>
  [:parameters (<parameter>[<facette>]*)]
  [:relations (<relation>[<facette>]*)])
```

A *facette* might be for example a default value or a measurement unit. The representation of concepts is equal to the frame-based representation in *KONWERK* (see [Gunter, 1995]).

The configuration process in *EngCon* can be structured in different phases which are described by control knowledge (see [Gunter, 1992]). In a strategy we can focus the agenda generation on specific configuration objects. The processing of the agenda can be controlled with agenda selection criteria. You can define a stack of calculation methods (see subsection 3.2) for the evaluation of a configuration step. Strategies are changed if the current agenda is empty or the start condition of a strategy with a higher priority is true.

The control knowledge of *EngCon* describes the functionality of an intelligent assistant. A user can configure either interactively or guided by the tool.

Other than in *KONWERK*, the look and feel of configuration objects in a user interface can be defined with declarative presentation concepts:

```
(def-presentation
  :name <name>
  :super-presentation <name>
  :concept <concept-name> | <task-name>
  [:strategy <strategy-name>]
  [:parameters (<parameter>*)]
  [:relations (<relation>*)])
```

A parameter of a presentation concept might be for example an icon of a component which should be presented for a configuration object in a specific situation of the configuration process. Depending on an interacting user, it is possible to describe different configuration modes (e.g. novice and expert) and present detailed informations.

3.2 Problem-Solving Methods

The configurator *EngCon* works agenda-based and provides several calculation methods to determine the value in a configuration step. In a top-down design a configuration object can be *parametrized*, *specialized* and *decomposed* into parts. In a bottom-up or mixed design, components can be *integrated* in complex configuration objects, too. A configuration decision can be made by:

- user-interaction
- default-value-takeover
- dynamic-default-calculation (e.g. min/max value)
- calculation functions
- constraints
- taxonomic inferences

Constraints are used to check the consistency of a partial solution and to restrict and propagate values. There are different constraint

relations provided in *EngCon* (tuple, function, java, specialize, decompose) which are defined in a conceptual way. The bindings of constraint pins are defined with variable-pattern-pairs. The constraint relations are instantiated incrementally during the configuration process. Only the constraints for actual configuration objects are in the scope of the constraint propagation.

```
(def-conceptual-constraint
  :name <name>
  :variable-pattern-pairs ((<var><pattern>)* )
  :constraint-calls ((<cons><var><slot>)*))
```

Tuple constraints can be defined directly in a relational database system. The evaluation is done by SQL-queries via JDBC.

It is possible to *specialize* or *decompose* configuration objects automatically via constraints.

JAVA constraints are a flexible and powerful way to define customized constraint relations. The functionality of a constraint can be implemented in a JAVA method with an clear API to the *EngCon* constraint-solver. With this powerful mechanism the availability of a component for example can be checked through a request to an ERP-system.

If the values of a configuration object are restricted in a way that there is only one possible specialization in the concept hierarchy, the specialization step will be automatically done by the taxonomic-inference mechanism of *EngCon*. Complex configuration objects will be decomposed automatically to the minimum number of components defined in the knowledge base.

3.3 Integration

The concept hierarchy of *EngCon* corresponds to a company's product model. In the knowledge acquisition process the concept definitions for the bottom levels of the concept hierarchy can be generated automatically from the database of an ERP-system (see figure 2). The higher levels of the concept hierarchy have to be modeled by a domain expert and knowledge engineer.

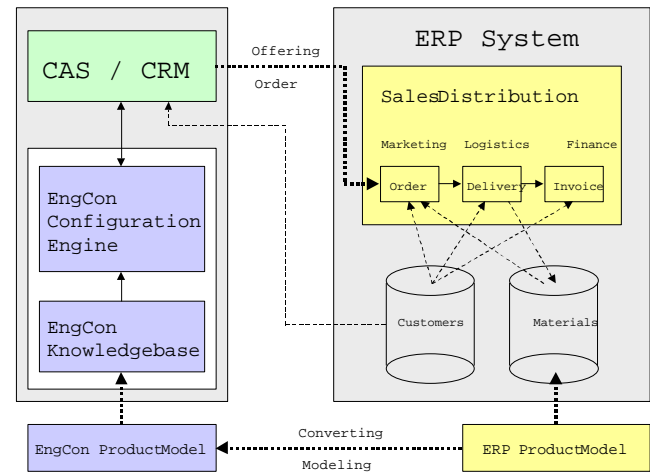


Figure 2. EngCon ERP-Integration

The solution of the configurator can be directly integrated in a supply or an order. *EngCon* provides interfaces to CRM-systems. The configuration process can be initialized by a CRM-system and the

output can be exported and included in the supply text or order text. A closer integration with an ERP-system of a company can be achieved by the implementation of JAVA constraints which initiate specific transactions (see subsection 3.2).

3.4 Implementation

The configurator *EngCon* has a flexible component architecture based on JAVA-2 technology. The platform-independent JAVA technology allows the usage of the configurator in heterogeneous software environments and provides a good basis for internet configuration. The interface of *EngCon* can be bridged for example with JAVA-RMI, and the user interface might be an Applet.

The kernel of *EngCon* can be integrated in domain-specific user interfaces. It is possible to embed the configurator via middleware technology in non-JAVA environments.

In an early phase of our project we integrated the configurator via SUN Microsystems *ActiveXBridge* in a Visual Basic user interface.

4 SUMMARY AND FUTURE DIRECTIONS

The configurator *EngCon* is successfully used in technical domains. A future extension might support 3-D configurations with spatial constraints.

Another task for the future will be the storage of solutions and their *case-based retrieval*. Time-intensive configuration processes can be saved, if the tool provides traditional solutions to a configuration task which can be reconfigured.

We are actually working on an online solution for the internet. Therefore the knowledge representation should be transformed to XML which seems to be the standard representation in eCommerce. There are several XML-approaches to represent product data in electronic catalogues⁵ or to define interchange formats⁶ which can be integrated in *EngCon*.

One more extension is the development of alternative (partial) solutions and their assessment for decision support during a configuration. This will be interesting for online configuration, too, because there is no expert available to influence the customer in his decision. Configurations have to fit different heterogeneous requirements which can lead to various conflict assessments. There is a conflict resolution framework described in [Hollmann et al., 2000] which can be integrated in *EngCon* for multicriteria evaluation support.

REFERENCES

- [Arlt et al., 1999] V. Arlt, A. Guenter, O. Hollmann, L. Hotz, T. Wagner. *Engineering & Configuration - a knowledge-based software tool for complex configuration tasks*, in AAAI-99 Proceedings, Orlando, AAAI-Press 1999.
- [Brinkop, 1999] A. Brinkop. *Variantenkonstruktion durch Auswertung der Abhaengigkeiten zwischen den Konstruktionsbauteilen*, Infix, 1999.
- [Boehm et al., 1999] A. Boehm, H.J. Mueller, J. Rahmer, S. Uellner. *A Discussion of Internet Configuration Systems*, in AAAI-99 Proceedings, Orlando, AAAI-Press 1999.
- [Guenter, 1992] A. Guenter. *Flexible Kontrolle in Expertensystemen zur Planung und Konfigurierung in technischen Domänen*, Infix, 1992.
- [Guenter, 1995] A. Guenter. *Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt PROKON*, Infix, 1995.
- [Guenter and Kuehn, 1999] A. Guenter and C. Kuehn. *Knowledge-Based Configuration - Survey and Future Directions*, in XPS-99 Proceedings, Lecture Notes in Artificial Intelligence No. 1570, Springer-Verlag, Wuerzburg, 1999.

⁵ <http://www.bme.de/bmecat>

⁶ <http://www.xmlmedi.org>

- [Hollmann et al., 2000] O. Hollmann, K.C. Ranze, H.J. Mueller and O. Herzog. *Conflict Resolution in Distributed Assessment Situations*, in H.J. Mueller and R. Dieng (eds.), *Computational Conflicts - Conflict Modeling for Distributed Intelligent Systems*, Springer-Verlag, 2000.
- [Sabin, 1998] D. Sabin, R. Weigel. *Product Configuration Frameworks - A Survey*, in IEEE Intelligent Systems July/August 1998, Seite 42-49.