A Framework for Rapid Development of Advanced Web-based Configurator Applications

Liliana Ardissono[1], **Alexander Felfernig**[2], **Gerhard Friedrich**[2], **Dietmar Jannach**[2], **Ralph Schäfer**[3], **Markus Zanker**[2]

Abstract. For the last two decades product configurators relying on Artificial Intelligence techniques have been successfully applied in industrial environments. These systems support the configuration of complex products and services in shorter time with fewer errors and therefore reduce costs of a mass-customization business model. The EU-funded CAWICOMS project aims at the next generation of Web-based configurator applications that cope with the new challenges of today's open, networked economy: this includes the support for heterogeneous user groups in an open market environment as well as the integration of configurable sub-products provisioned by specialized suppliers.

This paper describes the *CAWICOMS Workbench* for the development of configurator applications that support both adaptive and personalized user interaction as well as distributed configuration of products in a supply chain. The developed tools and techniques rely on a harmonized knowledge representation and knowledge acquisition mechanism, open XML-based protocols, and advanced distributed reasoning techniques. We applied the workbench based on the real-world business scenario of distributed configuration of services in the domain of IP-based Virtual Private Networks.

Keywords: Distributed product configuration, adaptive interfaces.

1 INTRODUCTION

Over the last decade the market's demand for customer-individual, configurable products at costs of mass-production has been constantly increasing and a mass-customization [15] business strategy has been adopted in many industrial sectors. Along with the growing complexity of the configurable products, the required supporting software systems (configurators) have been constantly improved and successfully applied in industrial environments [6, 11, 19]. As a result powerful tools based on different AI techniques like, e.g., Constraint Satisfaction [9, 10] are available on the market and are used to improve the corresponding business processes with respect to the reduction of order lead time or the number of faulty configurations.

However, today's networked economy imposes new demands on intelligent software support that are not adequately addressed by current configurator technology:

- More and more complex products and services that are offered to the customer as integrated *solutions* are themselves assembled from configurable sub-products supplied by highly-specialised providers. Due to organisational reasons or confidentiality issues, the integration of the complete configuration logic in one centralized knowledge base is not desirable. Therefore, adequate means for *distributed configuration* and integration of heterogeneous configuration systems have to be provided.
- Due to the increased complexity of configurable products, end users need deep knowledge about the available choices and the meaning of the individual parameters. Moreover the user's skills, interests, and their product knowledge can be quite diverse, depending on whether the user is the customer, a sales representative or a technical expert. As the development of a separate user interface for each class of users is not reasonable, our goal is to develop a generic interface that adapts itself to the current user's skills and preferences and guides the user through the configuration process.

The CAWICOMS[4] project aims at the next generation of web-based configuration systems and provides an integrated set of tools and techniques (*the CAWICOMS Workbench*) in order to improve the development process for these advanced configurator applications. The workbench combines newly developed techniques with *best-practice* methods stemming from long-term experiences in the corresponding fields and relies on industrial-strength tools.

Based on the real-world business case dealing with the configuration of networks and accompanying *services* and in the domain of Internet Protocol - Virtual Private Networks (IP-VPN) sketched in Section 2, we will subsequently describe the individual components of the CAWICOMS workbench and relate our work to existing and emerging approaches in the field.

2 CONFIGURATION OF IP-VPNs

The work in CAWICOMS is mainly driven by our industrial partner's[5] future business strategies that require distributed configuration and adaptive user interfaces. Within this section we will shortly sketch the business scenario for joint provision and configuration of IP-VPNs with multiple suppliers.[6] The description illustrates the specific challenges and requirements of the problem setting and we will subsequently show how the developed domain-independent tools and techniques can improve the development process for such applications.

[1] Dipartimento di Informatica, Università di Torino, 10149 Torino, Italy

[2] Computer Science and Manufacturing, University of Klagenfurt, 9020 Klagenfurt, Austria

[3] German Research Center for Artificial Intelligence (DFKI), 66123 Saarbrücken, Germany

[4] CAWICOMS is the acronym for "Customer-Adaptive Web Interface for the Configuration of Products and Services with Multiple Suppliers". This work was partly funded by the EU through the IST Programme under contract IST-1999-10688. (www.cawicoms.org)

[5] BTexact Technologies, Telecom Italia Lab, ETIS, ILOG SA.

[6] A second business case from the domain of telecommunication switching systems was used for evaluating the generality of our approaches and tools.

IP-based Virtual Private Networks are used to extend a company's private network to remote offices, business partners, or roaming users [7]. They provide secure and reliable connections and are run over the public Internet as well as dedicated backbone networks owned by the providers. While today's IP-VPNs are typically limited within country borders and the provider's backbone network, the future business model includes both transnational multi-provider networks as well as the provision of integrated *services* that also include, e.g., installation support or router configuration at the customer's site. From a sales perspective, specialised resellers offer their customers integrated solutions that consist of basic services provided by the suppliers. The configuration problem for the reseller consists of assembling and configuring an appropriate set of basic services that satisfy the customer's demands and obey technical and non-technical constraints. Note that this process also includes *supplier selection* and optimization, i.e., the same basic service can be provisioned by several suppliers. In the envisioned business scenario, suppliers *publish* their services, e.g., traffic capacity on the backbone network, to the resellers including technical details like the ranges of available bandwidth, quality of service, or prices.

Figure 1 sketches the problem setting for the reseller: The edges of the network depict the available network segments. For each external access point there are several alternative connections to the backbone network. The vertices within the backbone network represent interconnects between different company's backbone segments. In this example, the customer wants to connect his company sites in London, Paris, and Milan and specifies the required parameters like bandwidth or latency.

The configurator then starts the search process for an appropriate set of network segments, hardware components like customer-site routers and services like installation and maintenance support satisfying the customer's requirements. This high-level solution design serves as a basis for the quotation phase.

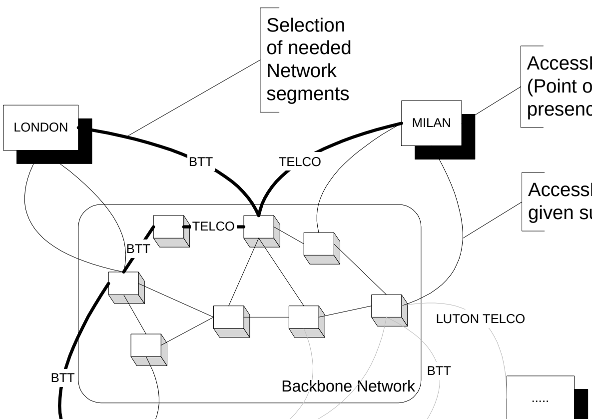

Figure 1. Reseller's view on IP-VPN Network

In a second step, the detailed configuration of the VPN solution has to take place: Therefore the results of the first step have to be transmitted to the specialized network configurators at the supplier's site. These systems are used to derive additional low-level technical settings (e.g., router configuration or bandwidth reservation) and return some of the parameters that are relevant for the customer back to the integrating configurator. These results are checked for consistency and then incorporated in the final solution for the customer.

Given this scenario the following key requirements not addressed by current configurator technology can be identified:

Product ontologies. Product classification and automated advertisement on electronic marketplaces for standard products is nowadays quite well supported. In contrast, there is little support for complex, customizable products. In order to jointly configure such solutions, the supply chain participants have to share knowledge about the products and have to agree on a common methodology to model the products and exchange information at run-time.

Distributed reasoning. Within the last years significant advances in the field of Distributed AI and, in particular, in the fields of Multi-Agent Systems and Distributed Constraint Satisfaction have been made [16, 22]. However, most of these approaches do not fit well the characteristics of distributed configuration for various reasons. For instance, configuration systems typically do not work autonomously or fully parallel like, e.g., agents do. Moreover, they require very specific problem solving mechanisms thus making the integration of legacy systems problematic. Therefore a simple and open protocol for distributed reasoning and an exchange format for complex data structures has to be employed.

Adaptive User interfaces (UI) and UI development. When introducing complex and evolving products, there is typically a shortage of trained sales people[7]. Therefore, during the quotation phase, people with different backgrounds, skills, and preferences will use the system. In order to be accepted by the users, a personalized user interface is needed that adapts itself to the user's characteristics. Another typical problem for such systems is that changes in the knowledge base (e.g., new parameters) have to be reflected in the UI. Therefore, the development and maintenance of the UI must be based on adequate tools and support for automated generation and maintenance of interface elements.

3 THE CAWICOMS WORKBENCH

Within this section we will shortly describe the components of the developed workbench and give references to additional publications and public project documents.

3.1 Knowledge representation methodology

Adequate representation of the configuration knowledge is the glue for the tools and techniques developed in the workbench. While it is well known that knowledge acquisition and maintenance is a critical factor when developing centralized configuration systems [6] it is even more important in our environment, because

- knowledge has to be shared among participants in the supply chain and different reasoning mechanisms and existing tools must be integrated, and
- personalized user defaults can be annotated in the product model and parts of the UI can be generated from the knowledge base.

Knowledge representation in CAWICOMS relies on the widely adopted *component-port* approach for configuration [6, 10, 13, 19]. We developed a domain-independent meta-model and a UML-based (Unified Modeling Language) graphical notation with precise semantics [4] that comprises modeling concepts typical for the configuration domain [17]. The resulting product models can be further enriched with constraints expressed in OCL (Object Constraint Lan-

[7] Note, that for simple services also the customer uses the configurator.

guage) [18] and are independent of specific tools and their proprietary knowledge representation mechanisms. In addition to improving the knowledge acquisition and maintenance process for centralized configurators this modeling techniques serves as a common language among the participants in the supply chain thus alleviating the knowledge-integration process.

Furthermore, we also defined a human-readable, textual representation of the product models based on XML-Schema[8] documents and implemented tools for automated document generation from commercial UML editors based on, e.g., Rational Rose[9]. These documents also serve as a starting point for further knowledge acquisition, i.e., knowledge about the available suppliers for components as well as personalization knowledge (see Figure 2).

According to the CAWICOMS architecture [1, 3] one designated *main configurator* serves both as mediator for the distributed configuration process and as interaction point for the user. This main configurator's role is taken by project partner ILOG's domain-independent, Java-based *JConfigurator* [9]. Therefore, a translator for *JConfigurator*'s Java-based knowledge representation was developed for the workbench. Finally, *JConfigurator* also includes a graphical editor for the definition of *business rules*, which allows the definition of simple preferences and constraints in a non-technical form.

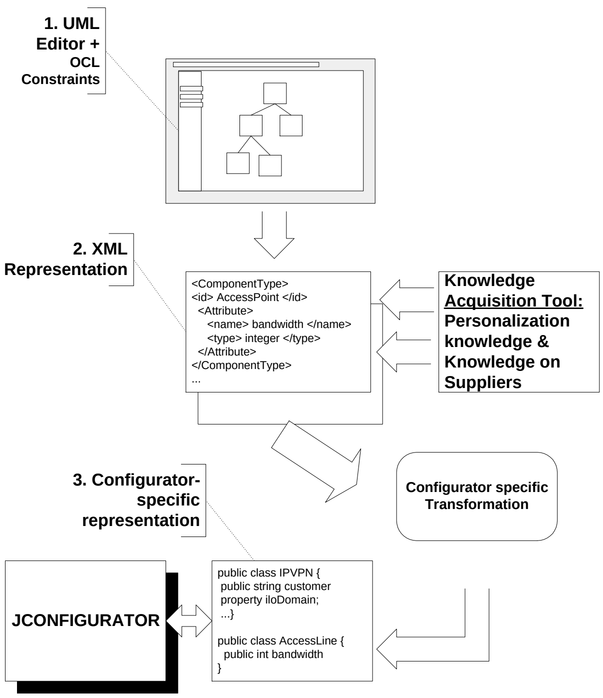

Figure 2. Knowledge Acquisition tools.

3.2 Problem solving - Distributed reasoning

Distributed Configuration problem solving in CAWICOMS relies on three pillars described in the following subsections (see Figure 3):

- knowledge sharing,
- advanced Constraint Satisfaction techniques, and
- open XML-based communication protocols

[8] see http://www.w3c.org for more information
[9] http://www.rational.com

Knowledge sharing. Distributed Problem Solving approaches always require ontological commitments on domain specific terms and conditions in order to assure the same interpretation of the content of communication by all participants. Therefore, the involved configurators must establish a shared view on the product which requires an integration and harmonization process during the setup of the supply chain. Similar to the approaches from Distributed Constraint Satisfaction [16] the involved systems can have a partial view on the product model of each other. This integration process is alleviated in our framework through the usage of a common modeling approach, i.e., in order to participate in the supply chain, suppliers have to model at least the public portions of their product models to their potential client. The mediating *main* configurator integrates that knowledge into its own product model and marks the corresponding subassemblies as to be configured by the supplier.

For non-standard, proprietary products like those available in the domain of telecommunication switches [5], knowledge is integrated in a knowledge engineering process and the cooperation is based on long-term business relationships and contracts. However, in domains like IP-VPN-provision where the configurable basic services are quite similar (e.g., Internet Service Provision) and there are many different suppliers for the same service, the goal is to come to an agreed-upon product model for these basic services. Given such an agreement, the involved communities can establish electronic marketplaces for configurable products in a specific domain. One future goal is therefore to extend existing eMarketplaces with a standardized (semantic) configuration *Web Service* [12] for its participants.

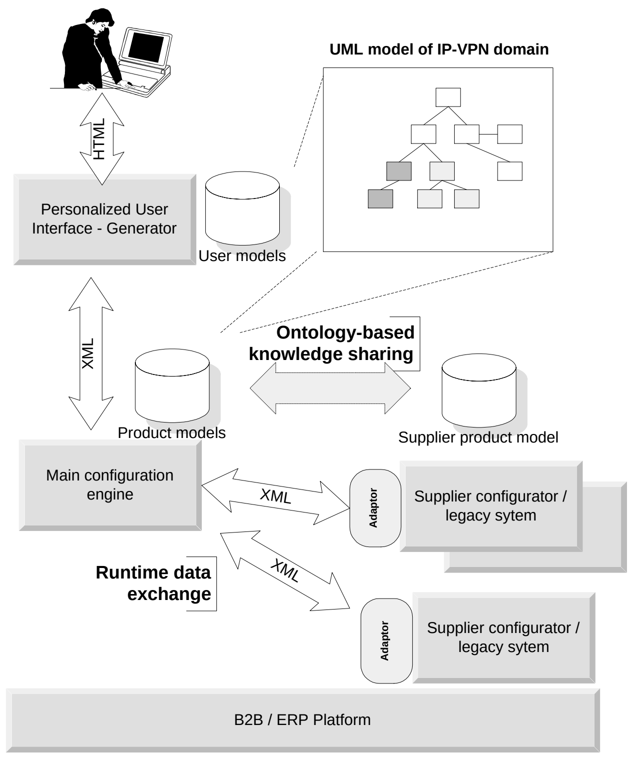

Figure 3. Architecture overview.

Distributed problem solving. Problem solving in the CAWICOMS main configurator is based on ILOG's Java-based JConfigurator [9] library that implements a Generative Constraint Satisfaction [6, 10] mechanism and provides an abstraction layer for the configuration domain. There have been some major advances in the field of Distributed CSPs (DisCSP) [16, 22] over the last years that are motivated by the inherent distributedness of the tackled problems and not by speeding up the solving process. However, current techniques have some limitations w.r.t. their applicability in the configuration domain: First, configuration *agents* do not necessarily start and work in parallel (like in Asynchronous Backtracking [22]) but are rather dynamically invoked according to the supply-chain. Second, as mentioned by [16], communication of *nogoods* implicitly exposes constraints to other agents which is problematic for confidentiality reasons. Finally, these approaches are often limited to binary constraints and static CSPs, not taking the dynamicity of configuration problems into account. Due to these limitations, the current version of the CAWICOMS workbench implements two application-dependent modes of operation for distributed configuration:

1. *Synchronous configuration* of subassemblies.
2. Two-phase *asynchronous configuration*.

In the first mode, the forward-checking, backtracking algorithm of JConfigurator is extended (without changing the core of the library) as follows: During the knowledge integration phase, the individual subassemblies (components and variables) of the product model are annotated with information about the supplier system that can configure the sub-assembly. The main configurator loads this knowledge base and starts the configuration task according to the user requirements. Once it detects that a supplier's subassembly has to be included and configured, the remote configuration system is contacted and the relevant portion of the current search state (current variable domains and assignments) are handed over. The main configurator then waits until the supplier system reports a solution or a failure in finding a solution. If the solution provided by the supplier is consistent with all constraints, the main configurator resumes its search process, else, backtracking in the main configurator occurs. Note, that this operational mode does not support any parallelism but is sound and complete because of exhaustive domain exploration.

The second mode is highly motivated by the specifics of the IP-VPN provision business case. Here, the main configurator carries out the configuration process without contacting the supplier systems in the first phase. From a CSP perspective, it simply does not assign labels to variables that are marked to be computed by a supplier system (it performs domain reduction for these variables though). Once such a partial solution is found that suffices for a quote, all the needed supplier systems are contacted in parallel and handed over the current values and domains, in order to complete their part of the configuration. This can be done because in this application domain we can assume that the supplier systems will always find a solution for any valid partial configuration computed by the main configurator. Another novel aspect covered in this mode is that we support long-lasting transactions, i.e., after sending out the requests the main configurator is de-activated because low-level configuration at the supplier site may require, e.g., human interaction.

Our experiences show that for some domains mechanisms can be employed that are simpler than current DisCSP approaches, but take specifics of the domain into account. Our current work extends DisCSP algorithms for the configuration domain, i.e., with dynamic generation of variables and dynamic involvement of configuration agents.

Open communication protocols and platform. A key issue also for centralized configurator applications is the integration into the corporate's software infrastructure like ERP systems or B2B platforms. This issue is addressed in CAWICOMS and JConfigurator by providing adequate interfaces [3], e.g., by allowing the configurator knowledge base to contain references to external data sources via a database connectivity interface and therefore, data does not have to be kept redundantly in the knowledge base. Moreover, the workbench is completely developed on the J2EE [20] platform, which is particularly designed for easy integration of components into enterprise environments.

There are several choices for connecting distributed systems both w.r.t. the technical level (like Java RMI, DCOM, or CORBA) as well as to the underlying communication paradigm (e.g., client-server style or agent-based negotiation). Communication in the CAWICOMS workbench is done based on an XML-based protocol (*WebConnector* protocol) developed for the configuration domain according to the evolving paradigm of XML-based SOAP messaging[10] and *Web Services* (see Figure 3). This protocol defines

- a fixed set of methods with defined semantics for the configuration domain, like creating components, setting values for parameters, initiation of the search process, or retrieving results,
- a mechanism to exchange complex data structures like configuration results and a language for expressing navigation expressions within these data structures, and
- extensibility mechanisms for special domains and support for transactions for HTTP-based configuration sessions.

The main advantages of this approach are in the independence from individual programming languages and security-related issues, e.g., the possibility to communicate over HTTP connections without a need to open corporate firewalls for IP-based connections[11]. As depicted in Figure 3, such documents are exchanged both for configurator-to-configurator communication, i.e., participating systems have to implement the protocol and interpret the messages with respect to the underlying system, as well as for communication with the user interface (see next section). Besides the definition of the protocol and the interfaces, the CAWICOMS workbench includes a set of (commercially available [9]) corresponding software tools like XML-parsers, components for transaction support, as well as tools for alleviated interface development like for XML-transformation or JSP tag-libraries [20].

3.3 Adaptive user interaction

Due to the aforementioned heterogeneity of the system's user groups and the complexity of the configuration task itself, our toolkit provides mechanisms for the development of user interfaces that adapt their presentation and interaction style to the current user's needs and skills (see [2]). In this way, the CAWICOMS interface makes the interaction easier for the user: for instance, the system reduces the number of questions by proposing appropriate default values or by giving additional information and more guidance. The CAWICOMS workbench supports these requirements in three dimensions:

1. Probabilistic user modelling: In order to personalize the interaction, the system needs information about the user's properties. Some properties are directly entered by the user, which can be facts like 'nationality' or self-assessed estimates on, e.g., 'experience'. Moreover, Bayesian networks [14] are used in the system to perform probabilistic estimations of uncertain user properties: The user's expertise is estimated based on the work from [8]. In order to propose the user a configuration (or individual defaults) that matches his/her interests best, the interests of the user have to be estimated. As evaluation

[10] see www.w3c.org for details.

[11] The WebConnector toolkit also supports Java-based communication for homogeneous environments.

process, we ascribe to the user the widely-adopted Multi-Attribute-Utility Theory [21]: After defining the *interest dimensions* in the application domain (e.g., *reliability* or *economy*) we define the relations between individual parameter values of the configurable product and these dimensions: As an example in our application domain, selection of some specific router type improves the configuration with respect to reliability but worsens it on economy. When a new user interacts with the system, a first estimate on the user's properties is made on the basis of his/her personal characteristics, which enable the system to classify the user in typical user classes (e.g., technicians, etc.). During the interactive configuration sessions the estimate of the user's interests in the aforementioned dimensions is updated by interpretation of the user's behaviour in the following typical interaction situations: expression of interest, change of some proposed default value, or acceptance or rejection of a proposed configuration.
2. Personalization based on user models: Given the information about the properties of the individual user, flexible dialogue management and adaptive hypermedia techniques are applied to tailor the configuration task, the elicitation of information about configuration parameters and the presentation of the configuration solutions to the user's expertise and interests. The personalisation strategies are also aimed at shortening the configuration session by inferring suitable parameter settings and proposing them to the user, therefore reducing, in general, the amount of data to be specified. As the user's behavior is continuously tracked by the system to revise the estimates about interests and expertise, the application of personalisation techniques and the dynamic generation of the user interface support a reactive personalisation of the interaction, so that the system can repair the possibly wrong personalisation behaviour (due to errors in the estimation of the user's properties) and also adapt to possible changes in the user's attitudes [2].

A rule-based mechanism is exploited to compute a personalisation strategy and the corresponding interface elements based on information from various sources. At each stage of the configuration process, the configuration reasoner can be queried about the current set of parameters that have to be asked to the user, in order to apply the personalisation strategies and choose the most convenient way for setting them (e.g., eliciting information from the user, automatically setting them, or asking the user about abstract concepts related to the parameters and infer the corresponding values to be set). Therefore the system can, e.g., propose values that match the user's interests or hide technical details in cases where his/her expertise is low.
3. Support for user interface generation: Due to frequent product innovations configuration knowledge bases are constantly changing. Moreover, the configuration process is driven by an underlying search process, i.e., the number, sequence and possible values for variables depend on previous user inputs. Maintainable user interfaces must therefore be dynamically generated at configuration time. Within the CAWICOMS workbench dynamic generation of (Web-)interfaces is made feasible by the provision of libraries (JSP tag libraries) for rapid development of web-based user interfaces. Moreover, the configuration server provides adequate APIs (Application Programming Interface) for querying the current status of the configuration process, e.g., the set of open choices with their current domains.

4 CONCLUSION

Driven by the new requirements of today's networked markets and based on emerging business models in the telecommunication domain a framework for rapid development of advanced web-based configurators was developed. The CAWICOMS workbench includes technologies and tools addressing the demands of personalized and adaptive user interfaces and advanced cooperative problem solving in the supply chain. The workbench's components are both based on the implementation of best practices in the corresponding fields as well as on newly developed mechanisms and algorithms. Furthermore, the innate need of openness and integratability in a dynamically changing, heterogeneous environment were taken into account by complying to existing and emerging standards and by relying on state-of-the-art software platforms and XML-based protocols.

REFERENCES

[1] L. Ardissono, A. Felfernig, G. Friedrich, D. Jannach, R. Schäfer, and M. Zanker, *Intelligent Interfaces for Distributed Web-Based Product and Service Configuration*. Proc. Web Intelligence 2001, Maebashi, Japan, pp. 184-188.

[2] L. Ardissono, A. Felfernig, G. Friedrich, A. Goy, D. Jannach, M. Meyer, G. Petrone, R. Schäfer, W. Schütz, and M. Zanker, *Personalising on-line configuration of products and services*, Proc. 15^{th} European Conference on AI (ECAI'02), Lyon, France, IOS Press, 2002.

[3] CAWICOMS Consortium, *D01: Requirements, Application Scenarios, Overall Architecture, and Test Specification*, *www.cawicoms.org*, 2001.

[4] A. Felfernig, G. Friedrich, and D. Jannach , *UML as domain specific language for the construction of knowledge-based configuration systems*, In: International Journal of Software Engineering and Knowledge Engineering (IJSEKE), **10**(4), 2000, pp. 449-470.

[5] A. Felfernig, G. Friedrich, D. Jannach, C. Russ, and M. Zanker, *Multi-site product configuration of telecommunication switches*, Proc. Applied Informatics (AI 2002), Innsbruck, Austria, 2002.

[6] G. Fleischanderl, G. Friedrich, A. Haselböck, H. Schreiner, and M. Stumptner. *Configuring Large Systems Using Generative Constraint Satisfaction*, IEEE Intelligent Systems. **13**(4), 1998, pp. 59-68.

[7] Infonetics Research: Virtual Private Networks, White Paper, *http://www.infonetics.com*, 1997.

[8] A. Jameson, *Knowing What Others Know - Studies in Intuitive Psychometrics*. PhD thesis, University of Amsterdam, Netherlands, 1990.

[9] U. Junker, *Preference-programming for Configuration*, Proc. IJCAI'01 - Configuration Workshop, Seattle, 2001

[10] D. Mailharro, *A Classification and Constraint-based Framework for Configuration*, AI in Engineering, Design and Manufacturing, **12**(98), Cambridge University Press, 1998, pp. 383-397.

[11] D.L. McGuiness and J.R. Wright, Conceptual Modeling for Configuration: A Description Logic-based Approach, AI in Engineering, Design and Manufacturing (AI EDAM), **12**(98), Cambridge University Press, 1998, pp. 333-344.

[12] S. McIlraith, T.C. Son, and H. Zeng, *Semantic Web Services*, IEEE Intelligent Systems, **16**(2), 2001, pp. 46-53.

[13] S. Mittal and F. Frayman, *Towards a generic model of configuration tasks*, Proc. Intl. Joint Conference on Artificial Intelligence (IJCAI'89), Detroit, Morgan Kaufmann, 1989, pp.1395-1401.

[14] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, CA, 1988.

[15] B.J. PineII, B. Victor and A.C. Boynton, *Making Mass Customization Work*, Harvard Business Review, Sep./Oct. 1993, pp. 109-119.

[16] M. Silaghi, D. Sam-Haroud, and B. Faltings. *Asynchronous Search with Aggregations*. Proc. AAAI/IAAI 2000, Austin, TX, 2000, pp. 917-922.

[17] T. Soininen, J. Tiihonen, T. Männistö, and R. Sulonen: *Towards a general ontology of configuration*, AI in Engineering, Design and Manufacturing, **12**(98), Cambridge University Press, 1998, pp. 357-372.

[18] J. Rumbaugh, I. Jacobson and G. Booch, *The Unified Modeling Language Reference Manual*, Addison-Wesley, 1998.

[19] M. Stumptner, *An overview of knowledge-based configuration*, AI Communications **10**(2), 1997, pp. 111-126.

[20] Sun Microsystems Java Web pages, *java.sun.com*, 2002.

[21] D. Winterfeld and W. Edwards, *Decision analysis and behavioral research*, Cambridge University Press, Cambridge, England, 1996.

[22] M. Yokoo, *Distributed constraint satisfaction - foundations of cooperation in multi-agent systems*. Springer Verlag, Berlin, Germany, 2001.