Adaptive User Interfaces for Web Configurators

Liliana Ardissono[♦], Mathias Bauer[✦], Luca Console[♦], Gerhard Friedrich[•], Markus Zanker[•]

[♦] Dipartimento di Informatica, Università di Torino
Corso Svizzera 185, 10149 Torino, Italy
{liliana, lconsole}@di.unito.it

[✦] DFKI
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
bauer@dfki.de

[•] Universität Klagenfurt
Institut für Wirtschaftsinformatik und Anwendungssysteme, Produktionsinformatik
Universitätsstraße 65-67, 9020 Klagenfurt, Austria
{gerhard.friedrich, markus.zanker}@ifit.uni-klu.ac.at

Abstract
In this paper we describe how the sales process of complex customizable products can be supported by an adaptive configurator interface. The task is to offer the appropriate choices to the customer in an adequate way of interaction. We illustrate our concepts with a short example from the domain of computer hardware configuration. Moreover, an architecture for an adaptive Web-based configurator is proposed.

1. Introduction to the problem domain

Internet technology and electronic commerce offer new possibilities of interaction between customers and suppliers in the sales of products and services. One of the main characteristics of electronic commerce is that it dramatically reduces the distance between buyer and seller. Thus, it enables businesses to target very small niches, develop individual customer profiles, and essentially provide a means of marketing on a one-to-one basis. These match exactly the way many successful enterprises are conducting their business: providing niche products by addressing the specific needs of small groups of customers and delivering sophisticated and highly configured solutions, thus exploiting economies of scope. This leads to the need for integrating configuration and Web-based technology.
However, the implementation of the presentation layer of complex customizable products and services still poses a major obstacle. We are therefore addressing these identified needs by proposing an adaptive interaction with Web-based configurators. As the volume of e-commerce sales is predicted to explode in the near future [OECD99], Internet presence becomes more and more a must, not only for sales purposes, but also to distribute information about the products and services offered. Placing the necessary information on line increases the productivity of sales people and helps to boost customer satisfaction: in fact, if consumers are well informed about a product, the probability of having misconceptions or false expectations about it, is significantly reduced. Our approach to present sophisticated information about highly configurable products in an adaptive manner, which takes into account the specific customer profile, can help to overcome acceptance problems and provide valuable information over the net, which leads to an enhancement of consumer-supplier relations.

2. Requirements for the User-Interaction

The adaptation and the personalisation of the interactions between the configuration application and the e-customer is a key factor for the applicability of configuration technology in e-commerce. A configurator offers the customer several possibilities for tailoring the product to her/his needs, presenting a high number of choices and providing the user with potentially very technical and complex information. For these reasons, the dialog between user and system during the interaction is an important prerequisite for the success of a Web-configurator. See [RBMU99] for a discussion on the different levels of complexity of the interaction with a configurator.
Current interfaces to configurators are programmed in the tradition of mask-oriented, procedural user interfaces. Conversely, in the configuration domain, the dialog with the user is driven by a search process through a solution space which is guided by a default sequence of logically dependent configuration steps [FF+98]. When and which information and decision is needed from the user depends on the search process.

Further demands driven by the configuration domain are the various interaction styles that must be supported, e.g., option based configurators, example based configurators, configurators with automatic generation of configurations, configurators which check user provided configuration only. These interaction styles change depending on the customer's objectives, skills, and knowledge. Since simple toolkits for building configurator user interfaces are not adequate for real world application domains, these user interfaces are programmed ad hoc leading to high implementation and maintenance costs.

These requirements let us conclude that the dialog with the user should be adapted to personal requirements (related, for instance, with his skill level and interests) and organisational preferences (e.g., terminology, embedded knowledge of organisational processes, existing inventory, help arrangements) [BD93]. During a configuration session, the user frequently explores various solution paths. Therefore, sophisticated assistance regarding explanations, revision of partial solutions (e.g., after a change of the inputs by the user, previous user-inputs must be preserved as far as possible), and inconsistent requirements is necessary [FF+99]. Moreover, personalization techniques should be used to tailor the interaction style to the user's preferences and interests. Such techniques are based on the exploitation of user models [WK89, MT93] that describe the user's characteristics and are updated during the interaction, to refine the description of the user's characteristics. Changes in the knowledge and interests of a customer will change the content, the presentation, and the configurable features offered, e.g., domain experts tend to use the configuration capabilities of the product very deeply, whereas inexperienced customers rather prefer a selection of standard configurations with small adaptations. Provided that a customer already purchased a product and requires some kind of upgrading would be another example. For instance, in the domain of computer hardware, the adaptive Web-based configurator should consider, e.g., knowledge about the hardware or operating system environment of the individual user in order to suggest extensions compatible with his equipment. As far as the personalization of the presentation is concerned, a further option is to explain unknown components and functionalities in terms of already known concepts (e.g. products the user is known to have some experience with).

3. User Adaptivity in Configuration Context

As the configuration step is to find a valid and completely specified product structure among the alternatives described by the generic architecture of the product family [SW98], the requirement is to understand accurately the customer's needs. Therefore, seen from a more general perspective, each configuration paradigm relies on some sort of user input in order to guide or at least to start the configuration process. The interface component has the task to elicit the control information, necessary for the guidance of the process, from the user. The classic problem set for engineering an adaptive user interface in an electronic commerce environment is to tailor the selection and presentation of products to the specific needs of their customers. Thus, the features and properties of products must match the interest and preferences of users (see [AG99]) and the amount and content of the provided information must optimally meet the user's needs. However, in the configuration domain the question is not which products to present, but with how many degrees of freedom a user has to cope when configuring a product. Similar to the case of an adaptive Web store, the task of an adaptive configuration interface is to offer the appropriate choices to the customer in an adequate way of interaction. This means that for different user profiles different choices and default-values should be deduced from the configuration knowledge base. For the way of presentation several different paradigms for interaction and combinations of them are available. Therefore, our approach to incorporate adaptivity aspects into the configuration process is twofold: first, we employ several paradigmatic interaction styles offering different classes of support (e.g., example based, option based with different search strategies) to differentiate between different user capabilities. Second, we define the characteristics of the choices, which the generic product architecture offers, and add them to the knowledge base.

4. Motivating Example

For the illustration of our concepts we give a simplified example from the domain of computer hardware configuration. The following three figures contain a graphical depiction of a view on the configuration knowledge base for three different types of users (inexperienced, sophisticated, expert). Since the complexity of the user interaction is determined by the style of interaction and the amount of choices the customer can make, our example shows how the degree of freedom steadily increases as the experience of the user augments. We use the intuitive UML (Unified Modeling Language) notation enriched with the specific modeling concepts described in [FFJ99].

Inexperienced User Figure 1 shows a PC component consisting of the sub-components EIDE, Floppy-Unit and Motherboard. The multiplicities of the aggregational relations define the amount of sub-components. The interaction style could be example-based, e.g., the user could choose between two partially configured PCs (low-budget version: one Floppy-Unit, small HDD-EIDE and cheaper CPU; high-end version: two Floppy-Units, large HDD-EIDE and fast CPU) and has the additional option of adding the classic support package (attribute of the PC component).

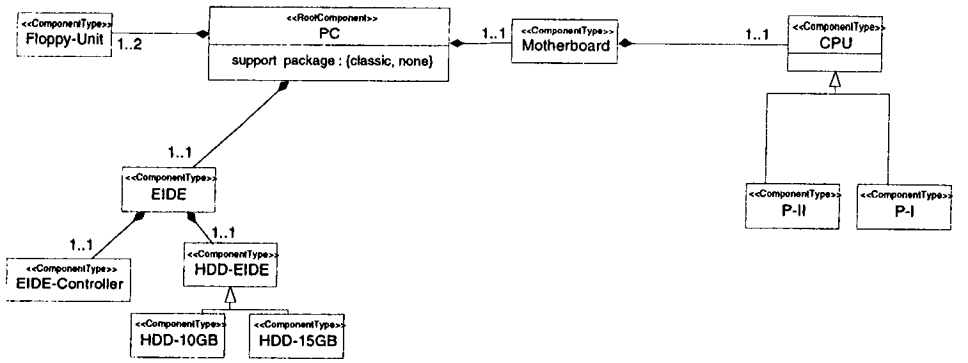

Fig. 1.: View on the configuration knowledge base for an inexperienced user

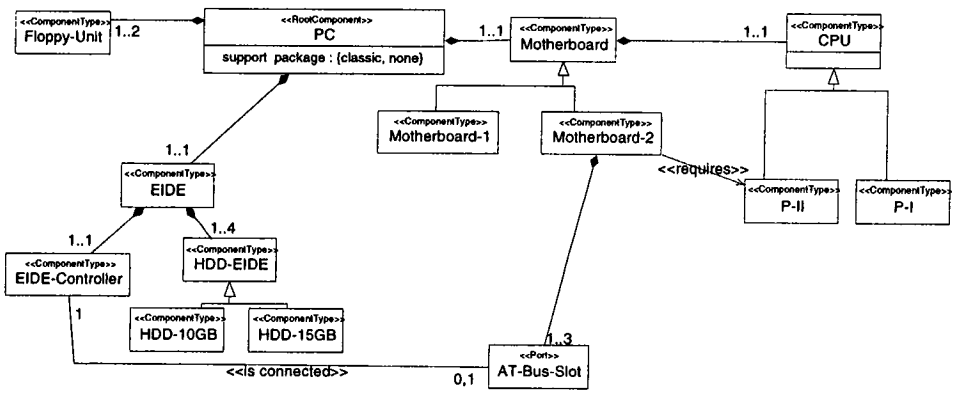

Fig. 2.: View on the configuration knowledge base for a sophisticated user

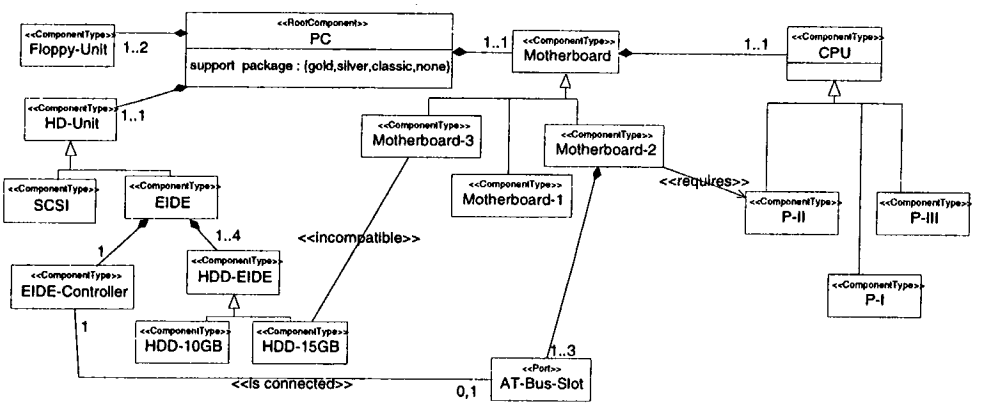

Fig. 3.: View on the configuration knowledge base for an expert user

Sophisticated User In case of a sophisticated user (see Figure 2) we would apply an option-based interaction style. Now the customer has an option between Motherboard-1 and Motherboard-2, further he can add up to four HDD-EIDEs and is informed in which slot the EIDE-Controller is inserted.

Expert User The expert user (compare to Figure 3) has even additional components to choose (e.g., a SCSI unit) and possibly wider value domains to select from (e.g., three motherboard and CPU types and two more support options). Further insightful explanations are offered why specific components can not be combined (e.g., incompatibility constraint between the large HDD-EIDE and Motherboard-3).

In the following we will shortly discuss a formal model of configuration knowledge based on the component port model, and extend it to include aspects of user adaptivity.

5. Configuration Knowledge Base

A configuration task can be characterized by a set of components, a description of their properties, namely attributes and possible attribute values, connection points (ports), and constraints on legal configurations. Given some customer requirements, the result of computing a configuration is a set of components, corresponding attribute valuations, and connections satisfying all the constraints and the customer requirements. This model has proven to be simple and powerful to describe general configuration problems and serves as a basis for configuration systems as well as for representing technical systems in general. The formulation of a configuration problem can be based on two sets of logic sentences, namely *DD* (domain description) and *SRS* (specific requirements).

DD includes the description of the different component types (types), named ports (ports), and attributes (attributes) with their domains (dom) of a configurator.

SRS specifies the user requirements on the product which should be configured. These requirements are the input for the configuration.

The configuration result is described through sets of logical sentences (*COMPS, ATTRS, CONNS*). In these sets, the employed components, the attribute values (parameters), and the established connections of a concrete customized product or service are represented.

- *COMPS* represents sets of literals of the form *type(c,t)*, where *t* is included in the set of types defined in *DD*. The constant *c* represents the identification for a component.
- *CONNS* represents sets of literals of the form *conn(c1,p1,c2,p2)*: *c1*, *c2* are component identifications from *COMPS*; *p1* (*p2*) is a port of the component *c1* (*c2*).
- *ATTRS* represents sets of literals of the form *val(c,a,v)*, where *c* is a component identification, *a* is an attribute of that component, and *v* is the actual value of the attribute, selected from the domain of the attribute.

The degrees of freedom of a product architecture depend on the size of the domains of the types, ports, attributes and dom predicates. Our approach is to offer different views on the configuration knowledge base, which have different degrees of freedom for customization. These views are parametrized on the basis of the users' expertise and interests. In our example we only introduced one parameter (i.e., user sophistication) to customize the view on the configuration knowledge base.
For instance, if the level of sophistication is low (inexperienced user), the view on the knowledge base contains among other logical sentences the following facts:

View on DD:
types = {pc, floppy-unit, motherboard, cpu, p-I, p-II, eide, eide-controller, hdd-eide, hdd-10gb, hdd-15gb}.
attributes(pc) = {support-package}.
dom(pc, support-package) = {classic, none}.

User models The user models describe the whole set of features of the customer relevant to the personalization of the interaction. They contain information concerning personal data, interaction needs and preferences for configuration settings.
As far as the interaction style is concerned, a "user sophistication" parameter specifies whether the user is inexperienced, sophisticated or expert (more parameters will be defined to describe other features useful for personalizing the interaction; e.g., the user's receptivity). The user sophistication is exploited to select the active view for the configuration process; moreover, it is exploited for the customization of the descriptions. In fact, this information enables the system to identify the concepts belonging to the active view which are difficult for the user and may need to be explained. For instance, when two options have to be proposed to the user, they could be just mentioned, or more information could be provided to discriminate among them.
The user preferences express constraints to be taken into account in the configuration process. For instance, if the user model contains a preference for high-speed computers, this information should be used to configure a computer choosing the fastest CPU available in the active view. The relation among user preferences and constraints on the active view is defined by means of a correspondance relation. Such a relation maps high-level preferences, representing a qualitative way of thinking about product variants (e.g., cheap vs. expensive, or slow vs. high-speed) on the specific components belonging to the active view.
The user model also contains information about the product variants selected by the user during the interaction.

User requirements (SRS) In our approach all the user-specific choices are included in the active view and can be automatically deduced out of the logical representation in the view itself, leaving the configuration knowledge base unchanged. The active view contains default values for configurations; moreover, it contains constraints coming from the user preferences, as well as example configurations (the user preferences are initially set into the view on the basis of the predictions contained in the user model). This assures that the configuration result is valid and complete, and contains more components than the user sees in his view: e.g., our example of a configuration result

contains the *Motherboard-1*, which is choosen by default. The customer could not influence this selection, because the component *Motherboard-1* was not in his view on the configuration knowledge. Moreover, the example contains the CPU *P-II* because the active view contained a constraint on the CPU type, coming from the mentioned user preference for fast computers.

Example of a configuration result (COMPS, CONNS, ATTRS):
type(p1,pc). type(m1,motherboard-1). type(c1,p-II).
...
val(p1, support, classic).

In the next section we roughly sketch a framework for adaptive configuration.

6. Architecture

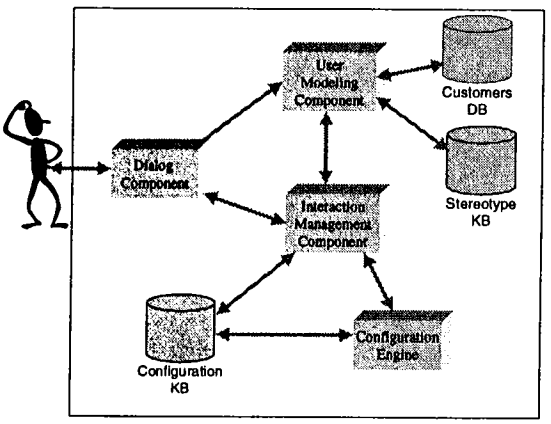

Fig 4.: Architecture for an adaptive configuration environment

Figure 4 illustrates the proposed architecture for an adaptive Web-based configurator. In the figure, the blocks represent the main components of the system and the cylinders denote the knowledge bases that they exploit in their activity.

The *Configuration KB* contains knowledge about the involved objects (often called components), conditions which must hold among these objects to ensure consistency and characteristics of the choices the generic architecture offers. Moreover, it contains the definition of the views on the knowledge about products, used to adapt the configuration task to the user. Each view is associated to a different setting of the user parameters (in our example, the views are only associated to the user sophistication). The *Configuration KB* also contains the textual descriptions of the defined components, attributes and values.

The *Stereotype KB* describes the characteristics of different customer classes specifying, for each class, the user parameters and the possible preferences over interaction styles. Moreover, this knowledge base specifies the typical domain expertise of the users belonging to the classes and the possible stereotypical preferences for configuration settings. The stereotypical information about customer classes is exploited when a new user interacts with the configuration system to achieve his initial description. The user's personal information (requested by the system at the beginning of the first interaction) is matched on the stereotypes, whose predictions are exploited to fill in the user model. Then, during the interaction, the user model is refined to take into account her/his real behavior.

The *Customers DB* contains specific information about the customers who visited the store: at the end of an interaction, the user model is stored by the system to keep permanent information about the customer. The records of this database will be used to initialize the user models of customers, when they access the configuration system again. In particular, the information regarding the users' purchases is important when consistent extensions to their equipment have to be suggested.

The *User Modeling Component (UMC)* manages the user models and reasons on the customers' expertise and preferences. The user models are updated during the interaction with the customer on the basis of the observation of his behavior. Several events are useful for learning individual information about the user: for instance, if the system presents some options, the user's selections can be registered not only to carry on the configuration process, but also to store information about his preferences. E.g., the preference for fast computers could be discovered from the fact that the system proposes the *P-I* and *P-II* processors and the user selects *P-II*. Moreover, the system might suggest some option and ask the user whether he would prefer something with different characteristics (e.g., a faster CPU, a cheaper equipment, and so forth; see [LHL97]).

The *Dialog Component* handles the logical interaction with the customer and maintains the dialog context, or history of the interaction, to keep track of the user's choices during the configuration process.

While the main configuration task is performed by the *Configuration Engine*, the *Interaction Management Component (IMC)* manages the personalized configuration process, by interfacing the *Configuration Engine* and the *UMC*. In particular, given the contents of the user model, the *IMC* selects from the *Configuration KB* the active view to be used during the interaction and updates it with the user preferences; moreover, it interacts with the *Configuration Engine* to demand (partial) configurations. Finally, it extracts the textual descriptions to be used during the interaction for asking questions, and presenting solutions and options in a form understandable to the user.

7. Project Outline

Current technology fails to address the personalization of configuration interactions adequately: the flexibility of the configuration process is not supported sufficiently in current toolkits. Therefore, we are starting a project whose main goal is the development of a toolkit for the creation of adaptive Web-based configuration systems. In this project, we will develop general description concepts for interaction styles, user interfaces, and support of the user during the configuration process. These concepts will be embedded in tools which significantly reduce the efforts to design and implement the user interfaces; these tools will also enhance the assistance to the user during the search for the most satisfying product configuration.

The success of this approach will drive the next generation of computer-based support for customisable products and services. One of the objectives of our project is the adaptation and the personalisation of the interactions with the configurator application according to the needs of the user. The behaviour of the system will be tailored with respect to many user properties, like expertise, preferences, or reason for using a system. It will provide methods and tools for the implementation of configurators and user interfaces with the ability to adapt automatically to an optimal interaction between users and configurators.

An e-commerce application based on our project's results will support the following tasks in order to achieve end-to-end satisfiable solutions for the customer:

- Identification of the customer's needs and of her/his skills, and adaptation of the interface for Web applications;
- Configuration of modular products, services, and systems integrating the offers of the sub-suppliers in order to satisfy optimally the customer's requirements and preferences;
- Adaptation of the interaction style to the customer, including a personalized presentation of the offer (configured product), according to her/his preferences and skill-level [AL+99].

Consequently, general modeling and knowledge representation concepts [FFJ99] will be developed in order to minimise the adaptation and maintenance costs for new applications. These adaptations will be implemented by the instantiation or the modification of the knowledge-base.

In order to guarantee the generality of our approach several application scenarios will be selected. These guiding examples will serve as test cases for the prototype workbench. All concepts, methods, and software development will be continuously evaluated with respect to the requirements of the test application scenarios.

References

[AG99] L. Ardissono, A. Goy. Tailoring the Interaction With Users in Electronic Shops. In Proceedings *7th International Conference on User Modeling*. Banff, MA, 1999.

[AA+99] L. Ardissono, A. Goy, R. Meo, G. Petrone, L. Console, L. Lesmo, C. Simone, P. Torasso. A configurable system for the construction of adaptive virtual stores. *World Wide Web*, 2(3): 143-159, 1999.

[BD93] D. Benyon. Adaptive systems: a solution to usability problems. *User Modelling and User-Adapted Interaction*, 3:65--87, 1993.

[FF+99] Alexander Felfernig, Gerhard E. Friedrich, Dietmar Jannach, and Markus Stumptner. Consistency based diagnosis of configuration knowledge-bases. In Proceedings *AAAI Workshop on Configuration*. Orlando, FL, July 1999.

[FF+98] Gerhard Fleischanderl, Gerhard E. Friedrich, Alois Haselböck, Herwig Schreiner, and Markus Stumptner. Configuring Large Systems Using Generative Constraint Satisfaction. *IEEE Intelligent Systems,* Vol. 13, No. 4, July/August 1998, pp. 59-68.

[FFJ99] Alexander Felfernig, Gerhard E. Friedrich, and Dietmar Jannach. UML as domain specific language for the construction of knowledge-based configuration systems. In Proceedings *Conference on Software Engineering and Knowledge Engineering* '99. Kaiserslautern, June 1999.

[LHL97] G. Linden, S. Hanks, and N. Lesh. Interactive assessment of user preference models: the Automated Travel Assistant. In Proceedings *6th International Conference on User Modeling*. Chia, Italy, 1997.

[MT93] M.F. McTear. User modelling for adaptive computer systems: a survey of recent developments. *Artificial Intelligence Review*, 7, 1993, pp. 157-184.

[OECD99] The Economic and Social Impact of Electronic Commerce: Preliminary Findings and Research Agenda, OECD Code 93 1999011P1, ISBN 92-64-16972-5, 02/99

[RBMU99] Jörg Rahmer, Andreas Böhm, Heinz-Jürgen Müller, and Stefan Ueller. A Discussion of Internet Configuration Systems. In *Proceedings of the AAAI Workshop on Configuration*, AAAI Press, Orlando, FL, July 1999.

[SW98] D. Sabin and R. Weigel. Product Configuration Frameworks-A Survey. *IEEE Intelligent Systems,* Vol. 13, No. 4, July/August 1998, pp. 42-49.

[WK89] W. Wahlster and A. Kobsa. *User Models in Dialog Systems*. Springer Verlag, Berlin, 1989.