Definition einer Parameter-Hierarchie zur Adaptierung der Benutzer-Interaktion in E-Commerce-Systemen

Wilken Schütz, Markus Meyer[1]

DFKI GmbH, Stuhlsatzenhausweg 3, 66123 Saarbrücken
{schuetz,meyer}@dfki.de

Zusammenfassung. In E-Commerce-Anwendungen stehen Kunden häufig vor dem Problem, das Produkt zu selektieren, das am besten den eigenen Präferenzen entspricht. Um die Präferenzen und Interessen des Kunden zu repräsentieren, kann die Modellierung seines Produkt-Bewertungsprozesses als Grundlage dienen. Eine multi-attributive Objektbewertung ist hierfür ein möglicher Ansatz. Ein System sollte die Interaktion mit dem Benutzer dynamisch gestalten und individuell auf den Kunden zuschneiden. In dieser Arbeit wird daher versucht, mit Hilfe einer Parameter-Hierarchie die Interaktion des Systems mit dem Benutzer zu adaptieren und auf dieser Basis eine Produktevaluierung durchzuführen.

1 Einleitung

In E-Commerce-Anwendungen stellt sich dem Kunden häufig das Problem, dass das Verkaufsangebot zwar bereits sehr umfassend ist, das Auffinden des Artikels, der am besten den eigenen Präferenzen entspricht, dagegen sehr schwierig ist. In Warenhäusern steht dazu ein Verkaufsberater zur Verfügung, der versucht, die Interessen und Wünsche des individuellen Kunden einzuschätzen und ihn daraufhin zu beraten. Eine Anwendung, die einen solchen Verkaufsberater in elektronischen Systemen ersetzen soll, muss ebenfalls den Prozess zur Ermittlung der Präferenzen des Kunden dynamisch gestalten und individuell auf diesen zuschneiden.

Dies ist sowohl für die Suche in Produktkatalogen, als auch für komplexere Anwendungen, wie die individuelle Konfiguration eines Produktes relevant. Das Zuschneiden eines fertigen Produktes durch Anpassung optionaler Parameter stellt als Zwischenform ebenfalls eine sinnvolle Anwendung dar. In [CC00] wird außerdem gezeigt, dass auch das Angebot und die Konfiguration von Dienstleistungen ein interessanter Aspekt für zukünftige Systeme im E-Commerce Bereich sein könnte. In den nachfolgenden Kapiteln wird aber dennoch stets von „Produkten“ die Rede sein. Da es sich bei den Kunden um die Anwender eines Software-Systems handelt, verwenden wir den Ausdruck „Benutzer“.

Im nächsten Abschnitt wird eine kurze Einführung in die Evaluierung von Produkten mit Hilfe der multi-attributiven Objektbewertung (*MAUT*[2]) gegeben und auf die Probleme bei deren Anwendung hingewiesen. Dies dient als Basis des neuen Ansatzes, der dann konzeptionell und algorithmisch im darauffolgenden Kapitel beschrieben wird. Im letzten

[1] Die vorliegende Arbeit wurde unterstützt durch das IST-Programm der Europäischen Union (Vertrag: IST-1999-10688 CAWICOMS). Weitere Informationen sind verfügbar unter: `http://www.cawicoms.org`.
[2] MAUT ist ein Akronym für „Multi-Attribute Utility Theory“, was als multi-attributive Objektbewertung übersetzt werden kann.

Abschnitt wird ein Ausblick auf weitere damit verbundene Fragestellungen und Anwendungsmöglichkeiten gegeben.

2 Evaluierung von Produkten mit MAUT

Ein entscheidender Aspekt zur Adaption des Verhaltens eines Systems an die Interessen und Präferenzen des Benutzers ist die Repräsentation des Evaluierungsprozesses von Produkten. Die in dieser Arbeit verwendeten Ansätze basieren auf MAUT [WE86].

Bei MAUT werden die Präferenzen des Benutzers in Abhängigkeit von festgelegten Attributen des Produktes beschrieben. Jedes Attribut erhält von dem Benutzer ein *Gewicht* und eine *Bewertungsfunktion*. Das Gewicht eines Attributes drückt aus, wie stark dessen Bewertung relativ zu allen anderen Attributen in die Gesamtbewertung einfließt. Die Bewertungsfunktion beschreibt, wie das Attribut in Abhängigkeit seiner möglichen Werte evaluiert wird. Sei nun also $eval_a(l(a))$ die Evaluierung des aktuellen Wertes $l(a)$ eines Attributes a und sei g_a das Gewicht für dieses Attribut, dann ergibt sich die *Gesamtevaluierung* des Produktes P wie folgt:

$$eval(P) = \sum_{a \in A} g_a \cdot eval_a(l(a))$$

In obiger Formel ist A die Menge aller Attribute des Produktes und es gilt $\sum_{a \in A} g_a = 1$.

Mit der direkten Anwendung von MAUT stellt sich das Problem, dass es für den Benutzer sehr aufwendig ist, die Gewichtungen und Bewertungsfunktionen für alle Attribute zu spezifizieren. Dies ist einerseits häufig nicht im Interesse des Benutzers und kann andererseits auch nicht in allen Fällen von ihm bewerkstelligt werden, da hierzu ein hohes Domänenwissens vorausgesetzt wird [CP01].

Der Ansatz wird durch die Einführung sogenannter Interessensdimensionen, wie zum Beispiel *Sportlichkeit* oder *Zuverlässigkeit* des Produktes [SCH98], [JSS+95] erweitert. Man konstruiert die Dimensionen so, dass jede Dimension einen Nutzen bringt, dass also jeder Benutzer jeweils an einer hohen Bewertung aller Dimensionen interessiert sein sollte. Die Gesamtbewertung ergibt sich dann aus der Summe der Evaluierungen der einzelnen Dimensionen, wobei jeweils die Bewertung einer Dimension mit dem Interesse des Benutzers an dieser Dimension gewichtet wird. Definiert man nun die Abhängigkeit zwischen einer Interessensdimension und den für sie relevanten Attributen des Produktes innerhalb der Domäne fest, so genügt es also lediglich noch die Interessen des Benutzers an den Dimensionen zu modellieren. In der Praxis geht man geht hier in der Regel von der vereinfachten Annahme aus, dass die Sicht aller Benutzer auf die jeweilige Interessensdimension gleich ist, d.h. die Evaluierungsvorschrift der Dimension *Sportlichkeit* ist a priori fest definiert. Geht man von dieser Idee aus, so bieten sich zwei Anwendungsmöglichkeiten. Zum Einen kann ein Produkt bereits aufgrund sehr weniger Angaben, nämlich der Selbsteinschätzung des Benutzers von seinem Interesse an den einzelnen Dimensionen, bewertet werden. Zum Zweiten lässt sich aus der Spezifikation von Werten für verschiedene Produktattribute durch den Benutzer auf dessen Interessen schließen.

Der in dieser Arbeit vorgestellte Ansatz sollte nun zu mehr Flexibilität führen und dem Benutzer die Möglichkeit geben, in Abhängigkeit seines Wissensniveaus und seiner Interessen, Werte von spezifischen Detailattributen bis hin zu sehr abstrakten Parametern zu spezifizieren.

3 Produktevaluierung auf der Basis einer Parameter-Hierarchie

3.1 Parameter-Hierarchie

Um mit Produkten umgehen zu können, ist in E-Commerce-Systemen das konzeptionelle Domänenwissen in Form sogenannter Produktmodelle repräsentiert. Ein solches Modell kann von einer einfachen Auflistung der möglichen Attribute des Produktes bis hin zu einem Objektmodell, das als Grundlage für Konfigurationssysteme dient ([Stu97], [SW98]), reichen. Alle darin definierten Parameter können als Attribute bei der Anwendung von MAUT dienen, sie werden im folgenden als *Basis-Parameter* bezeichnet. Entscheidend ist hier, dass das zugrunde liegende System direkt Aussagen über diese Basis-Parameter machen kann, und auch deren Werte direkt spezifiziert werden können.

Die Basis-Parameter sind die Blätter der nachfolgend definierten *Parameter-Hierarchie*. Zusätzlich können nun auch *abstraktere Konzepte* definiert werden, die selbst wiederum neu eingeführte Parameter enthalten und diese zu einer Sinneinheit zusammenfassen. Für jeden in der Hierarchie definierten Parameter wird ein eigener Wertebereich spezifiziert.

Desweiteren werden Relationen eingeführt, welche als *Abstraktionsvorschriften* dienen. Sie beschreiben formal die Abhängigkeit zwischen dem abstrakteren Konzept und einem spezielleren Parameter und werden nachfolgend als *Generalisierungsfunktionen* bezeichnet. Sowohl die abstrakten Konzepte als auch die Generalisierungsfunktionen werden a priori für eine Domäne festgelegt.

Die auf diese Art definierte Parameter-Hierarchie ist Teil einer Ontologie, in der domänenspezifisches Wissen abgelegt wird. Neben der strukturellen Information enthält sie auch strategische Information, d.h. Beschreibungen von Inferenztechniken, mit denen die Interaktion mit dem Benutzer unterstützt werden kann.

In Abbildung 1 ist beispielhaft ein Ausschnitt aus einer Parameter-Hierarchie für die Auto-Domäne dargestellt. An den Blättern befinden sich die Basis-Parameter, wie zum Beispiel *Motorleistung*, *Verbrauch* oder *Anzahl an Lautsprechern*. Außerdem sind hier die zusätzlich eingeführten, abstrakten Konzepte *Motor*, *Auto* und *HiFi-Anlage* abgebildet. Neu eingeführte Parameter wie *Leistungsfähigkeit* oder das typische *Einsatz*gebiet des Wagens dienen dabei zur Spezifikation der abstrakten Konzepte.

Versucht ein System nun den Benutzer bei der Auswahl eines für ihn angemessenen Produktes aus einem Katalog oder bei der Konfiguration eines Produktes zu unterstützen, so muss das System die Präferenzen des Benutzers repräsentieren, d.h. es muss dessen Evaluierungsprozess modellieren. Verwendet man MAUT in diesem Kontext direkt, so müsste der Benutzer seine Präferenzen bezüglich jedes Basis-Parameters durch Spezifikation eines Gewichtes und einer Bewertungsfunktion ausdrücken. Bei Verwendung einer Parameter-Hierarchie kann der Benutzer auch stattdessen seine Präferenzen bezüglich abstrakterer Konzepte angeben. In unserer Beispieldomäne könnte es zum Beispiel sinnvoll sein, zwar Präferenzen zu Basis-Parametern wie *Hubraum* oder *Zylinderanzahl* direkt anzugeben, aber die *HiFi-Anlage* des Wagens nur durch eine Bewertung des entsprechenden *Komfort*s zu spezifizieren. Die durch den Benutzer spezifizierten Präferenzen bezüglich eines spezielleren Konzeptes überschreiben stets die auf der Basis seiner Angaben zu abstrakteren Konzepten berechneten Werte. So ist es in unserer Beispieldomäne zwar möglich, möglichst niedrige *Motorkosten* zu bevorzugen, aber dennoch eine hohe *Leistungsfähigkeit* bezüglich des Motors zu wünschen.

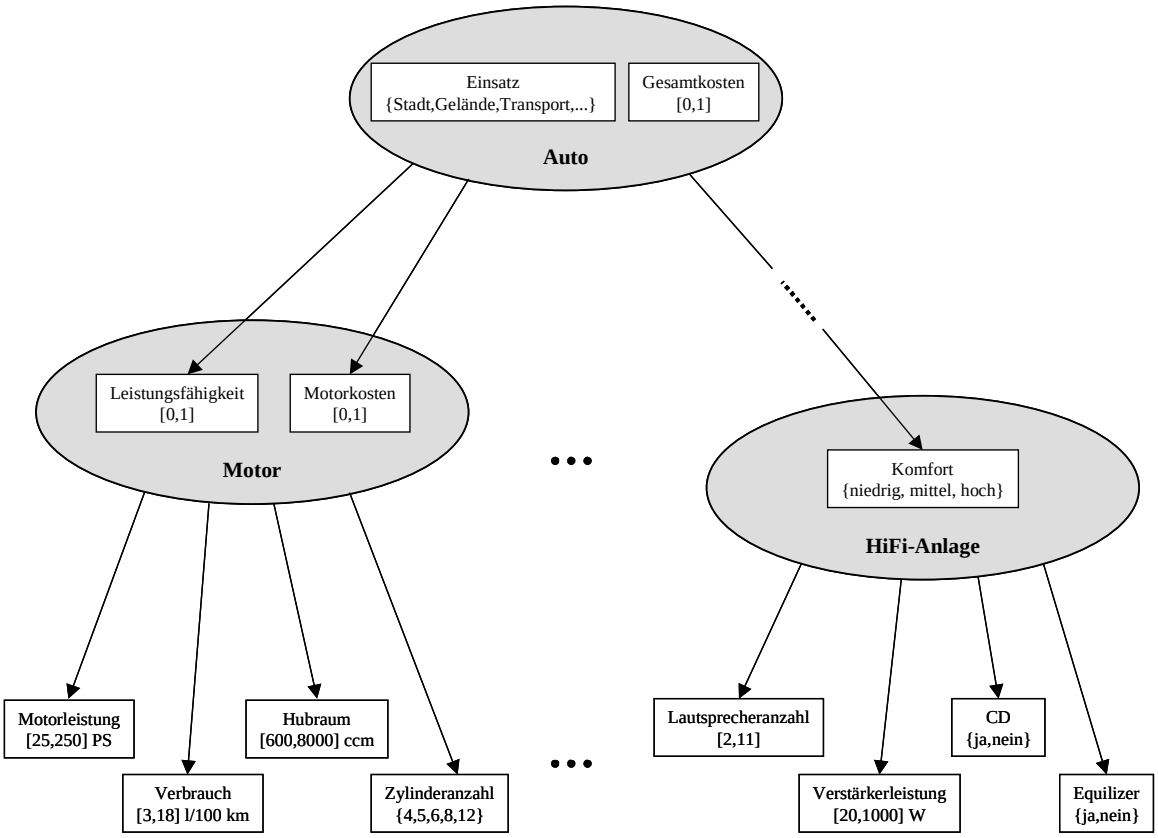


Abbildung 1: Ausschnitt einer Parameter-Hierarchie am Beispiel der Auto-Domäne

Es ist wichtig, darauf hinzuweisen, dass die Definition der Präferenzen durch den Benutzer in der Praxis stark vereinfacht gemacht werden kann, in dem der Benutzer zum Beispiel aus drei verschiedenen, festen Gewichtungen auswählt und das System dann automatisch über der Menge der Gewichte normiert. Die Bewertungsfunktion kann zum Beipiel einfach dadurch vom Benutzer beschrieben werden, dass er angibt, er wolle für einen bestimmten Parameter eher hohe Werte, woraus sich idealisiert eine steigende lineare Funktion ableiten ließe. Im Gegensatz solcher Präferenzen (*Soft-Constraints*) kann der Benutzer auch bestimmte Werte fest spezifizieren (*Hard-Constraints*), wobei man die entsprechenden Parameter dann bei der Anwendung von MAUT nicht mehr zu berücksichtigen braucht, da ihre Bewertung fix ist.

Da lediglich über Basis-Parameter im zugrunde liegenden System direkt eine Aussage getroffen werden kann und auch nur diese direkt spezifiziert werden können, ist eine Anwendung der in der Hierarchie definierten Relationen sinnvoll, um von den Präferenzen bezüglich abstrakter Konzepte auf die Präferenzen für Basis-Parameter zu schließen. Ein entsprechender Algorithmus wird im folgenden beschrieben.

3.2 Produktevaluierung

Gegeben sei die Menge P aller in der Hierarchie definierten Parameter sowie die Menge C der abstrakten Konzepte, mit $c \subset P$, $\forall c \in C$. In unserem Beispiel kapselt also das abstrakte Konzept *Motor* die beiden Parameter *Motorkosten* und *Leistungsfähigkeit*. Sei nun außerdem die Relation $r_{c,p}$ vom Konzept c zum Parameter p gegeben.

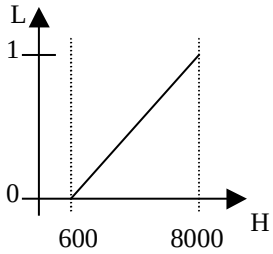
Die Relation $r_{c,p}$ besteht aus einer Menge von Generalisierungsfunktionen, die wie folgt definiert sind:

$$r_{c,p} = \{generalize_{q,p}(D_p) \mid q \in c\}$$

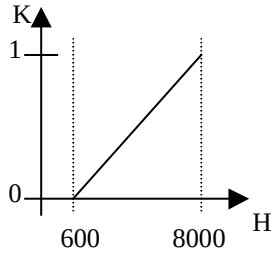
Das heißt für jeden Parameter $q \in c$ enthält $r_{c,p}$ eine Generalisierungsfunktion, die beschreibt, wie man aus dem Wert des spezifischeren Parameters p den Wert des abstrakteren Parameters q berechnet:

$$l(q) = generalize_{q,p}(l(p))$$

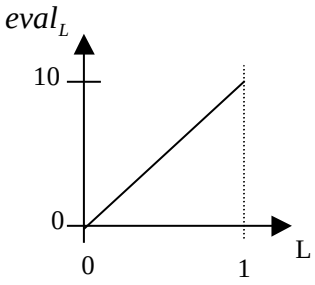
Im folgenden Beispiel gehen wir davon aus, dass der Benutzer seine Präferenzen für das abstrakte Konzept *Motor* spezifiziert. Das heißt, er gibt seine Präferenzen bezüglich der Parameter *Leistungsfähigkeit (L)* und *Motorkosten (K)* an, nicht aber bezüglich der Basis-Parameter *Motorleistung*, *Verbrauch*, *Hubraum* und *Zylinderanzahl*. In Abbildung 2 a) und b) ist dargestellt wie das Verhältnis dieser beiden Parameter zum Basis-Parameter *Hubraum (H)* definiert wurde, wobei hier zum Beispiel gelten soll, dass mehr *Hubraum* zu einer höheren *Leistungsfähigkeit* führt.



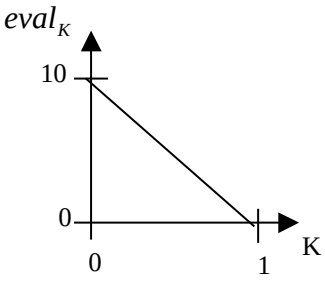
a) $generalize_{L,H}$: Abhängigkeit zwischen L und H



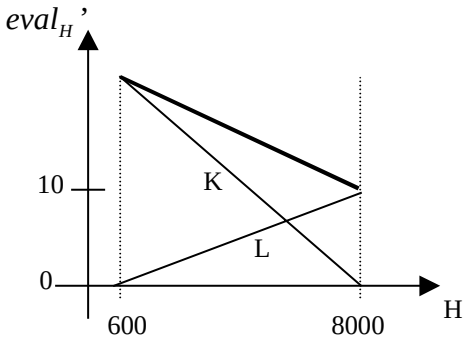
b) $generalize_{K,H}$: Abhängigkeit zwischen K und H



c) Evaluierung von L durch den Benutzer



d) Evaluierung von K durch den Benutzer



e) Resultierende Evaluierungsfunktion für H

Abbildung 2: Berechnung einer Evaluierungsfunktion

Der Benutzer gibt nun seine Präferenzen an, indem er die Bewertungsfunktionen $eval_q(l(q))$ und die Gewichte g_q für alle Parameter $q \in c$ definiert, wobei die Gewichte g_q noch nicht über der Menge aller Gewichte normiert sind. Wir gehen im folgenden davon aus, dass das Ergebnis der Evaluierung eines Parameters stets auf einer Skala zwischen 0 und 10 liegt, wobei 10 sehr gut und 0 sehr schlecht ist. In Abbildung 2 c) und d) sieht man also, dass der *Motor* vom Benutzer als gut bewertet wird, wenn er sehr leistungsfähig ist und einen geringen Kostenaufwand verursacht.

Um die Bewertungsfunktion für den spezifischeren Parameter p zu erhalten, werden zunächst die gewichteten Bewertungsfunktionen der abstrakten Parameter summiert:

$$eval'_p(l(p)) = \sum_{q \in c} g_q \cdot eval_q(generalize_{q,p}(l(p)))$$

In Abbildung 2 e) ist $eval'_H$ fett dargestellt, unter der Annahme, dass $g_L = 1$ und $g_K = 2$ ist.

Da der Wertebereich einer Evaluierungsfunktion stets im Bereich zwischen 0 und 10 liegen muss, wird diese Funktion anschließend normiert:

$$eval_p(l(p)) = \frac{eval'_p(l(p))}{g'_p}, \text{ mit } g'_p = \frac{1}{10} \cdot \max_{l(p)}(eval'_p(l(p)))$$

Das Gewicht des Parameters p lässt sich dann wie folgt berechnen:

$$g_p = \frac{g'_p}{\left|\{v \mid \exists r_{c,v} : r_{c,v} \neq \varnothing\}\right| > 0}$$

Dieses Gewicht muss später noch über der Menge aller Basis-Parameter normiert werden.

Wir haben nun also gezeigt, wie man eine Generalisierungsfunktion anwenden kann, um aus einer Präferenz für ein abstrakteres Konzept die Präferenz bezüglich eines spezifischeren Parameters zu berechnen. Wird dies stets bis zu den Blättern der Hierarchie, also den Basis-Parametern, durchgeführt, so kann MAUT direkt angewendet werden.

4 Ausblick und Ziele

Die vorliegende Arbeit stellt einen MAUT-basierten Ansatz zur Evaluierung von Produkten auf der Basis einer Parameter-Hierarchie dar. Die Zielsetzung besteht darin, den Interaktionsprozess an den Benutzer anzupassen, d.h. stets den für die aktuelle Situation angemessenen Parameter aus einer gegebenen Hierarchie auszuwählen, um den Benutzer seine Präferenzen bezüglich dieses Parameters spezifizieren zu lassen. Diese Auswahl wird von drei Aspekten abhängig sein:

- *Expertise des Benutzers*: Falls es dem Benutzer aufgrund mangelnden Domänenwissens nicht möglich ist, seine Präferenzen bezüglich eines bestimmten Parameters zu beschreiben, so muss ihm ein abstrakterer Parameter präsentiert werden.
- *Interesse des Benutzers*: Je stärker der Benutzer an bestimmten Aspekten des Produktes interessiert ist, desto eher ist er bereit sehr viele, detaillierte Angaben zu machen.
- *Nützlichkeit des Parameters*: Um den Suchraum zu reduzieren, sollten bei der Auswahl des nächsten Parameters solche Parameter bevorzugt werden, die den Wertebereich anderer Parameter möglichst stark einschränken.

Dem Benutzer sollte außerdem stets die Möglichkeit gegeben sein, selbst durch die Abstraktionshierarchie zu navigieren, um sich so für das seiner Meinung nach

angemessene Abstraktionsniveau einer Fragestellung zu entscheiden [FW96]. Auf der Basis des beobachteten Verhaltens des Benutzers, sollte eine Applikation die Interessen und die Expertise des Benutzers modellieren und den Befragungsprozess so dynamisch während der Interaktion anpassen.

Eine weitere interessante Anwendungsmöglichkeit könnte der gezeigte Evaluierungsansatz in einem System finden, das auf dem Kandidat-Kritik Modell beruht [LHL97]. Dabei werden dem Benutzer verschiedene Produkte präsentiert und der Benutzer kann spezifizieren, welche Aspekte des präsentierten Produktes nicht seinen Vorstellungen entsprechen. Die vorgestellte Parameter-Hierarchie könnte dem Benutzer die Möglichkeit von sehr allgemeiner bis hin zu sehr spezifischer Kritik bieten, je nachdem, wie zufrieden er bereits mit dem gezeigten Produkt ist.

Die erfolgreiche Modellierung der Expertise des Benutzers innerhalb der Hierarchie wird entscheidend zum Erfolg des Modells beitragen. Hierzu müssen noch Modelle entwickelt und umgesetzt werden.

Die Auswahl passender Produkte aus einem Katalog wird zunächst mit dem vorgestellten Modell leichter zu realisieren sein, da bereits ein Verfahren zur Bewertung der einzelnen angebotenen Artikel definiert wurde und diese dann nur noch abhängig von dieser Bewertung miteinander verglichen werden müssen. Die Umsetzung bei Konfigurationssystemen wird sich voraussichtlich aufgrund des Problems der Optimierung als deutlich komplexer herausstellen.

Literaturverzeichnis

[CC00] CAWICOMS Consortium (2000). *Deliverable D01 – Requirements, Application Scenarios, Overall architecture, and Test Specification.*

[CP01] Chin, D., and Porage, A. (2001). *Acquiring User Preferences for Product Customization*. In Proceedings of the 8th International Conference, UM 2001.

[FW96] Faltings, B., and Weigel, R. (1996). *Abstraction Techniques for Configuration Systems*, AAAI, Fall Symposium.

[JSS+95] Jameson, A., Schäfer, R., Simons, and J. Weis. T. (1995). *Adaptive provision of evaluation-oriented information: Tasks and techniques*. In C.S. Mellish (ed.), Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pp. 1886 – 1893.

[LHL97] Linden, G., Hanks, S., and Lesh, N. (1997). *Interactive Assessmentn of User Preference Models: The Automated Travel Assisstant*. In Proceedings of the 6th International Conference, UM 2001

[Sch98] Schäfer, R. (1998). *Benutzermodellierung mit dynamischen Bayes'schen Netzen als Grundlage adaptiver Dialogsysteme*, Ph.D. thesis, Universität des Saarlandes, Saarbrücken.

[Stu97] Stumptner, M. (1997). *An overview of knowledge-based configuration*, in AI Communications, 10(2).

[SW98] Sabin, D., and Weigel, R. (1998). *Product configuration frameworks – a survey*, in Special Issue on Configuration of IEEE Intelligent Systems, 13(4), pp. 42-49.

[WE86] Winterfeld, D. von and Edwards, W. (1986). *Decision Analysis and Behavioral Research*. Cambridge, England: Cambridge UniversityPress.