IMPROVING USER CONFIDENCE IN DECISION SUPPORT SYSTEMS FOR ELECTRONIC CATALOGS

THÈSE N^{O} 3737 (2007)

PRÉSENTÉE LE 20 DÉCEMBRE 2007

À LA FACULTÉ INFORMATION ET COMMUNICATIONS

Laboratoire d'intelligence artificielle

SECTION D'INFORMATIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

David PORTABELLA-CLOTET

Ingénieur en télécommunications, Université Polytechnique de Catalogne, Espagne
et de nationalité espagnole

acceptée sur proposition du jury:

Prof. E. Telatar, président du jury
Dr M. Rajman, directeur de thèse
Prof. I. Kopecek, rapporteur
Prof. M. López-Sánchez, rapporteur
Dr P. Pu Faltings, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Lausanne, EPFL
2008

Abstract

Decision support systems for electronic catalogs assist users in making the right decision from a set of possible choices. Common examples of decision making include shopping, deciding where to go for holidays, or deciding your vote in an election. Current research in the field is mainly focused on improving such systems in terms of *decision accuracy*, i.e. the ratio of correct decisions out of the total number of decisions taken. However, it has been widely recognized recently that another important dimension to consider is how to improve *decision confidence*, i.e. the certainty of the decision maker that she has made the best decision.

We first review multi-attribute decision theory –the underlying framework for electronic catalogs– and present the state-of-the-art research in e-catalogs. We then describe objective and subjective measures to evaluate such systems, and propose a system baseline for achieving more accurate and meaningful comparative evaluations.

We propose a framework to study the building of decision confidence within the *query-feedback search* interaction model, and use it to compare different types of system feedback proposed in the literature. We argue that different types of system feedback based on constraints (e.g. *conflict* and *corrective* feedback), even if not novel as such, can be combined in order to improve decision confidence. This claim is further validated by simulations and experimental evaluation comparing constraint-based feedback to ranked list feedback.

Keywords: decision support tools, human computer interaction, electronic catalogs, e-commerce, databases, multi-attribute decision theory, preference elicitation, decision accuracy, decision confidence.

Résumé

Les systèmes d'aide à la décision assistent les utilisateurs à trouver l'objet désiré à partir d'un ensemble existant d'objets. Des exemples usuels de décision incluent des achats, décider où aller en vacances, ou décider pour qui voter aux élections. Les études précédentes ont avant tout visé à améliorer tels systèmes dans leur précision de décision, c'est-à-dire la proportion de décisions correctes à partir d'un nombre total de décisions. Cependant, récemment la communauté a reconnu qu'une autre dimension importante à considérer est comment améliorer la confiance de décision, c'est-à-dire la certitude qu'a le preneur de décision d'avoir pris la meilleure décision possible.

Tout d'abord nous examinons la théorie de décision multi-attribut –le cadre général des systèmes d'aide à la décision– et présentons l'état de l'art de la recherche sur les catalogues électroniques. Ensuite, nous présentons des mesures objectives et subjectives pour l'évaluation de tels systèmes, et proposons un système de base pour atteindre des évaluations comparatives plus précises et significatives.

Nous proposons un cadre pour étudier la création de la confiance de décision dans le modèle d'interaction question-explication, et l'utilisons pour comparer différents types d'explications trouvés dans la littérature. Nous argumentons que différents types d'explications basés sur contraintes (l'identification de conflits et explications correctives, par exemple), même s'ils ne sont pas nouveaux comme tels, peuvent être combinés pour améliorer la confiance de décision. Cette argumentation est encore validée avec des simulations et une étude d'utilisateur comparant les explications basées sur contraintes avec celles basées sur un classement.

Mots clés : systèmes d'aide à la décision, interaction homme-machine, catalogues électroniques, e-commerce, bases de données, théorie de décision multi-attribut, extraction des préférences, précision de décision, confiance de décision.

Acknowledgments

It is said that a PhD cannot be a PhD without a crisis. Whether this is true or not, I can say that I have very much fulfilled this pre-requisite, after which I started enjoying my research.

This work was supported by the Swiss National Science Foundation (grant no. 20020-105455). This thesis has benefited from the help of many researchers and colleagues. First I would like to thanks my thesis supervisor, Dr. Martin Rajman, who gave me the opportunity to enter into research. With him, I have learnt about research, but also about structuring ideas, looking at positive sides of issues, writing, supervising and motivating students and ethics. I also remember having meetings until midnight or even on Sundays. I acknowledge his patience during my deep 2005 crisis and his support in the turbulent last months.

I am very grateful to the members of the jury: Prof. Maite Lopez, Prof. Ivan Kopecek and Dr. Pearl Pu, as well as the president of the jury, Prof. Emre Telatar. They provided me with very interesting feedback, also through the course of writing my thesis.

My gratitude also goes out to the members of the decision group at EPFL for their helpful feedback, in particular to Dr. Pearl Pu who pointed out to me that confidence and trust are important in e-commerce. I would like to acknowledge Paolo Viappiani who first provided me with a catalog of apartments to test my initial ideas. Also, thanks to Jiyong Zhang and Li Chen for our fruitful interaction on decision support systems, and to Vincent Schickel for the same reason and also for his devotion as our network administrator.

In June 2006 I participated in a great hands-on workshop in the Black Forest organized by Prof. Pierre Dillenbourg where I meet PhD students in cognitive science who helped me with the cognitive framework, the experimental design and the analysis of the statistical results of this thesis. Among them were Dorothe Kienhues, Johannes Gurlitt, Dr. Katharina Scheiter and Sabine Hauser from Germany, Sangin Mirweis, Enrico Costanza, Nicolas Nova, Dr. Jean-Cedric Chappelier from EPFL and Agnes Lisowska from UNIGE. Thanks also to Mark Meagher for his valuable comments and proofreading parts of this dissertation. Special thanks go to Dr. Patrick Jermann, at EPFL-CRAFT, for his patience and availability in this area. Thanks also to Kamni Gill for proofreading the whole dissertation.

I am amazed by the great atmosphere of support among researchers, even without personally knowing each other. In most cases, I have received a helpful answer with very short delays when requesting clarification of an author's publication or even for more general questions. I would especially like to thank Darius Braziunas from the University of Toronto, whom I still have not met in person, for his patience in answering my questions on decision theory and who revised related sections of this dissertation. I also would like to thank the anonymous reviewers at the AAAI and ECAI conferences for their useful feedback. I am also grateful to the Swiss company *Comparis.ch* for sharing their real database of second-hand cars with the research community.

I also would like to thanks to all the colleagues who helped me in overcoming my 2005 crisis: Radu Jurca, Dr. Michael Ignaz Schumacher, Dr. Fabio Porto, Prof. Francesc Comellas, Prof. Carles Sierra, José Diego De la Cruz, Vinh-Toan Luu and many others.

Thanks to the colleagues of the predoctoral school, the members of EPFL-LIA and the lab's secretary, Mme Decrauzat, for her devotion. Also, thanks to the computer gurus at EPFL-IC: Benjamin Barras and Stéphane Ecuyer, Sébastien Cevey, Leander Eyer and Charles François Rey. Thanks also to the undergraduate students that have collaborated in my projects: Alex Prudencio, Vlad Tanasescu, Muriel Gummy, Nadir Habdi, Vincent Etter, Baptiste Dubuis, Asheesh Gulati and Laetitia Henriot, as well as Dr. Ion Constantinescu, Mirek Melichar and Marita Ailomaa for co-supervising.

During this long process, I have enjoyed activities such as skiing, karting, cycling, jogging, hiking, squash, language tandems and *macumbazos* with my friends here in Lausanne: Dr. Rodrigo Garcia, Marcos Perez, Josep Garriga, Dr. Francesc Font, Dr. Guillermo Barrenetxea, Raül Rabionet, Stefan de la Roche, Genet Philippe, Vincent Lausselet and many others. I will never forget Dr. Santiago Macho and his funny *performances*.

I would like to especially thank two guys who have been beside me at every stage since the very beginning of this adventure, as researchers, with inspiring discussions, valuable feedback and proofreading, and as friends: Dr. Sergio Mena and Dr. Vincenzo Pallota.

Thanks also to my friends in Manresa for the shared activities and support: Aleix Estruch, Monica Izquierdo, Jordi Coromina, David Verde, Angels Xarpell, my colleagues at the *Universitat Politècnica de Catalunya* (UPC), my family and especially my brother, Marc. Specials thanks to the Easyjet and Skype companies for mitigating the distance factor.

I am very grateful to my parents, Fèlix and Dolors, who worked hard to give us the possibility of going to university and for their unconditional support and love.

Finally, thanks to my girlfriend, Núria, for your help and proofreading of this work, and more importantly, for your love.

Contents

List of Figures

List of Tables

List of Examples

Glossary and Acronyms

Note: Italicized words have a separate glossary entry.

Attribute: An abstraction of a characteristic of an *item*, e.g. “price”, “color” or “delivery time”. See Section 2.2.

Attribute utility: A number that expresses the desirability of an *attribute-value pair* (assuming *decision theory under certainty* and *additive independence* assumption). See Section 2.3.

Attribute utility function: A function that maps an *attribute value* to an *attribute utility* for a given attribute (assuming *decision theory under certainty* and *additive independence* assumption). See Section 2.3.

Attribute value: The value of an *attribute* for a given *item*. For instance, the value of the “brand” attribute of a particular car item can be “BMW”. See Section 2.2.

Attribute-value pair: Given an *item* and an *attribute*, it is the pair attribute and value of that attribute for the given item. See Section 2.2.

Catalog: A set of homogeneous *item descriptions*. By homogeneous, we mean that the item descriptions share the same set of predefined attributes. See Section 2.2.

Catalog decision problem: The multi-attribute *decision problem* of finding the *item* in the *catalog* that best satisfies the preferences of the decision maker (assuming *decision theory under certainty*). See Section 3.3.

CBTE: Confidence building tradeoff explanation**.** A type of *system feedback* that presents the query tradeoffs in terms of conflicts and relaxations that can potentially contribute to increase *decision confidence*. See Section 3.4.

Certainty (decision theory under certainty): The situation in which the consequences (items) of actions are certain, and thus an action is equivalent to simply choosing an item from the set of available items. See Section 2.3 for more about it. Note however, that we do not assume that users have a complete knowledge of their preferences. Actually, in most of the times users construct their preferences during the decision process.

Choice: See *decision.*

Collaborative catalog system: A *catalog system* that attempt to predict *items* that a user may be interested in based on the similarity of her *profile* with the profiles of many other users.

Confidence: See *decision confidence.*

Confidence building tradeoff explanation: See *CBTE.*

Constraint: See *parametric constraint.*

Constructive preferences: The concept that decision makers begin the decision process with a vague preference model and augment and revise it as they learn more about the available items in the catalog. See Section 2.4.

Correct decision: A *decision* is correct if it corresponds with the *desired item.* See also *decision accuracy.* See Section 2.6.

Criteria: See *query.*

Criterion: See *attribute.*

Database: see *catalog.*

Decision: The *item* that the user (the *decision maker*) chooses at the end of the *decision process.* See also *correct decision.* See Section 2.3.

Decision accuracy: The ratio of *correct decisions* out of the total number of *decisions.* It is therefore an objective measure. See Section 2.6.

Decision aid: See *decision support system.*

Decision analyst: A human expert in *decision theory* that helps a *decision maker* to take a *decision* through a *decision process.* See Section 2.3.

Decision confidence: The certainty of the decision maker that she has made the best decision. See Section 3.3.

Decision confidence in query-feedback search: See *DECOFE.*

Decision justification: An overall explanation *internalized* by the user (the *decision make*r) that justifies the *decision.* See Section 3.3.

Decision maker: A human that needs to makes a *decision* throughout a *decision process.* See also *catalog decision problem.* See Section 2.3.

Decision problem: A problem in where a *decision maker* needs to make a *decision* through a *decision process.* See also *catalog decision problem.* See Section 2.3.

Decision process: A process in which a *decision maker* and a *decision analyst* or *decision support system* interact in order to make a *decision* (solve a *decision problem*). See Section 2.3.

Decision support system (DSS): A computer program that helps a *decision maker* to make a *decision* through a *decision process.* See Section 2.3.

Decision support system for electronic catalogs: See *DSSEC.*

Decision theory: A discipline that has its roots in operations research, psychology, economics and artificial intelligence and studies how a *decision process* is done or should be done. See Section 2.3.

Desired item: The *item* from the *catalog* with the highest *utility* for the user. See Section 2.3.

DECOFE: A framework to study the building of *decision confidence* in the *query-feedback search interaction* model. See Section 3.3.

DSSEC: A content-based *decision support system* that helps *decision makers* to select an *item* from a *catalog* (assuming *decision theory under certainty*). See Section 2.1.

Electronic catalog: See *catalog.*

Electronic commerce: Business transactions which are conducted over an electronic network such as the Internet. See also *on-line store.*

Failing query: See *over-constrained query.*

Feedback: See *system feedback.*

Global relaxation set: See GRS.

GRS (Global relaxation set): A type of *system feedback* that provides a set of options for solving all the conflicting objectives in a query at once, and thus it represents a clear tradeoff question. See Section 3.4.

Hard constraint: See *parametric constraint.*

Information visualization Seeking (VIS): A *query navigation system* in which the *items* are displayed using a visual representation. See Section 2.4.7.

Internalize: To internalize a system feedback is to incorporate that feedback into one's mental model. Concretely, if a user has internalized a given feedback, it means that she has understood it, that she can remember and process it later. See Section 3.3.

Interaction session: See *decision process.*

Interaction cycle: An iteration step in which the user makes a *query* and the system returns a *feedback*. See Section 2.4.

Item: A product or service, such as a movie, car or insurance. See Section 2.2.

Item description: A set of *attribute-value pairs* that describe an *item*. See Section 2.2.

Local relaxation set: See *LRS*.

LRS (Local relaxation set): A type of *system feedback* that tells how much the user needs to relax a single *constraint* in order to solve a given *conflict*. See Section 3.4.

MCS (Minimal Conflict Set): A type of *system feedback* that shows precisely which parts of the *query* are conflicting. See Section 3.4.

Minimal conflict set: See *MCS*.

Objective: Something toward which interest is directed. An example objective can be to "minimize price". See Section 2.3.

On-line store: An *electronic commerce* application from which users (*decision makers*) can buy products or services (*items*). Also called "Internet shop" or "Webshop". See Section 2.1.

Over-constrained query: A *query* (as a set of *parametric constraints*) that no *item* in the *catalog* satisfy. See Section 3.5.1.

Parametric constraint: A condition about an *item* that must be satisfied. We assume the condition to be defined over a single *attribute* (we do not consider conditional constraints). It differs from an objective in that it is either achieved or not. An example constraint can be: "price less than 2'000 Euros". See Section 2.4.

Parametric weighted additive strategy: See *PWADD*.

Preference: In the context of decision theory, a preference is an order relation of the set of available *items* in a catalog, and it can be expressed by a *utility function*. However we use here the term *preferences* with its more general colloquial meaning.

Product Search Engine: See *DSSEC*.

Profile: Information about a user, such as a history of items she has watched or purchased in an *on-line store*, or a *rating* for a set of *items*.

PWADD: Parametric weighted additive decision strategy: the approach of using *parametric attribute utility functions* with the weighted additive (WADD) strategy. See Section 2.4.2.

Query: A set of either *parametric constraints* or *parametric utility functions*. An example of query can be “price less than 2’000 Euros, four doors and with airbag”. See Section 2.4.

Record: an *item description* stored in a database. See Section 2.2.

Selection: See *decision*.

System feedback: A feedback produced by a DSSEC for a given user *query*. Examples of types of system feedback: matching list, ranked list, diversity, suggestions, conflict, corrective and visualization explanations. See Section 2.4.

Target item: See *desired item*.

Trust: The belief of a customer that she can rely on the assistance of a vendor, or the willingness to depend on a vendor. See Section 2.5.

User: See *decision maker*.

User interface: The means by which a user and a software application communicate. The most common methods for such communication are text, buttons and menus by using a screen, a keyboard and a mouse. See Section 2.1.

Utility: A number that expresses the desirability of an *item* (assuming *decision theory under certainty*), typically between 0 (least desired) and 1 (most desired). See Section 2.3.

Utility function: A function that maps an *item* to a *utility* (assuming *decision theory under certainty*). In the *weighted additive strategy*, it is a weighted sum of *attribute utility functions*. See Section 2.3.

Value: See *attribute value*.

WADD: A compensatory decision strategy. See Section 2.3.1.

Weighted additive decision strategy: See *WADD*.

Weight: A number that expresses the importance of an *attribute utility* associated to an *attribute* (under the *additive independence* assumption), typically between 0 (not important) and 1 (very important). See Section 2.3.1.

Chapter 1

Introduction

1.1 Context of the research

With the exponential growth of Internet and the **World Wide Web**, individual, companies, and organizations in general are able to communicate in an instantaneous and ubiquitous way. This enabled the development of a wide range of applications such as e-commerce, publishing information, auctions and social networks.

One of the most important applications nowadays is **electronic commerce (e-commerce)**. E-commerce can be defined as the automation of commercial transactions when using computer and communication technologies [94]. In fact, its importance is apparent when we see to which extent retail e-commerce has increased since the introduction of World Wide Web browsers.

Retail vendors usually offer a large set of their products and services in **electronic catalogs** and thus, comparing products from different vendors has become much faster for the consumers than physically going to traditional shops or reading paper catalogs. However, the speed of Internet has made the consumer impatient [96]. They expect an even higher level of performance from e-commerce applications, especially for those processes that could be automated like comparing different products. There is a new type of on-line companies that has been specifically created to satisfy this need. They aggregate the products present in the catalogs from different vendors and allow users to compare all of them within a single web site. This option is much faster than visiting different individual vendors sites.

However, consumers are often overwhelmed by so many choices, sometimes many more than they had expected to find. It is now much more difficult to commit to a choice, compared to going to a traditional shop where there is a limited number of products and an attentive sale assistant. E-commerce is far more impersonal, anonymous and automated [79] and thus it is not easy for the user to trust the e-vendor. Trust is a complex concept that depends on, among others, the vendor's integrity, security, social presence and competence as perceived by the consumer. Thus, a booming area for further development in e-commerce is to design strategies that increase user's trust [78, 79, 81, 85, 86, 88, 64, 89].

Decision support systems (DCS) are systems that help decision makers to make good decisions. The general underlying framework behind such systems is *decision theory*, a discipline has its roots in operations research, psychology, economics and artificial intelligence. DSS can be applied to many areas such as shopping, deciding where to go for holidays, or deciding your vote in an election.

Decision support systems for e-catalogs (DSSECs) are decision support systems commonly used in e-commerce that help decision makers to select a product from a catalog[1], and it is the applicative focus of the thesis. Thus, DSSECs can be seen as intelligent tools that guide consumers in finding the right choices.

Recommender systems (RSs) are systems that attempt to predict products that a user may be interested in, given some information about his profile. Often, a RS is implemented as a collaborative filtering algorithm, which attempts to predict items that a user may be interested in based on the similarity of her *profile* with the profiles of many other users. Collaborative filtering differs from DSSECs (content-based systems) in that they don't necessarily have a description of the products in terms of attributes, and make recommendations such as "people that bought this product also bought product X". Consider, for example, a consumer looking for a car. He would probably use a DSSEC. However, once she has committed to a given car, a RS could recommend him to also buy a product to clean the car based on past experience from similar users.

Previous research in DSSECs has focused in improving decision accuracy. We believe that increasing decision confidence, i.e. the certainty of the decision maker that she has made the best decision, is also an important property for DSSECs. The research on decision confidence for DSSECs is quite recent and has mainly focused on carrying out experiments in order to understand the antecedents and benefits of decision confidence, rather than studying how to improve it.

In Section 1.2 we justify the importance of improving decision confidence in DSSECs. In Section 1.3 we provide an illustrative example of a consumer who wishes to buy a used car, comparing the normative decision theory approach and a more natural vendor-consumer conversation. This example illustrates many of the ideas behind DSSECs and it is going to be referred throughout this thesis. Finally, in Section 1.4 we outline this thesis and contributions.

[1]In decision theory, the consequences of actions are generally uncertain. For instance, a consumer may decide between buying a photo camera either in a traditional store or in some not fully-reliable auction website for a cheaper price, with the uncertainty that she may not receive the product. However, DSSECs focus on decision under certainty, i.e. situations where the consequences of the actions are certain.

1.2 Motivation

Usually, people need to feel confident before committing to a decision. *Decision confidence* can be defined as the certainty of the decision maker that she has made the best decision. Decision confidence can be beneficial in several ways. First, having a good decision confidence helps to finally commit to the decision. Consider, for instance, a consumer who is hesitating between two cars to purchase. Even if both of them are very good choices, being unable to commit to one of them may be counterproductive for the consumer, as she might eventually never buy a car.

This problem is especially important for *e-commerce* due to the fact that e-commerce is far more impersonal, anonymous and automated than traditional commerce. According to [94], only 1.6% of visits to an e-commerce site result in a purchase, and two-thirds of the shoppers who get as far as putting items in a virtual shopping cart abandon the process before checking out. As it already has been shown in [90, 63], this low purchase rate can be significantly raised by improving decision confidence, as it positively affects the intention to buy.

Also, the consequence of a particular decision involving a group of people may not depend on the decision itself, but on the degree of confidence held by the decision maker and communicated to the others [76]. In these cases, raising the decision maker's conviction of doing the right thing can be a desirable property [77].

Previous studies concerning DSSECs have mainly focused on improving *decision accuracy*, i.e. the ratio of correct decisions out of the total number of decisions. See [27] in 1978, [31] in 1992, [33] in 1997 and [62] in 2006 for some representative examples. In this perspective, most of the research work has concentrated on the design of different types of *system feedback* supporting hidden user preference elicitation and leading to higher decision accuracy. However, increasing decision confidence is also an important property for DSSECs. Unfortunately, the research on decision confidence for DSSECs is quite recent (with some preliminary experiments in 2000 [35]) and, with the notable exception of Pearl Pu et al since 2004 [53, 56, 63], has mainly focused on carrying out experiments in order to understand the antecedents and benefits of decision confidence, rather than understanding the direct effect of different types of DSSECs on decision confidence. Therefore, the main motivation of this thesis is to provide further insights in the process of building decision confidence in DSSECs.

1.3 An illustrative example

1.3.1 Problem description

Examples help to improve the readability of ideas and concepts. We therefore provide a decision problem below that will be referred to throughout this thesis in order to explain concepts and ideas.

Consider that a consumer wishes to buy a used car and assume that she has a catalog with all the available cars. The problem is to select the best one according to the user's preferences. In order to get a more realistic example, we have contacted a company[2] that aggregates the inventory of used cars from several concessionaries in Switzerland. They have provided us with a list of 61443 cars, in which 7924 cars are described with 35 attributes.

To keep the example useful and tractable, we have selected forty cars and slightly changed some attribute values to better illustrate the different concepts developed in this thesis. All used cars are described by two continuous attributes (price and mileage), one nominal attribute (color) and one boolean attribute (airbag option). Our example catalog is shown in Example 1.1.

The decision problem consists in finding the one best used car from the forty available according to the consumer's preferences. The normative way to achieve this goal is to apply prescriptive decision theory, which studies how the decision process should be done in order to make a good decision. An example of such an approach is provided in the next section. However, another way to reach decisions is to rely on a more natural conversation between vendors and consumers. Such approaches are studied by descriptive decision theory. Again, an illustrative example is given in Section 1.3.3. Decision support tools for e-catalogs are designed to take into account both prescriptive and descriptive decision theory.

1.3.2 Solving the example by applying decision analysis

Most decision support tools for e-catalogs are based on decision theory. *Prescriptive decision theory* studies how the decision process should be in order to make good decisions. Decision analysis is the practical application of prescriptive decision theory, and aims at finding procedures and tools to help people make better decisions. As most consumers are not experts in decision theory, a decision process is typically supported by an expert taking the role of *decision analyst* or a software tool designed to be a *decision support system*.

[2]We are grateful to Comparis.ch for sharing an excerpt of their database. See Appendix 2 for more information about this database.

Example 1.1 Illustrative example. A catalog with forty used cars. Each car is described by two continuous attributes (price in Euros and mileage in km), one nominal attribute (color) and one boolean attribute (airbag option). The price is specified in Euros and the mileage in km.

id	price	mileage	color	airbag	id	price	mileage	color	airbag
#1	2666	76000	blue	no	#21	12338	10200	blue	yes
#2	3348	78000	blue	no	#22	13268	11000	blue	yes
#3	4216	79000	red	no	#23	13578	65200	black	yes
#4	4836	197000	red	no	#24	14074	62000	blue	yes
#5	5890	115000	blue	no	#25	14570	78000	black	yes
#6	6448	55700	black	yes	#26	14632	72000	blue	yes
#7	6450	97500	blue	no	#27	14880	67000	red	yes
#8	6510	76500	red	yes	#28	16052	37500	blue	yes
#9	6515	76400	red	yes	#29	16678	71000	black	yes
#10	9238	17300	blue	yes	#30	16678	45400	black	yes
#11	9858	113000	blue	yes	#31	17050	21000	black	yes
#12	9858	15700	black	yes	#32	17918	57700	black	yes
#13	10292	71000	white	yes	#33	18290	10500	black	yes
#14	10416	96900	black	yes	#34	19778	39000	black	yes
#15	10521	76650	black	yes	#35	20398	107000	black	yes
#16	11036	16700	black	yes	#36	20770	10199	blue	yes
#17	11346	62000	black	yes	#37	24118	36000	blue	yes
#18	11718	69000	black	yes	#38	24738	42000	black	yes
#19	11718	38100	blue	yes	#39	28830	20500	black	yes
#20	12338	20810	black	Yes	#40	30876	99000	black	yes

Let us illustrate such a decision process using the catalog of Example 1.1 presented above. In Example 1.2, we show a possible simplified conversation between a decision analyst and a decision maker to identify the best used car.

Example 1.2 Illustrative example. A simplified example of a decision process involving a decision maker (DM) and a decision analyst (DA).

DA> Hello, I am here to help you find your best used car from this catalog.
DM> Ok, actually I think that I would like to spend around…
DA> Wait a moment please. I will ask you a series of questions and, at the end, I will tell which should be your best choice.
DM> Ok.

Eliciting preference about price:
DA> Do you have any preference on the price?
DM> Yes, as I was saying, I am ready to spend around 10'000 Euros.
DA> Why is that? You would not like to have a cheaper car, for example?
DM> Sure, but I don't think that a car for… say 5'000 Euros, would be a good car.
DA> If there would be two used cars which are exactly the same, but one costs 5'000 Euros and the other costs 10'000 Euros, would you take the cheaper one?
DM> Yes, sure, but I don't think there are such bargains.
DA> Let's see. Just tell me what you really would like.
DM> Ok, the cheaper, the better.

Eliciting preference about mileage:
DA> Ok. Do you have any preference on the mileage?
DM> Sure, the lower, the better.

Eliciting preference about color:
DA> Perfect. What about the color? black, blue, red or white?
DM> Well, I don't like red very much.
DA> Ok, which one do you prefer the most?
DM> Blue or black.
DA> Can you choose between them?
DM> Blue would be perfect.
DA> Ok, this makes the following reference order: blue, black, white and red.
Between 0 and 100, let's give 'blue' a score of 100, as it is your perfect choice.
Can you give me a score for black, white and red?
DM> Black is a very reasonable choice also. Let's give it 90. For white, let's say half the blue, 50. I don't like red very much, but it is not so bad; let's give it 25.
DA> Perfect. Summarizing, 100 for blue, 90 for black, 50 for white and 25 for red. Is it right?
DM> Yes, well, if the car is really cheap, I don't really care about the color.

DA> Eh, for the sake of simplicity, let's ignore what you have just said.
DM> Ah, ok; you are the expert.

Eliciting preference about the airbag option:
DA> Perfect. Concerning the airbag, I assume you prefer to have one?
DM> Sure.

Eliciting the weights for each preference:
DA> Now, let's see how do you balance your preferences.
Let's say that there exists a car, which would cost 30'876 euros, have 197'000 km, red color and no airbag. From the other four hypothetical cars in the following table, which would be the best one?

Option	**Attribute swung from Worst to Best**	**price**	**mileage**	**color**	**airbag**
	(Worst car)	30'876	197'000	red	no
1	Price	2'666	197'000	red	no
2	Mileage	30'876	10'199	red	no
3	Color	30'876	197'000	blue	no
4	Airbag	30'876	197'000	red	Yes

Table 1.1: Illustrative example. Swing-weight assessment table.

DM> That's difficult to say. I have to compare all of them?
DA> Yes, please. Go ahead.
DM> Well, clearly mileage, color and airbag are important, but I would not pay 30'876 Euros for those cars. So the best option would be option 1.

DA> Good. We give a score of 0 to the car we started with, and a score of 100 to option 1. Now, how much satisfaction do you get, in percentage, by switching mileage from 197'000 km to 10'199 km when compared to switching price from 30'876 Euros to 2'666 Euros?
DM> Hey, that is very difficult to quantify!
DA> I know. Say it approximately. Half satisfied? A quarter satisfied?
DM> Half satisfied, or even a little bit less. Let's say 40% satisfied.

DA> Perfect. What about color? How much satisfaction do you get by switching from red to blue when compared to switching price from 30'876 Euros to 2'666 Euros?
DM> Let's say 50% satisfied.

DA> Good. What about the airbag? How much satisfaction do you get if having an airbag when compared to switching price from 30'876 Euros to 2'666 Euros?
DM> 60% satisfied.

DA> Great. We are done. I will introduce your preference model to the computer and

it will tell us which one is the best choice.
DM> Ok. I wait.

DA> Voilà. Your best choice is car #10: 9'238 Euros, 17'300km, blue and with airbag.

DM> Ok... but why?
DA> The computer has computed a score for each item according to your preferences, produced a ranked list according to these scores, and car #10 is the one with the highest score.
DM> Eh, can I see the rankings?
DA> Sure. See Table 1.2. If you are rational, you are going to take the top car in the list.

id	price	mileage	color	airbag	score	id	price	mileage	color	airbag	score
#10	9238	17300	blue	yes	90	#30	16678	45400	black	yes	74
#6	6448	55700	black	yes	88	#36	20770	10199	blue	yes	74
#12	9858	15700	black	yes	87	#29	16678	71000	black	yes	72
#21	12338	10200	blue	yes	86	#32	17918	57700	black	yes	72
#19	11718	38100	blue	yes	85	#13	10292	71000	white	yes	71
#16	11036	16700	black	yes	85	#34	19778	39000	black	yes	71
#22	13268	11000	blue	yes	85	#1	2666	76000	blue	no	70
#20	12338	20810	black	yes	83	#9	6515	76400	red	yes	69
#17	11346	62000	black	yes	81	#8	6510	76500	red	yes	69
#15	10521	76650	black	yes	81	#2	3348	78000	blue	no	69
#11	9858	113000	blue	yes	81	#37	24118	36000	blue	yes	67
#24	14074	62000	blue	yes	79	#35	20398	107000	black	yes	64
#28	16052	37500	blue	yes	79	#7	6450	97500	blue	no	63
#18	11718	69000	black	yes	79	#38	24738	42000	black	yes	63
#14	10416	96900	black	yes	79	#5	5890	115000	blue	no	62
#26	14632	72000	blue	yes	78	#39	28830	20500	black	yes	59
#23	13578	65200	black	yes	77	#27	14880	67000	red	yes	58
#31	17050	21000	black	yes	76	#40	30876	99000	black	yes	50
#33	18290	10500	black	yes	75	#3	4216	79000	red	no	48
#25	14570	78000	black	yes	75	#4	4836	197000	red	no	37

Table 1.2: Illustrative example. Ranked list according to user's preferences.

DM> It seems a good choice.
Actually the second is also a very good choice.
DA> Yes, you can see that they have a pretty close score, 90 and 88.

From this conversation, we can derive several observations:

- **The decision analyst leads the conversation (the decision maker has no initiative).** During the whole conversation, the decision analyst asks the questions, and the decision maker answers them. This contrasts with a mixed-initiative approach where both subjects can switch the leading role at any time during the conversation.

- **The decision maker does not see any car descriptions until the end of the decision process.** During the whole process the decision maker answers the questions about attributes and weights, but she cannot browse or compare the available cars. It is only at the end of the process that the decision analyst provides her with a list of ranked cars.

- **Accurate quantitative preference model at the end of the decision process.** Prescriptive decision theory can achieve good decision accuracy at the end of the decision process, if carried out properly.

- **Mean objectives.** At the beginning of the conversation, the decision maker said that she would like to spend around 10'000 Euros. This is a *mean objective*, i.e. the decision maker indeed prefers cheap cars, all else being equal, but she states a goal on price in the belief that she will be able to have good values on the other attributes. A decision analyst needs to detect mean objectives and ensure to elicit only real preferences.

- **Quantitative justification for the rational decision.** The score measures the desirability of a car given the preference model. Thus, a rational decision maker will take the car with the highest score. However, the decision maker has not learnt any insight about the set of available cars. For instance, she does not know if the airbag option influences the price of the car or not. She manifests this concern by asking the reason behind the solution given by the decision analyst.

One clear advantage of this approach is that the process guarantees good decision accuracy if carried out properly. This can be a requirement in decision problems that have a serious impact on other people (rather than a personal decision), such as deciding where to build a new freeway.

When catalogs are big, another advantage of such an approach (see Section 2.3 for an introduction to prescriptive decision theory) is that the effort to elicit the preference model does not depend on the number of items in the catalog, but only on the number of attributes used to describe the items. Thus, once the preference model has been elicited, a computer tool can rank the items in a catalog of any size automatically. Moreover, a computer tool can make use of the constructed preference model to find satisfying products as vendors update their catalog. This type of decision support system could inform the consumer each time it finds a product with a score above a given threshold. This possibility is increasingly being exploited in business-to-business electronic commerce, such as in [13].

However, people may not be ready to participate in such an interrogatory process for low-risk decisions, such as deciding which movie to watch or which photo camera to buy. Even more importantly, there may be a lack of **confidence** in such an approach because there is not a qualitative justification.

1.3.3 Solving the example in traditional commerce

Many ideas in decision support tools for e-catalogs have been inspired from traditional commerce. In traditional commerce, a vendor and a consumer have a much more unstructured conversation compared to the example of the decision analyst. The consumer is not looking only for a good decision, but she may also enjoy taking part in the decision process by browsing and critiquing the products. Moreover, she also needs to be **convinced** by qualitative feedback rather than by some score. More generally, *descriptive decision theory* [5] studies how people actually make decisions (rather than how they should do it in order to make a good decision, as is the case in prescriptive decision theory).

Let us illustrate such a decision process by using the catalog of Example 1.1 presented earlier. In Example 1.3, we show a possible conversation between a vendor and a consumer in a traditional commerce.

Example 1.3 Illustrative example. An example decision process involving a vendor (V) and a consumer (C).

C> Hello, I am looking for a used car.
V> You have come to the right place! We have a selection of 40 used cars in great condition.

V> Do you have an idea of what you are looking for?
C> I am not sure; What do you have for 7'000 Euros.

V> Let me see ...
We have this car for 6'515 Euros with 76'000 km, *(pointing to car #9)*
or this one for 9'238 Euros with only 17'000 km, *(pointing to car #10).*

C> I don't know much about cars. Is 76'000 kilometers a lot for a used car?
V> Well, I would expect a life span of seven years for the first one, and of ten for the second one. What is more important to you, the price or the life span?

C> I see. Well, actually I don't like the color red for a car, *(pointing to car #9)*
and 9'238 Euros is a little too much, although I appreciate that it has longer life span. What other cars do you have?

V> We also have a car for 6'450 Euro, but without airbag *(pointing to car #7).*
C> I definitively want to have an airbag.

V> With an airbag, the cheapest ones are this one *(pointing to car #6)*,
which costs 6'448 Euros, has 55'700 km and is black,
or the one I showed you first *(pointing to car #10).*

C> Do you mean that you have nothing between 6'448 Euros and 9'238 Euros?
V> I am afraid not; there are indeed three such cars, but two are red and the other one does not have an airbag.

C> Well, then both options are reasonable. I need to think more about it.

V> Typically, people that spend more than 6'000 Euros are very sensitive to the mileage, so car #10 would be better. Moreover, people prefer blue cars these days.
C> I still don't know; both seem very good choices.

V> If I were you, I would take this one *(pointing to car #10)*: the owner had to sell his almost new car very quickly, half price because she had to move to Japan; it is a bargain!
C> Great! I'll take it.
V> Perfect! I suggest you also consider this product to keep your car shiny...

We can also derive several observations from this conversation:

- **Initial vague preferences.** At the beginning of the conversation, the consumer did not have any fully-developed preferencse. She could only state the qualitative preference "around 7'000 Euros".

- **Mixed-initiative interaction.** This conversation is not a sequence of questions and answers as in the case of the decision analyst. Both the vendor and the consumer ask and answer questions in a much more unstructured conversation. The reaction to a question can be to answer that question or to ignore it by asking or answering something else. This conversation model is called mixed-initiative interaction.

- **Eliciting preferences by showing appropriate items.** People have a vague idea of their preferences. Indeed, psychological studies have shown that people construct their preferences while learning about the available products [8]. The authors in [62] suggested that people can react to examples with positive and negative critiques. We can see an example of negative critique in the previous conversation when the consumer was presented with a red car which made her realize and express that she does not like this color. Also, a positive critique was done concerning the mileage.

- **Explaining diversity.** The vendor showed the best car according to the consumer's initial preference about price (6'515 Euros is the closest price to 7'000 Euros) and, more importantly, he also gave a second option with the purpose of helping the consumer to express more preferences. In this case, the second option was much better in terms of mileage and had a different color. As we saw before, it helped the user to express that she cares about the mileage. Note that if the vendor had shown as a second option the second most similar car (car #8), it would not have helped the user to express more preferences because car #8 and car #9 are practically identical.

- **Explaining conflicting objectives and relaxations.** The initial preferences[3] of the consumer (around 7'000 Euros and low mileage) cannot be satisfied with the available cars. In the case of the decision analyst, all the conflicts are somehow summarized into a score for each item. Instead, here the vendor explains the conflicts to the consumer. Even more useful, the vendor also explains how much she needs to relax with preferences in order to solve the identified conflicts: either you have a cheap car with 55'700 km, or you have a car with low mileage but you pay 9'238 Euros. The consumer gains a clear knowledge of the compromises and can use this knowledge to ground her decision, which makes her feel more **confident** in her decision.

- **Recommendations other than content-based.** Most of the sentences in the conversation have been motivated by the features of the cars. However, at the end of the conversation, the vendor makes recommendations which are based on popularity, e.g. "people prefer blue cars", and preferences from consumers similar to her, e.g. "people that spend more than 6'000 Euros are very sensitive to the mileage" or "I also suggest you to take this product to keep your car shiny". This knowledge can be typically extracted from a purchase history and is useful for *discovering* products that might not have been found by the customers themselves.

- **Rational expression of preferences.** By the end of the conversation, the consumer understands that the choice is between car #6 and car #10. Even if she realizes that she prefers a blue car (by seeing the two cars), she does not express this preference to the vendor because she does not expect the vendor to be able to help her further in deciding between these two cars. In the framework of decision support systems, authors suggested [62] that stating a preference involves some user effort and rationally users would do only expend this effort if they perceive a benefit in doing so.

- **Adaptation to the consumer's level of expertise about cars.** In the example, the consumer does not know how to interpret how good 76'000 km for a used car is, and thus the vendor translates the mileage information into life span. As well as in the case of the decision analyst, adapting to the consumer's level of expertise about the domain can generally be done by deriving attributes that are more understandable for the type of customer, as in [40].

- **Qualitative preference model.** As with the decision analyst, the vendor learns the consumer's preference model during the conversation. However, the vendor learns it in a qualitative way, rather than in a quantitative, accurate way. For instance, the vendor knows that the consumer does not like red cars, but she

[3]In decision theory, a *preference* has a precise semantics: a binary ordering of items; however we use here the term *preferences* with its more general colloquial meaning.

cannot quantify how much the consumer dislikes them. Moreover, even if the vendor has learnt that the consumer cares about price and mileage (among other features), the exact balance between them is not known.

- **Incomplete preference model at the end of the decision process.** Not only is the preference model not expressed quantitatively, but it is also incomplete. For instance, the customer expressed that she does not like red cars, but no preference order is given about the other colors.
- **Qualitative justification for the decision by the use of feedback.** During the conversation, the consumer has acquired an understanding of the car domain for her scope of interest. For instance, she knows that there are only two cars that cost between 6'448 and 9'238 Euros which are not red and which have an airbag, and so she clearly understands the tradeoff between price and mileage. This understanding has been achieved thanks to the qualitative type of feedback given by the vendor, rather than to a numerical score as in the case of the decision analyst. The consumer is able to justify her decision to herself and to others, and is therefore **confident** about it.
- **Common grounding.** By the end of the conversation, the consumer asks the following imprecise question: "Do you mean that you have nothing between 6'448 Euros and 9'238 Euros?", which indeed both vendor and consumer know to be false. What happened is that they had previously built a shared understanding ("grounding") around the criteria that the chosen car must have an airbag and cannot be red, and so the query was meant to be in this context.
- **Social presence.** At the beginning and at the end of the conversation, we observe samples of human warmth and sociability (social presence). For instance, at the end of the conversation, in order to convince the consumer, the vendor exposes the confidential information "the owner had to sell her almost new car very quickly, half price because she had to move to Japan; it is a bargain!", information that obviously cannot be found in a catalog of used cars for privacy reasons. This is important because the warmth of a vendor can help to **convince** the consumer to conclude a purchase.

This approach has the main advantage that it is more natural, i.e. people are used to this type of conversation, and there is a level of human warmth and sociability (not present in the case of the decision analyst), which makes the consumer feel more **confident** about her decision and potentially more ready to commit a purchase [90].

The main disadvantage is that the decision process is not well grounded, i.e., it does not guarantee that, at the end of the process, a good decision will be made. Moreover, the decision accuracy depends on the capabilities of the vendor to elicit the preference model and to be aware of the whole catalog. As the number of items in the catalog increases, it is more difficult for the vendor to know all of them. In a traditional commercial center, this is typically solved by splitting the domain in several subsets and having many vendors, each of them specialized in a sub-domain.

1.4 Thesis outline and contributions

In Chapter 2 we introduce the area of *Decision Support Systems for Electronic Catalogs* (DSSECs): decision support systems commonly used in e-commerce that assist *decision makers* in finding *desired items* from a set of available items. We present how electronic catalogs can be implemented using databases. We present the general underlying framework behind DSSECs, the *multi-attribute decision theory* under certainty and, in particular, we review a specific prescriptive decision strategy, *weighted additive* (WADD), and a specific descriptive decision strategy, *elimination-by-aspects* (EBA). Next we describe how several DSSECs have taken inspiration from traditional commerce, introduce the *query-feedback search* interaction model, and analyze several types of *system feedback* used in these systems: *matching list*, *ranked list*, *diversity*, *suggestions*, *system-proposed critiquing*, *look-ahead frequency*, *conflict*, *corrective* and *visual* feedback. We then describe objective and subjective measures to evaluate the performance of such systems with user studies. In order to achieve accurate and meaningful comparative evaluations, we assemble a DSSEC that solves most of the known deficiencies and yet remains quite simple. We call this system the baseline DSSEC.

In Chapter 3 we argue that decision confidence, the certainty of the decision maker that she has made the best decision, is also an important property to look at in DSSECs. Thus, we provide a general framework, the *decision confidence in query-feedback search* (DECOFE) framework, in order to study system feedback in terms their potential contribution to decision confidence building. Using this framework, we study the advantages and disadvantages for different types of feedback discussed in the previous Section, and we identify the most promising ones, *conflict* and *corrective* feedback. We then propose a potentially better type of system feedback, the *confidence building tradeoff explanation* (CBTE) feedback, which explains the query tradeoffs in terms of conflicts and relaxations that can potentially contribute to decision confidence building accordingly to DECOFE. We run two simulations to study the viability of CBTE.

In Chapter 4 we describe the user study that we carried out in comparing *ranked list* and *CBTE* types of feedback (with the DSSEC system described in Chapter 2 and *matching list* feedback as a baseline) and we obtain positive evidence for the validity of the DECOFE framework introduced in the previous Chapter. We introduce a behavioral research model to evaluate different DSSECs in terms of the objective and subjective measures introduced in Section 2.5. While results need to be interpreted with care as described in Section 4.6, the experimental study shows that CBTE outperforms ranked list feedback in all the subjective constructs presented in the research model, and that decision accuracy is not penalized. The study also provides evidence that the positive increment in decision confidence in CBTE is not only a consequence of the positive effect of the improvement in decision accuracy, but it is also derived from the its compliance with the properties given in the *decision confidence in query-feedback search* (DECOFE) framework. Other interesting results are explained.

We conclude in Chapter 5 with a summary and a discussion about possible future work.

Chapter 2

Decision Support Systems for Electronic Catalogs (DSSECs)

2.1 Introduction

Decision Support Systems for Electronic Catalogs (DSSECs) are decision support systems, commonly used in electronic commerce (*e-commerce*), that assist *decision makers*, the consumers, in finding *desired items* from a set of available items, such as products or services.

An *electronic catalog* (e-catalog) can be defined as a set of *item descriptions*, such as products or services. Each item is described by a predefined set of *attributes*, where an attribute is an abstraction of a characteristic of an *item*, such as "price", "color" or "delivery time". Formally, we define a *catalog* as a tuple *(A, D, O)* where $A = \{A_1, \ldots A_n\}$ is a finite set of attributes, each associated with a domain $D = \{D_1, \ldots D_n\}$ and O is a set of available item descriptions. If the domain values of an attribute a have an order, then a is said to be an *ordered attribute*. Given an item $o \in O$, $a_i(o)$ represents the domain value of attribute A_i for item o. Electronic catalogs can be implemented using databases and a particular database model, which, for the sake of simplicity and without losing applicability, we assume to be a flat database model.

The general underlying framework behind DSSECs is *multi-attribute decision theory*, a discipline that has its roots in operations research, psychology, economics and artificial intelligence and studies how a *decision process* is done or should be done. Given a consumer and a catalog, the decision problem is to find the item(s) in the catalog that best satisfies the preferences of the consumer. Usually, decision problems involve multiple conflicting objectives. Objectives conflict with each other in the sense that improving one objective (such as minimizing the price in the catalog of Example 1.1) may only be achieved at the expense of another objective (such as minimizing the mileage). The decision problem is solved throughout a decision process, which can be characterized by the tradeoff between *decision accuracy* (i.e. the quality of the process) and the *cognitive effort* required from the consumer. The normative way to solve such a decision problem is to perform a structured conversation guided by a decision analyst or decision support system and aiming at solving the conflicting objectives by computing a score for every item in the catalog. We have presented an example of how such a procedure could be in Section 1.3.2. The disadvantages of this normative decision process are the high elicitation effort and the low acceptance by many consumers to engage in such unnatural conversations. Descriptive decision strategies are much easier to apply and require less cognitive effort, at the expense of lower decision accuracy.

In traditional commerce, a vendor and a consumer engage in a much more natural conversation in order to solve a decision problem. Several researchers have taken inspiration from traditional commerce and decision theory to design DSSECs that are *better accepted by users.* While this is a very ambitious approach, quite a lot of work has been done in this direction. In the example of Section 1.3.3, we have shown how the vendor elicited preferences from the consumer by giving appropriate feedback and examples. This procedure has been adapted to DSSECs by several researchers with ideas such as *retrieval by formulation*, *conversational systems*, *information visualization seeking* (InfoVis) and *dynamic querying*, *candidate/critique*, *interactive data exploration and analysis* (IDEA), *example-critiquing*, *navigation by asking* and *navigation by proposing* interaction models. We call this type of interaction model a *query-feedback search.* It corresponds to an iterative process in which a user makes queries to the system in order to develop her domain knowledge and preferences through the system feedback, until she finds a good decision. Several types of *system feedback* have been proposed, namely: *matching list*, *ranked list*, *diversity*, *suggestions*, *system-proposed critiquing*, *corrective* and *visual* feedback.

When DSSECs are used in the framework of e-commerce by an e-vendor, new problems related with confidence and trust arise. Indeed, as noticed in Section 1.2, only 1.6% of the visits to an e-commerce site result in purchases, and two-thirds of shoppers who get as far as putting items in a virtual shopping cart abandon the process before checking out [94]. More precisely, *decision confidence* is the certainty of the customer that she has made the best choice, while *customer trust* is the belief of the customer that she can rely on the e-vendor [75]. Both constructs affect *trusting intentions* (such as the *intention to buy*). Although decision theory is not explicitly concerned with confidence and trust, these issues are gaining more attention in the area of e-commerce [78, 79, 81, 85, 86, 88, 64, 89], where the problem of low confidence and trust is amplified by the fact that e-commerce is far more impersonal, anonymous and automated than traditional commerce carried out between people [79] (see the example of Section 1.3.3).

The performance of DSSECs can be evaluated with objectives and subjective measures. Objectives measures include the quality of the decision (*decision accuracy*). Subjective measures include perceived user effort to make a decision (*perceived cognitive effort*), *perceived ease of use* and *perceived usefulness* (from the *technology acceptance model*), *perceived cognitive effort*, *trusting beliefs* (such as *competence*, *benevolence* and *integrity*), *trusting intentions* (such as the *intention to buy*) and especially important in this thesis, the certainty of the decision maker that she has made the best choice (*decision confidence*). The evaluation of DSSECs often requires user studies. However, these studies are quite time consuming and recruiting participants for a user study is quite a tedious task. For this reasons, true user studies have been considered only recently.

One common flaw in comparative user studies in our community is to compare new innovative DSSECs with rudimentary ones such as form-filling interfaces. While form-filling indeed corresponds to the most common approach currently used on the Web, it is widely recognized as having an extremely poor design. However, simple solutions for most of these deficiencies have already been documented in the literature. These solutions can be easily assembled to design a baseline system that corrects most of the identified drawbacks while remaining very simple. Such an enhanced baseline system then potentially leads to more accurate and meaningful comparative evaluations.

This chapter is organized as follows. In Section 2.2 we describe how electronic catalogs can be stored and queried using databases. In Section 2.3 we summarize decision theory which is the general underlying framework of DSSECs. In Section 2.4 we describe the query-feedback search interaction model and revise several types of system feedback used in DSSECs. In Section 2.5 we introduce objective and subjective measures to evaluate the performance of these systems. In Section 2.7 we describe a baseline approach for DSSECs which solve most of the deficiencies of rudimentary interfaces, and will be used later in this thesis for a comparative user study. Finally we conclude with Section 2.8.

2.2 Databases and Electronic Catalogs

In general, a *database* can be defined as a representation of part of the world [15], or simply a collection of information or pieces of knowledge that has been systematically organized for easy access and analysis. Common databases examples include the accounting of an enterprise, an encyclopedia or the stock of a vendor. A *database management system* (DBMS) is a computer program that manages and queries a database, and they were first introduced in the 60's. A *database model* defines the structure of the data and the set of operations that can be performed on it. Examples of database models are the flat, the relational, the deductive and the object-oriented models [17]. A *query language* is a high-level declarative language used to query a database.

The flat model consists of a single *relation*, which is defined as a set of tuples that all have the same attributes. The database is usually represented by a table, where each column represents an attribute and each row or *record* represents an item. For instance, in the database of Example 1.1, all the cells of the second column have values about price and all the cells of a given record describe one same car. The *domain* of an attribute is the set of possible values that can be assigned to that attribute in a record. For instance, we could restrict the domain of the "color" attribute in the database of Example 1.1 to be {"blue", "black", "red" and "white"}. An *ordered attribute* is an attribute where the values of the domain have a natural order. For instance, the "price" attribute is ordered because we can compare two domain values in terms of "cheaper" (or even "more expensive"), while the "color" attribute is not ordered, because the domain values cannot be naturally compared.

A relational model is a mathematical model defined in terms of predicate logic and set theory, and it consists of multiple related tables. While relationships between tables are not defined explicitly, attributes are used to match up records in different tables. Figure 2.1 shows an example relational database composed of three tables: (1) a table of used cars shown previously, (2) a table of purchases and (3) a table which states which cars have been acquired by each purchase (in this example we assume that a customer could buy more than one car at once). For instance, we can see that the customer named "Akira Macho" bought car #1 and car #7 in 12th February 2003.

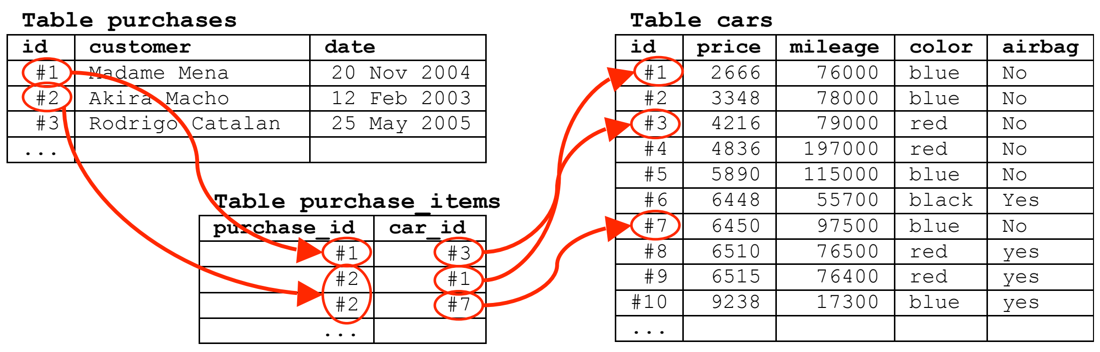


Table purchases

id	customer	date
#1	Madame Mena	20 Nov 2004
#2	Akira Macho	12 Feb 2003
#3	Rodrigo Catalan	25 May 2005
...		

Table purchase_items

purchase_id	car_id
#1	#3
#2	#1
#2	#7
...	

Table cars

id	price	mileage	color	airbag
#1	2666	76000	blue	No
#2	3348	78000	blue	No
#3	4216	79000	red	No
#4	4836	197000	red	No
#5	5890	115000	blue	No
#6	6448	55700	black	Yes
#7	6450	97500	blue	No
#8	6510	76500	red	yes
#9	6515	76400	red	yes
#10	9238	17300	blue	yes
...				

Figure 2.1: Illustrative example. A relational database.

One purpose of organizing information into a flat or relational database is to be able to easily query that information. For instance, we can ask a list of all the cars that cost less than 7'000 Euros. A more complex scenario would be storing the accounts of an enterprise in order to keep track of the pending payments.

Another purpose of using databases is to analyze the information by using automated computer programs. Data mining, also called Knowledge-Discovery in Databases (KDD) or Knowledge-Discovery and Data Mining, is defined as the nontrivial extraction of implicit, previously unknown, and potentially useful information from data [16]. For instance, association rules can be extracted from Example 1.1 stating that the selling price of red and white cars is less than 14'880 Euros, while blue cars go up to 30'876 Euros. A consumer could then take this information into account at the time of purchasing a car, knowing that she will be able to make more money when later selling her car if it is blue. Another more complex example is the market basket case analysis. If we extend the example of Figure 2.1 so that purchase orders include cars and accessories, we might determine that consumers purchasing a blue car also buy cleaning merchandise. Then a vendor can exploit this information as explained in observation 4 of Section 1.3.3.

A query to a relational database results in a new table derived from other ones. The standardized *structured query language* (SQL) can be used to describe the operations of combining several tables, grouping and sorting records by some attributes, and filtering records using a combination of conjunctive and disjunctive constraints. In Example 2.1, we show a query written in SQL language that computes the total price for each purchase. The results are shown in Table 2.1.

Example 2.1 Illustrative example. A query to a relational database using SQL language.

```
 SELECT
    id,
    customer,
    date,
    (SELECT SUM(price)
      FROM purchase_items pi LEFT JOIN cars ON (pi.car_id = cars.id)
      WHERE pi.purchase_id = purchases.id) AS total_price
FROM purchases;
```

id	**customer**	**date**	**total_price**
#1	Madame Mena	20 Nov 2004	4'216
#2	Akira Macho	12 Feb 2003	9'116
...			

Table 2.1: Illustrative example. Result of the Example 2.1 query.

The execution of a query has a computational cost associated with it. For instance, to query all the cars with a price of less than 7'000 Euros results in testing this condition 40 times (once for each car in the database). Indexes on an attribute (or a combination of attributes) provide a way to compute queries more efficiently and are usually implemented using B+ trees [14]. If the table has an index on price, the previous task would result in testing the condition 5.32 times in average (the logarithm in base 2 of the number of records in the table).

Databases can also be modeled using the constraint satisfaction paradigm, as proposed in [24, 21], although they are typically used in configuration, planning, resource allocation and scheduling problems. As in the relational model, a *constraint satisfaction problem* (CSP) [20] defines a set of attributes or *variables*. However, instead of describing items in terms of rows or records (an assignment of values for each attributes related to one same item), a CSP is defined by a set of constraints among the attributes. For instance, a constraint could state that all cars with a red or white color are cheaper than 14'880 Euros. Defining the proper set of constraints, the same database as in Example 1.1 can be described as a CSP. Once the database is described as a CSP, well known CSP algorithms such as backtracking [19] can be used to find all the solutions of the CSP, which represent the set of available items in the catalog. Then, a query such as "all the cars that cost less than 7'000 Euros" can be answered by adding the constraint that all cars cost less than 7'000 Euros and then using the same algorithm to find all the solutions of the modified CSP. Moreover, soft constraints can be used to specify preferences, such as "I prefer blue cars that cost less than 7'000 Euros", which with the help of optimization algorithms can be used to get a ranked list of all the available cars. Modeling databases as CSP problems instead of using a simple relation model is appropriate when the database is complex, such in the case of configurable products as non-direct flight tickets or insurance policies.

Electronic catalogs (or e-catalogs) are collections of items, such as products or services. They are stored and queried using a database model. Although they may require complex structures, for the sake of simplicity and without losing applicability, we assume in this thesis that e-catalogs are described using the flat model, i.e. one single table, with each attribute properly indexed for fast access.

Thus, we define an *electronic catalog* as a set of *item descriptions*, such as products or services. Each item is described by a predefined set of *attributes*, where an attribute is an abstraction of a characteristic of an *item*, such as "price", "color" or "delivery time". Formally, we define a *catalog* as a tuple *(A, D, O)* where $A = \{A_1, \ldots A_n\}$ is a finite set of attributes, each associated with a domain $\mathrm{D} = \{\mathrm{D}_1, \ldots \mathrm{D}_n\}$ and O is a set of available item descriptions. If the domain values of an attribute a have an order, then a is said to be an *ordered attribute*. Given an item $o \in O$, $\mathrm{a}_i(\mathrm{o})$ represents the domain value of attribute A_i for item o.

Note that the above definition, often used in the DSSEC literature, coincides with the definition of "Pawlak information system", also known as "attribute-value system" or "classification system".

2.3 Multi-attribute decision theory under certainty

Decision theory studies how a *decision maker* makes (or should make) a choice or *decision* [2, 5, 8, 11]. It is a discipline that has its roots in operations research, psychology, economics and artificial intelligence. Assuming that the set of options to choose from is well known in advance by the decision maker, a *decision problem* can be defined as the problem of finding the best item[1] from a set of available (or feasible) items, according to the decision maker's preferences. A *decision-making process* is therefore the process

[1]The terms *action, alternative*, *item* and *option* are used in the literature to refer to the same concept

of solving a decision problem. Such decision process can be carried out by a single person or by a group of persons, and it can be done alone or guided or helped by an expert. Common examples of decision making include shopping, deciding where to go for holidays, and deciding your vote in an election. This section is a summary from [1, 2, 3, 5, 8, 9, 11] of the main concepts used in this thesis regarding decision theory.

A decision problem under *uncertainty* is a decision problem where the *consequence* (or *outcome*) of a choosing an item (or *action*) is uncertain. For instance, let's say that the decision maker in Example 1.2 is now a contestant in a competition, and she has to decide between two actions: (1) to not take any risk, which has the consequence that she instantaneously win car #25 which has a score (or *utility*, see below) of 75 or (2) to take a risk by opening a door, with two possible consequences: (1) the contestant wins nothing if there is a rabbit behind the door, or (2) the contestant wins car #10 which has the maximum score, 90. In this example, there are two *world states*, either (1) there is a rabbit behind the door, or (2) not. If we don't know if there is a rabbit behind the door or not, then we say that the *state of the world* is unknown. When uncertainty over world states is quantified probabilistically, utility theory prescribes an action that leads to the highest expected utility [11].

On the other hand, a decision problem under *certainty* is a decision problem where choosing an item gives as a consequence the item itself (or more generally, each item is associated to a certain outcome). For instance, deciding to take the action of buying car #10 has the certain consequence that the consumer gets car #10. In such situations, the terms *items* and *outcomes* are equivalent, as are *value* and *utility*. Decision problems under certainty are the typical situation when a consumer wants to purchase some product or service, and are thus the focus of this thesis. **We assume decision theory under certainty for the rest of this thesis**[2].

Usually, decision problems involve multiple conflicting objectives. Objectives conflict with each other in the sense that improving one objective may only be achieved at the expense of another objective. For instance, the decision maker in Example 1.2 stated that she would like to minimize the price and the mileage at the same time. However, the cheapest car which costs 2'666 Euros has a mileage of 76'000 km, while the car with the lowest mileage, 10'199 km, costs 20'770 Euros. Thus, the decision maker needs to value this tradeoff, i.e. how much achievement on price is she willing to give up in order to improve achievement on mileage by some fixed amount.

Decision analysis aims at finding strategies (or a framework of thought [5]) that guarantees that the decision-making process leads to a good decision. Thus, the goal of decision analysis is to construct or *articulate* user preferences accurately. User preferences can be modeled by a value or *utility*[3] *function* that measures the desirability of each item [1].

[2]Note however, that we do not assume that users have a complete knowledge of their preferences. Actually, in most of the times users construct their preferences during the decision process.

[3]We will use the term *utility* to refer to value or utility (under certainty) in the decision theory sense,

Then, the best or *correct decision*[4] is the item that is most desired. *Decision analysts* (or *facilitators*) can support decision makers in a decision-making process in order to work out knowledge deficiencies in decision analysis strategies.

A basic way to construct a utility function for a decision problem is to ask the decision maker to rank all the available items, but this procedure becomes impractical as the number of items increases. However, the process can be simplified if the decision problem exhibits sufficient structure. *Additive independence* [5] is the most commonly used structural assumption, even if it is not always satisfied. A decision problem is *additive independent* if every subset of the attributes is mutually preferentially independent. A set of attributes are *mutually preferentially independent* if every subset of these attributes is preferentially independent of its complementary set. A subset of the attributes X is *preferentially independent* of its complementary set, *X'*, if the preference order of items involving only changes in the values in X does not depend on the values at which attributes in *X'* are held fixed [5, 10]. For instance, in the simplified Example 1.2, price is preferentially independent of mileage, color and airbag because the decision maker prefers to minimize the price no matter what the mileage, the color or whether or not the car has an airbag. However, if we do not ignore the decision maker's comment "if the car is really cheap, I don't really care about the color", then price is not preferentially independent of color, and the decision problem no longer satisfies the additive independence assumption anymore.

Other less strong structural assumptions exist. For instance, the generalized additive independence structural assumption is a generalization of the additive independence assumption, where independence holds among certain subsets of attributes, rather than single attributes [11]. In this case, the decision analyst would have asked questions concerning both the price and the color, rather than asking first about price and then about color.

On the other hand, psychological studies [8] have shown that the preferences of a user are vague when first confronted with a decision problem, and that they are constructed (or revised) during the decision-making process. Thus, the strategies used in a decision-making process are determinant for identifying the correct decision. *Descriptive decision theory* studies how decision makers use different strategies to solve a decision problem, balancing decision accuracy and the effort required to take the decision. That is, in some low-risk situations a person may be content to identify a satisfying decision, rather than the best decision, if it involves less effort in the decision making process. For instance, people typically do not spend much time and effort in selecting what movie to watch. Several decision strategies have been studied, including *elimination-by-aspects*, *satisficing*, *lexicographic*, *majority of conforming directions* and *frequency of good and bad features.*

---

and the term *value* to refer to the domain value of a cell of a table given an item and an attribute in the database sense. For instance, using the catalog of Example 1.1, the price value of car #1 is 2666 Euros.

[4]It is also referred as the *target item* [12]

In the following two sections we describe the most representative prescriptive decision strategy, the *weighted additive* strategy, and the most representative descriptive decision strategy, the *elimination-by-aspects* strategy.

2.3.1 Weighted additive (WADD) decision strategy

The *weighted additive* (WADD) strategy is a compensatory decision model in that it allows tradeoffs between attributes, i.e. provides the ability that good utilities on some attributes can compensate for bad utilities on other attributes. To construct the utility function, this approach only requires the assessment of the attribute utility functions and the weights. If they can be assessed accurately, the utility function is accurate (under the additive independence assumption) and thus picking the item with highest utility leads to the correct decision.

Under the additive independence assumption, the weighted additive strategy (WADD) [8] reformulates the utility function as a weighted sum of *attribute utility functions*:

$$u(x) = \sum_{i=1}^{n} w_i u_i(x_i) \tag{2.1}$$

where x is an item, x_i is the domain value of the item x for attribute A_i, w_i is a scaling factor or weight used to represent the importance of attribute A_i and u_i is an *attribute utility function* over attribute A_i. The u_i functions are scaled from zero (least desired attribute value) to one (most desired attribute value), $\sum w_i = 1$, $w_i > 0$, and thus u also ranges from zero to one.

Decision analysis prescribes several approaches for assessing attribute utility functions, such as *proportional scores* and *ratios*. The *proportional scores* approach can be applied to objectives that are associated to an attribute with an interval or ratio measurement, and it basically scales the attribute so that the utility of the best attribute value is one, the utility of the worst attribute value is zero and the utility of the intermediate attribute values reflects the distance between the best and the worst [9]. For instance, in Example 1.2 the decision maker prefers to minimize the price. We assign a utility of one to the cheapest price, 2'666 Euros in the catalog, and a utility of zero the most expensive, 30'876 Euros.

$$U_{price} = \frac{worst\_price - x_{price}}{worst\_price - best\_price} = \frac{30876 - x_{price}}{30876 - 2666}$$

The *ratios* approach is based on turning attributes with any level of measurement to ratio measurements using subjective comparisons. In Example 1.2, the decision maker compared the blue and white color saying that blue was perfect and white was half as good as blue. Using more comparisons, she assigned 100 points to blue, 90 to black, 50 to white and 25 to red. The decision analyst needs to simply scale these ratio assessments from zero to one. A similar approach could also be used to assess the price in case the preferences on price do not exhibit a linear model as in Example 1.2.

$$\begin{array}{lll} U_{color} = a + b * points & & U_{color}(blue) = 1 \\ & & U_{color}(black) = 13/15 \\ 0 = a + b * 25 & a = -1/3 & U_{color}(white) = 1/3 \\ 1 = a + b * 100 & b = 1/75 & U_{color}(red) = 0 \end{array}$$

Once the attribute utility functions have been assessed, we need to elicit the weights. A common approach is called *swing weighting*, and it is based on comparing well-constructed hypothetical items. First, a *worst item* is constructed by assigning the worst value to each attribute. Then, for each attribute, a new item is constructed by taking the worst item and switching the value of that attribute from worst to best. The list of constructed items in Example 1.2 can be seen in Table 1.1 at page 7. The decision maker needs to choose which of these constructed items is best, hereafter called the *best item*. Then, the procedure continues much like the previous *ratios* approach: for each attribute, the decision analyst asks how much satisfaction the decision maker gets (hereafter the *rate*, in percentage) by switching the value of that attribute from worst to best compared to switching from the *worst item* to the *best item*. With this procedure, we are eliciting the weight associated to each attribute because we are using equation (2.1) with a constructed item that has a weight of one for that attribute and a weight of zero for the rest of attributes. As the sum of weights is defined to be one, the weight for each attribute is simply the given rate of the associated constructed item divided by the sum of all the rates.

Once we have assessed all the attributed utility functions and the weights, we can compute the utility of all the items using equation (2.1).

2.3.2 Elimination-by-aspects (EBA) decision strategy

The *elimination-by-aspects* (EBA) heuristic describes a strategy commonly used by decision makers and it was first introduced in [3]. It contrasts with the weighted additive strategy in that it is more accepted by people [3] with the disadvantage that it is not a sound strategy, i.e. it does not guarantee that the decision maker will take the correct decision.

The EBA strategy is a sequential elimination process. The decision maker begins the process by defining the most important *aspect* that candidate solutions should possess. An aspect can be defined by an attribute and its desirable minimum individual utility. The items that do not include this aspect are removed from consideration. The process continues by selecting the second most important aspect and so on until there remains a single candidate item, which is selected as the decision. Thus, an aspect can be modeled as a *constraint*, as defined in the databases domain. If the utility function of an ordered attribute is monotonic[5], then the minimum individual utility can be replaced by the desired minimum attribute value (or maximum if the utility function is decreasingly monotonic).

For instance, in the catalog of Example 1.1 the decision maker could decide that a "price limit of 10'000 Euros" is the most important aspect. This aspect already eliminates 28 of the 40 cars. The second most important aspect could be "to have the airbag option". At this step there are only six cars left. Assuming that the order of preference for color is blue, black, white and red at last, as in Example 1.2, the next chosen aspect could be "at least as good as black". As there is still more than one candidate item, she selects the aspect "less than 50000 km", which leads to two cars, car #10 and car #12. The decision process would continue with the next important aspect; however, in this small illustrative example there are no more attributes to choose from. Moreover, we can see in Example 1.2 that these two cars have a very similar utility, making it difficult to differentiate using the elimination-by-aspects approach. This example also highlights that decision makers combine multiple decision strategies for one given problem as we will see in Section 2.3.3.

Tversky defines the *state of mind* as a sequence of aspects which leads to a unique candidate item. He argues that this state of mind is easy to apply, involves no numerical computations, and it is easy to explain and justify. As we mentioned in the beginning of the section, EBA has the drawback of having low decision accuracy. This problem comes from the fact that EBA cannot ensure that candidate items are better than those eliminated at each step. For instance, choosing the aspect "price limit of 10'000 Euros" would eliminate a very good car that has zero mileage but costs just a little more.

2.3.3 Discussion

Decision accuracy and decision confidence
We have revised a prescriptive decision strategy, WADD, and a descriptive one, EBA. Conflicting objectives are confronted and resolved by considering the extent to which one is willing to trade off more individual utility of one attribute for less individual utility of another attribute. *Compensatory* decision strategies are those that allow trade-offs between attributes, i.e. provide the ability that good utilities on some attributes can compensate for bad utilities on other attributes. Some decision strategies such as WADD confront the conflicts (*compensatory*), while others such as EBA avoid them

[5]A utility function u is increasingly monotonic if, whenever item x_i is preferred or indifferent to item x_j, then $u(x_i) >= \mathrm{u}(x_j)$. Conversely, a utility function u is decreasingly monotonic if, whenever item x_i is preferred or indifferent to item x_j, then $u(x_i) <= \mathrm{u}(x_j)$. A utility function u is monotonic if it is increasingly or decreasingly monotonic. Stated differently, a monotonic utility function is one that preserves the order.

(*non-compensatory*). Strategies that avoid conflicts tend to have lower decision accuracy.

Even if EBA has low decision accuracy, in [3] Tversky argues that people prefer this qualitative reasoning strategy rather than quantitative reasoning strategies such as WADD. He gives two reasons for this. Firstly, because it requires less computational cognitive effort to apply. However, this reason is not applicable in the case of decision makers using decision support systems (the focus of this thesis) to whom the computational task is delegated.

The second more crucial reason here is the concept of confidence. Tversky argues that people are reluctant to accept the principle that decision should depend on computations based on subjective estimations of utilities in which the decision maker herself has only limited confidence, as in the case of WADD. EBA is easy to explain and justify and thus it leads to more confidence in the rationality of the decision.

Adaptive decision maker
The author in [8] explains that decision makers may choose which strategy to use depending on (1) the decision problem and (2) their goal for the decision, either to be accurate or to minimize the effort of the decision-making process. Moreover, decision makers may use a combination of the basic decision strategies. For instance, if in the catalog of Example 1.1 there were more than one thousand cars, the decision maker could first use the elimination-by-aspects approach as a first filter based on mandatory requirements and then use a more accurate decision strategy on the remaining items.

Not monotonic construction of preferences.
Another important aspect of the strategies studied in decision theory is that they build a preference model incrementally along the decision process in a monotonic way, i.e. learning a new piece of knowledge does not conflict with the current preference model, but just adds to it. However, psychological studies [8] have shown that decision makers update or *revise* their current preference model with conflicting knowledge during the decision process, for instance by relaxing or narrowing their chosen aspects in the case of the elimination-by-aspects strategy. In the example of Section 2.3.2 the decision maker uses the elimination-by-aspects approach and finds two candidate cars while there are no more aspects to add to the preference model. Instead of allowing the decision process to become paralyzed and fail, in a real situation, the decision maker would typically revise the preference model by narrowing some of the aspects in order to eliminate one of the two candidates. Restricting the cars to the ones with blue color (the most desirable one), would successfully solve the decision problem. Thus, a decision-making process is a process where decision makers iteratively *revise* their preference model by adding knowledge or updating the current one (unless using a normative strategy such as WADD and probably with the guidance of a decision analyst). Also, psychological studies [8] suggest that decision makers revise their preference model as they learn more about the different items in the catalog, and thus the decision process itself is influenced by the

items that the decision makers inspect during this process.

2.4 Query-feedback search

Most people may not have an accurate idea of their preferences while looking for an item such as cars and movies, and moreover may they have an ill-defined understanding of the item domain [32]. Psychological studies [8] suggest the idea of *constructive preferences*, where decision makers begin the decision process with a vague set of preferences and refine them as they learn more about the domain (e.g., which attributes are of interest to her, the set of available values for the attributes or the available items in the catalog). In DSSECs, decision makers can learn about the domain through *feedback* provided by the system. Thus, in DSSECs the decision and the decision process itself are influenced by feedback that the system provides to the decision maker during the decision process.

For instance, in the illustrative example of Section 1.3.3 the consumer begins with the vague preference model "cars around 70'00 Euros". Then the vendor shows him two cars that cost 6'515 and 9'238 Euros. Next, the consumer inspects these two cars, one of which is red, and realizes that she does not like red cars[6]. By stating this new preference, she has revised her preference model. Thus, the consumer is constructing her preference model based on the cars that the vendor shows her.

Several authors have incorporated this concept into the design of user interfaces for searching in electronic catalogs [29, 32, 33, 24, 46, 50, 51, 52, 56, 57, 58] proposing the concepts of *retrieval by formulation* [29], *conversational systems*, *information visualization seeking* (InfoVis) and *dynamic querying* [66], *candidate/critique* [33], *interactive data exploration and analysis* (IDEA), *example-critiquing* [46], *navigation by asking* [37] and *navigation by proposing* [37] interaction models, among others.

Systems proposed in the literature differ in (1) how the preference model is modeled, (2) what type of feedback the system provides to the user and (3) what type of feedback the user is allowed to provide to the system. These three properties are however, often quite interrelated. For instance, in FindMe [32] the user feedback is limited to choosing among a set of options provided by the system such as "this apartment is OK, but make it (1) bigger or (2) cheaper" and a mathematical preference model is implicitly constructed by aggregating all the (possibly contradictory) user feedback.

In some systems, the preference model is a direct mapping of the user feedback. This is the case where the preference model simply consists of a *query*, such as "cars cheaper than 10'000 Euros and with blue color", and the type of user feedback received allows for the direct modification of a query, such as adding the extra constraint "with airbag".

[6]Note that this statement is a vague *preference* in a colloquial form, and lacks formal semantics, i.e. the context is not clear: does the user prefer any other car to a red one? does she prefer other colors, as long as other attributes are the same (ceteris paribus)?

In this thesis we are interested in this case, where the preference model is simply an explicit query that the user can directly modify.

We define a user *query* as a set of either *parametric constraints* or *parametric utility functions*. *Constraints* are conditions that need to be satisfied and they are typically used in descriptive decision strategies such as *elimination-by-aspects*, *satisficing*, *lexicographic*, *majority of conforming directions* and *frequency of good and bad features. Utility functions* can handle user preferences and they are typically used in prescriptive decision strategies such as *weighted-additive* and *equal weight* strategies, which have the ability that good utilities on some attributes can compensate for bad utilities on other attributes. Given such a query, several types of system feedbacks have been proposed in the literature and are discussed below. Depending on the type of feedback, queries can be handled as a set of parametric constraints or as a set of parametric utility functions.

For instance, given the query "maximum price of 10'000 Euros", if treated as a constraint it would filter out 28 from the 40 cars from the catalog of Example 1.1. The same query treated as a parametric utility function would assign a utility for each possible price. Combining the utilities for several attributes, this latter approach can then provide a ranked list of cars to the user.

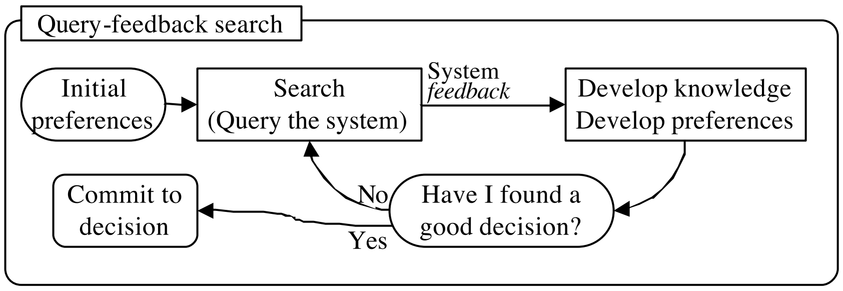

Query-feedback search
Initial preferences
Search (Query the system)
System feedback
Develop knowledge Develop preferences
Commit to decision
No
Yes
Have I found a good decision?

Figure 2.2: Decision process in the query-feedback search interaction model.

Thus we define the *query-feedback search* interaction model as an iterative process in which a user makes *queries* to the system in order to develop her *domain knowledge* and *preferences* through the *system feedback*, until she finds a good decision. It is illustrated in Figure 2.2. The query-feedback search interaction model consists of mainly of four steps:

1. **initial query**[7]: the user has some vague preferences, and issues an initial query to the system;
2. **system feedback**: the system generates feedback (e.g. a subset of the items of the catalog or even questions) taking into account the user query;

[7]This step is optional for some systems

3. **develop knowledge and preferences**: the user studies the system feedback, thus improving her knowledge about the domain and refining her preferences.
4. **modify query**: the user modifies the query to continue searching according to her preferences.

This process continues by repeating steps 2, 3 and 4 until the user makes a decision by selecting one of the items of the catalog.

Several classes of system *feedback* have been proposed[8], namely: *matching list*, *ranked list*, *diversity*, *suggestions*, *system-proposed critiquing*, *look-ahead frequency*, *conflict*, *corrective* and *visualization.* Note that these classes of system feedback are mostly inspired from conversations in traditional commerce, as the illustrative example provided in Section 1.3.3. In Table 2.2 we give a quick overview of these classes and they are then discussed in the following sub-sections.

Feedback class	**Message that it conveys**
Matching list	The following items match your query.
Ranked list	The following items are ordered by how close they satisfy your query.
Diversity	Your query is too vague. The following items may help you to focus more.
Suggestions	Explicit: Maybe the following items could also be of interest to you. Implicit: You may still have hidden preferences.
System-proposed critiquing	Compare your current choice to the rest of the items in the catalog.
Conflict	The following is the mismatch between what you want and what it is in the catalog.
Corrective	You need to relax your query in one of the following ways.

Table 2.2: Classes of system feedback revised from the literature, and the informal message they convey to the user.

2.4.1 Matching list feedback

A *query-building interface* was the first type of user interface that exploited the idea of the query-feedback search interaction model. It allows users to enter a query and view

[8]Note that in this thesis we take into account content-based systems only, thus intentionally excluding collaborative types of feedback, such as "people that bought this product also bought product X". For a survey on collaborative methods, see [59].

all the items *matching* the query (and thus, filtering out the ones that do not match). After users inspect the matching items, they can revise their query and get the new list of matching items. It was proposed in [29] under the notion of *retrieval by reformulation*, in order to assist a user in searching in a database. A screenshot of their system called RABBIT is shown in Figure 2.3. Here the query is a list of constraints, and it is made explicit by displaying the query during the whole interaction. Referring to the *query-feedback search* interaction model, the feedback provided by the system is just the list of all the items in the catalog matching the query. Concretely, the system displayed a list of short descriptions of the matching list and the user could select one of the items to display in detail. The user could select options such as “require” or “prohibit” a certain value of an attribute from the current displayed item. The updating of the query was straightforward and the system presented the new query and matching list of items. The case of a failing a query was not treated in this work and we guess that the system did not provide any assistance other than allowing a chronological retraction of previous updates or manual modification of the query.

Dynamic querying [66] is a similar interaction model in the sense that a user interface allows the progressive refinement of the query while seeing the matching items available in the catalog. The difference is that the query is elicited using graphical widgets and the items are displayed using visual representations. It is explained in more detail in Section 2.4.7.

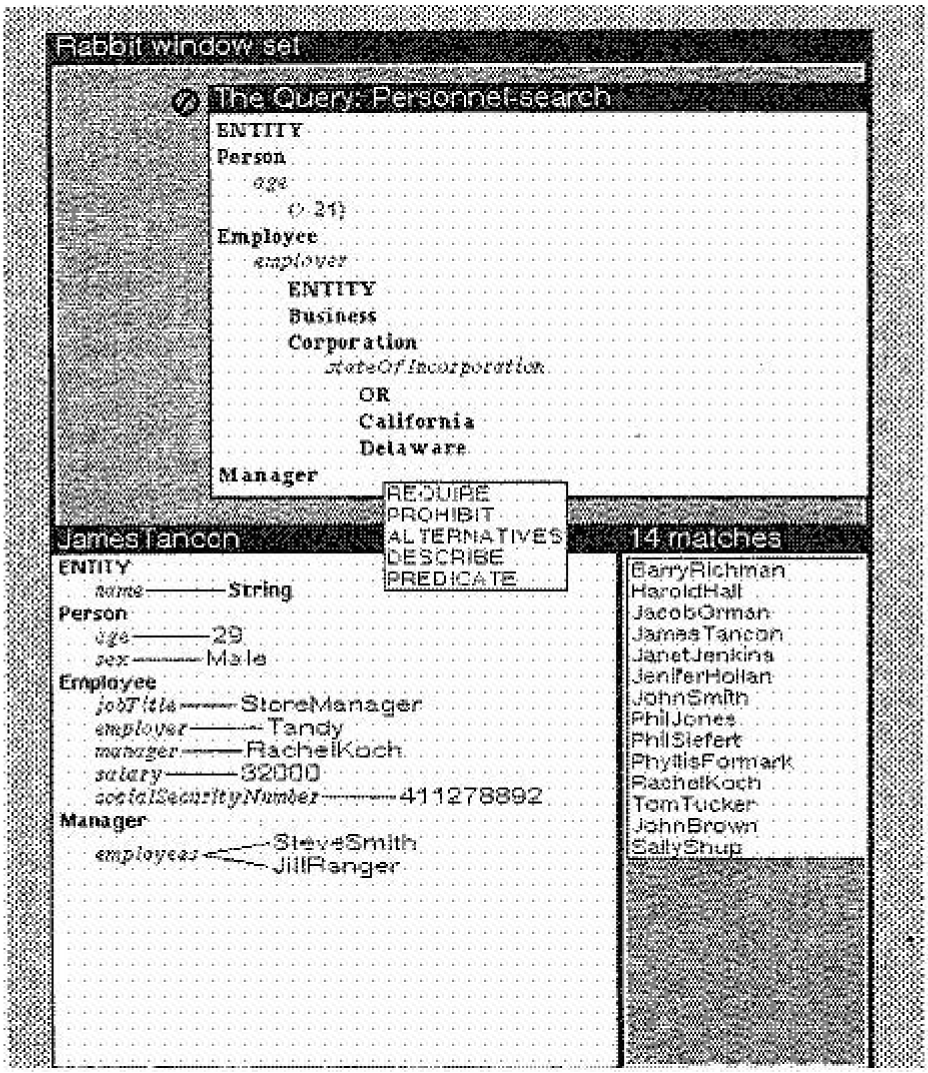
The decision process in these two DSSECs is similar to the elimination-by-aspects (EBA) decision strategy. As explained in Section 2.3.2, with the EBA strategy, decision makers construct a set of *aspects* used to filter out the items that do not satisfy them. However, in these systems the set of aspects is modeled as an explicit query which users can revise at any time by adding new aspects or revising previous ones at any time in any order, thus unblocking situations like the arrested decision process presented in Section 2.3.3.

2.4.2 Ranked list feedback

Rankings can be based on different criteria, such as *utility*, *similarity* or *pareto-optimality*. In Section 2.3.1 we introduced the weighted additive (WADD) decision strategy, which decomposes the *utility* function as a sum of *attribute utility functions.* The utility function can be constructed with the elicitation of attribute utility functions and the weights. If they can be elicited correctly, the utility function would be accurate, under the assumption of additive independence. However, the approaches prescribed in classical decision theory to elicit them are not well accepted for causal decision makers, who prefer a conversation as in the example in Section 1.3.3 rather than an interrogatory as in the example of Section 1.3.2.

In order to simplify the elicitation of the attribute utility functions and to have a more

[9]Figure reprinted from [29], Copyright ©1984, with kind permission of Elsevier.



```
Rabbit window set

The Query: Personnel-search
ENTITY
Person
    age
        (> 21)
Employee
    employer
        ENTITY
        Business
        Corporation
            stateOfIncorporation
                OR
                California
                Delaware
Manager

JamesTancon
ENTITY
    name———String
Person
    age———29
    sex———Male
Employee
    jobTitle———StoreManager
    employer———Tandy
    manager———RachelKoch
    salary———32000
    socialSecurityNumber———411278892
Manager
    employees—SteveSmith
             —JillRanger

14 matches
BarryRichman
HaroldHall
JacobOrman
JamesTancon
JanetJenkins
JeniferHollan
JohnSmith
PhilJones
PhilSiefert
PhyllisFormark
RachelKoch
TomTucker
JohnBrown
SallyShup
```

Figure 2.3: A screenshot of the RABBIT system [29][9].

natural conversation between a user interface and a user, *parametric attribute utility functions* are commonly used [12, 13]. We call *parametric weighted additive* (PWADD) strategy the approach of using parametric attribute utility functions with the weighted additive strategy. Using parametric functions has the main advantage of reducing the elicitation effort. On the other hand, there is not a theoretical correct way to design these types of functions while the whole process of utility elicitation is "as much of an art as it is a science" [5].

In Section 1.3.3 we described the *swing weighting* approach to assess the weights. Again, this approach is hard to use and other strategies are proposed in natural user interfaces. The most common approach is to let the user rate the importance of each attribute utility function, such as giving a number between 1 and 5 or selecting a rate from a predefined set of ratings such as "few important", "important" or "very important". As in WADD, these weights are scaled so that $\sum w_i = 1$, $w_i > 0$.

In the example-critiquing interaction model, the items displayed are the k items with highest utility. The user can criticize the displayed items by changing the parameters of the attribute utility functions or the weights. There exist algorithms such as [22] that can identify the top k-items without computing the utility for each item in the catalog.

Next we describe two examples of parametric attribute utility functions.

Example 1

In Figure 2.4 we see an example of a parametric attribute utility function for the attribute "price". It has two parameters, *Max_userprice* and *Offset*. *Max_userprice* is a value that the user can specify to indicate the desired maximal price. *Offset* indicates a margin for acceptable values, and it could be specified by an absolute value, or by a percentage value of *Max_userprice.* The computation of this parametric function requires eliciting only one value from the user, *Max_userprice.* If the price of the item is lower or equal to *Max_userprice - Offset*, the individual utility is 1.0. If the price of the item is bigger or equal to *Max_userprice + Offset*, the individual utility is 0.0. If the price of the item corresponds exactly to the price specified by the user, then the individual utility is 0.5. Otherwise the individual utility is a liner function between the two neighbor domain values.
One problem with this parametric attribute utility function is that the designer of the catalog would need to carry out a user study to determine the value of the *Offset* parameter that best fits the intended users.

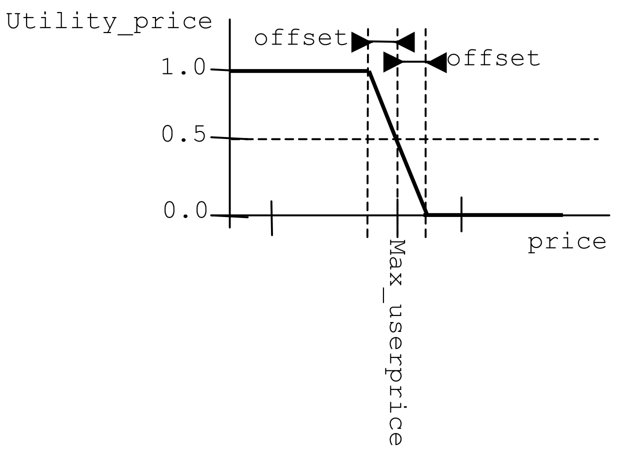


Figure 2.4: Example of a parametric attribute utility function for attribute ”price”.

Example 2

In Figure 2.5 we see another example of a parametric attribute utility function for the attribute “price”. It has 3 parameters: *Min_dbprice*, *Max_dbprice* and *Max_userprice*. *Min_dbprice* (*Max_dbprice*) is the domain value of attribute price of the item in the database which price is the cheapest (more expensive)[10]. As in the previous example, *Max_userprice* is a value that the user can specify to indicate the desired maximal price. As well as in the previous example, the computation of this parametric utility function requires eliciting only one value from the user, *Max_userprice.* If the price of the item is lower or equal to the lowest price in the catalog, the individual utility is 1.0. If the price of the item is bigger or equal to the maximum price in the catalog, the individual utility is 0.0. If the price of the item corresponds exactly to the price specified by the user, then the individual utility is 0.5. Otherwise, the individual utility is a liner function between the two neighbor domain values.
The main advantage of this parametric individual utility function in respect to the previous example is that the designer of the catalog does not need to carry out a user study to determine the parameters *Min_dbprice* and *Max_dbprice* because they are taken from the database and therefore there are fewer assumptions made about the intended users.

2.4.3 Diversity feedback

Showing a ranked list is very useful because it lets the user focus only on the k-best items. However, while the k-best items are very relevant to the user query, they may be

[10]Alternatively, as in [13], *Min_dbprice* and *Max_dbprice* can be explicitly given. In this case, parametric utility functions would not change if new items are added into the catalog.

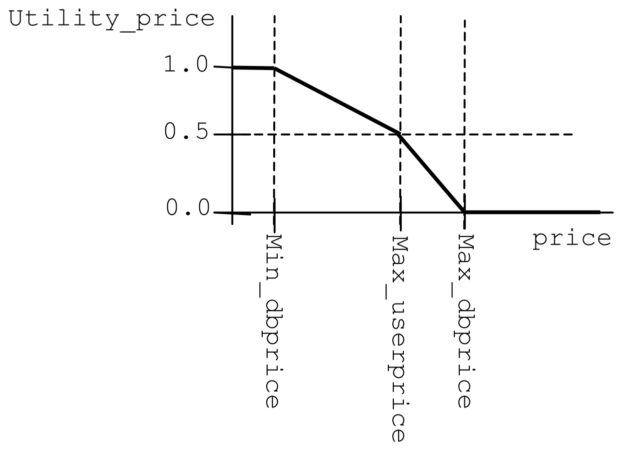


Figure 2.5: Another example parametric attribute utility function for attribute ”price”.

also very similar to each other and the user would not learn much about the domain. It would be better to present the best items relevant to the user query and at the same time show the range of different possibilities or the coverage of the recommendation space. We have seen such a case in the example of Section 1.3.3, where the second option the vendor showed was not the second best car because it was practically identical to the first one.

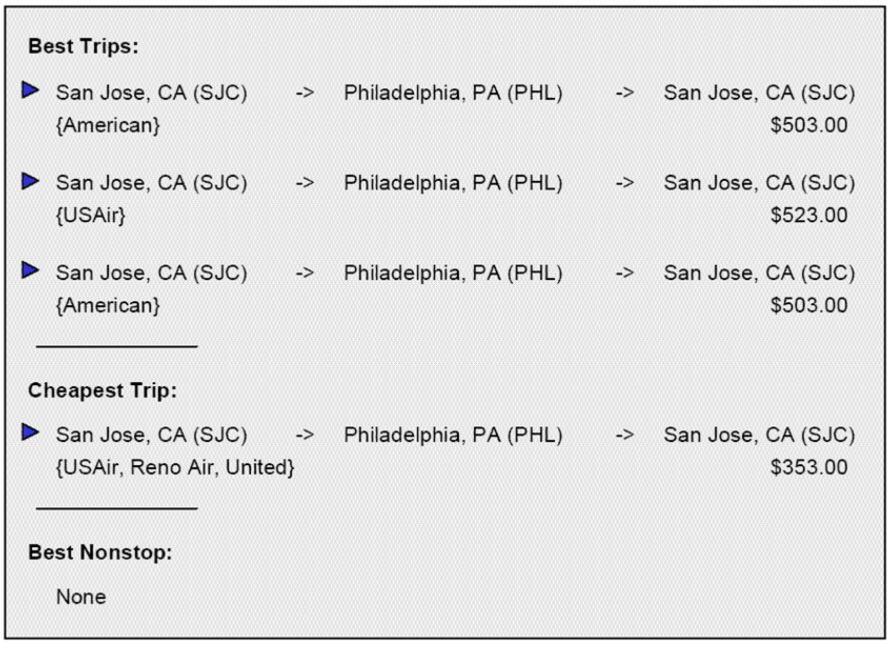
Showing *diversity* is a kind of *feedback* used in DSSECs. One of the first systems to implement such approach was the *Automated Travel Assistant* (ATA) [33]. Since then, other researchers [57, 37, 39, 50] have worked on this concept, proposing several metrics to measure diversity. The tradeoff to show diversity is often a decrease in similarity of recommended set in respect to the requested query. In [44], the author proposes to define a maximum of loss in similarity in domains in which the loss of similarity should be strictly controlled.

In [47], the authors observed that showing diversity during the whole decision process may actually result in a loss of decision accuracy. They showed that there are two phases: (1) first, the decision maker has a vague knowledge of her preferences and showing diversity is beneficial, and (2) a second phase where she has reached a more complete knowledge of her preferences and showing diversity may result in the target option being removed from consideration. Thus, in this second phase it would be preferable to show similarity (ranked list) rather than diversity in order to improve decision accuracy.

2.4.4 Suggestions feedback

As we explained previously, most people may not have an accurate idea of their preferences while looking for an item such as cars and movies. Linden introduced the idea that DSSECs should not only presents the k-best ranked items but also a set of items (or *suggestions*) which are significantly different among them and *not directly matching* the current preference model, in order to stimulate the expression of more preferences [33]. Thus, suggestions have the potential to stimulate the expression of preferences, thus getting more accurate preference models and, in turn, better decision accuracy.

In the *Automated Travel Assistant* (ATA) [33], the system shows items with respect to the preference model, and in addition, items with extreme values for some of the attributes in an attempt to elicit a new preference. For instance, if the user has expressed her intention to minimize price and have the airbag option, the system could also show the car with the minimum mileage, potentially stimulating the user to express a preference about mileage. See Figure 2.6 for a screenshot of the ATA system.


Best Trips:

▶ San Jose, CA (SJC) -> Philadelphia, PA (PHL) -> San Jose, CA (SJC)
{American} $503.00

▶ San Jose, CA (SJC) -> Philadelphia, PA (PHL) -> San Jose, CA (SJC)
{USAir} $523.00

▶ San Jose, CA (SJC) -> Philadelphia, PA (PHL) -> San Jose, CA (SJC)
{American} $503.00

Cheapest Trip:

▶ San Jose, CA (SJC) -> Philadelphia, PA (PHL) -> San Jose, CA (SJC)
{USAir, Reno Air, United} $353.00

Best Nonstop:

None

Figure 2.6: Screenshot of the Automated Travel Assistant (ATA) system [33][11].

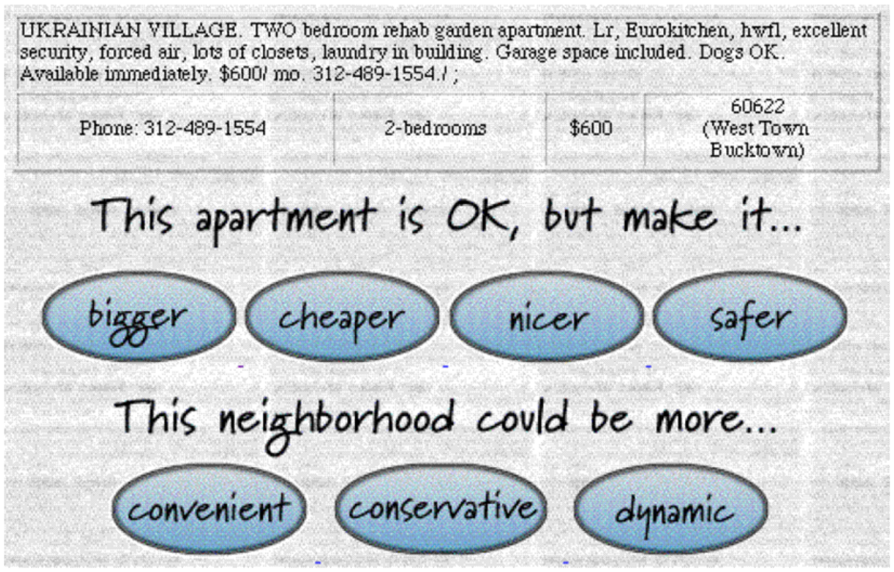
However, the heuristic proposed in ATA has not been evaluated with a user study. In [51, 61, 62], the authors suggested that this heuristic may not be very effective because the suggestions would not be reasonable choices and thus users would not perceive the benefit of adding a new preference. They proposed instead the *look-ahead* principle strategy, which states that suggestions should not be optimal under the current preference model, but should provide a high likelihood of optimality when an additional preference is added. For instance, considering the previous example, the system could suggest the best car

[11]Figure reprinted from [33], Copyright ©1997, with kind permission of Springer Science and Business Media.

in the catalog that minimizes price, mileage and has the airbag option.

2.4.5 System-proposed critiquing feedback[12]

A user interface with *easier questions* is the solution proposed in [32] with their FindMe system. See Figure 2.7 for an example screenshot. Concretely, the system selects only one example and during the feedback step, the system gives a very limited set of vague critique options like “this item is OK, but make it X”, where in the previous example ‘X’ can be ‘bigger’, ‘cheaper’, ‘nicer’ or ‘safer’. This type of question requires less cognitive effort to answer compared to more precise questions requiring a specific value such as “indicate the maximum price” because the answer contains less information and it is contextualized to one concrete example. The system updates the preference model according to user critiques and in the case of failing queries, the system performs a predefined domain-dependent ordered list of operations to relax the preference model. As user critiques can be contradictory during the course of an interaction and the preference model is hidden, the user may eventually not understand the justification of the displayed example nor, even more importantly, have the possibility to redirect it to a known state (apart from restarting the system) which may lead to confusion.


UKRAINIAN VILLAGE. TWO bedroom rehab garden apartment. Lr, Eurokitchen, hwfl, excellent security, forced air, lots of closets, laundry in building. Garage space included. Dogs OK. Available immediately. $600/ mo. 312-489-1554./ ;

Phone: 312-489-1554	2-bedrooms	$600	60622 (West Town Bucktown)

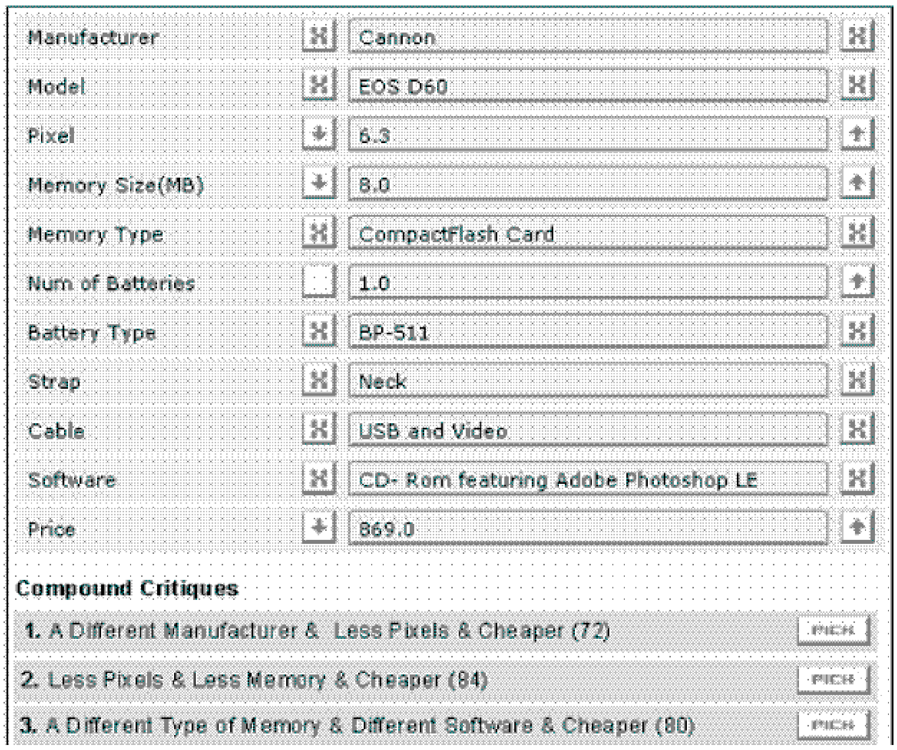
This apartment is OK, but make it...
bigger
cheaper
nicer
safer
This neighborhood could be more...
convenient
conservative
dynamic

Figure 2.7: A screenshot of the FindMe system [32][13].

[12]The term system-proposed critiquing has first appeared in [63], where the authors wanted to differentiate it from a user-motivated critiquing approach.

[13]Figure reprinted from [32], Copyright ©1996, with kind permission of the American Association for Artificial Intelligence.

Researchers [48, 52, 49, 64] have extended the previous work by proposing *compound critiques* feedback, where the system displays the best item according to the current preference model, and organize the remaining items in a set of categories explaining qualitative tradeoffs in comparison to the best one. Under the name of *dynamic critiquing*, in [48, 52] the authors propose to select the categories (or *compound critiques*) with lower support values, as these would eliminate many items from consideration if chosen. This approach has the advantage that, if one of the compound critiques turns out to be interesting to the user, selecting it would probably require less cognitive effort than manually inspecting all the matching items, and extracting and applying the individual critiques one by one. A screenshot of the system is shown in Figure 2.8.



Manufacturer	Cannon
Model	EOS D60
Pixel	6.3
Memory Size(MB)	8.0
Memory Type	CompactFlash Card
Num of Batteries	1.0
Battery Type	BP-511
Strap	Neck
Cable	USB and Video
Software	CD- Rom featuring Adobe Photoshop LE
Price	869.0

Compound Critiques

1. A Different Manufacturer & Less Pixels & Cheaper (72)
2. Less Pixels & Less Memory & Cheaper (84)
3. A Different Type of Memory & Different Software & Cheaper (80)

Figure 2.8: A screenshot from [47] implementing the *dynamic critiquing* approach[14].

Under the name of *incremental-critiquing*, in [49] the authors consider also past critique selections in order generate the categories. The approach in [64] differs in the method of generating the categories and the proposition to include a few items in each category.

In [45], the author proposes to feed the user critiques as semantic ratings into a collaborative filtering algorithm. For instance, a restaurant recommender system may be able to identify the class of users who select the "nicer" critique frequently and the

[14]Figure reprinted from [47], Copyright ©2004, with kind permission of Springer Science and Business Media.

"cheaper" critique rarely, and make new recommendations to them based on inter-user comparisons.

2.4.6 Look-ahead frequency, conflict and corrective feedback

Already in 1982, Kaplan showed the importance of system feedback in case of failing queries [28] in databases, in his system called CO-OP. The author proposed to show the *minimal conflict set* (MCS) which is a sound and minimal explanation of why there are no items in the catalog satisfying a query. For instance, given the catalog of Example 1.1 and the query "price $\leq$ 6'500 Euros AND mileage $\leq$ 50'000 km AND color = blue AND airbag = yes", the minimal conflict set tells that there are no items with a "price $\leq$ 6'500 Euros AND mileage $\leq$ 50'000 km". The author also proposed to show a set of possible relaxations which turns a failing query into a successful query. The type of relaxations proposed was to remove some of the constraints in the query. For instance, in the previous example the user has the choice to "remove constraint about price" or "remove constraint about mileage". This approach was named *cooperative answering* in databases, or *cooperative databases*.

The authors in [30] showed the importance of improving the type of proposed relaxations in the situation of failing queries, in their system called FLEX. Concretely, given a failing query *q*, they proposed to show a query *q'* as close as possible to *q* with the property of being a successful query. Therefore, in this case a relaxation could consist of slightly modifying a constraint rather than removing it as previously. For instance, in the previous example the user has the choice to "change price to 9'238 Euros (instead of 6'500 Euros)" or to "change mileage to 55'700 km (instead of 50'000 km)".

However, the task of computing this type of system feedback is NP-hard [18]. In this sense, researchers have focused on heuristics and methods to circumvent this drawback. In [30], the authors used heuristics to solve this task, but they did not provide a study of the computational cost. The authors in [55] proposed to speed up this process by using a small portion of the database to learn the domain's causal structure, which was then used to produce the modified queries.

This type of system feedback has also been proposed for *interactive problem solving* (such as *complex product configuration*) or *diagnosis* [23] tasks using the *constraint satisfaction problem* (CSP) framework, as in the case of the ILOG configurator [38]. In this case, this type of feedback has been named *corrective* feedback. QuickXPlain [25] is a conflict detection algorithm that can be applied to arbitrary constraint propagation CSP algorithms to find minimal conflicts. In [26], the authors proposed to first compile the configuration problem in an automaton representing the set of solutions of the CSP, and then use this data structure to provide corrective feedback efficiently.

The look-ahead frequency feedback shows the number of items that satisfy the current query in the case that an attribute-value pair was added (to the current query). For instance, given the query "price $\leq$ 11'000 Euros AND airbag=yes", the look-ahead frequency for the attribute *color* would state that there are 4 black cars, 2 blue cars, 2 red cars and one white car. This type of feedback requires a database test for each attribute-value pair.

2.4.7 Visual feedback

Information visualization, or *Information visualization seeking*, is the use of computer tools to interactively visualize representations of abstract data in order to amplify cognition [69, 92]. It relies on the extraordinary human visual capability to quickly find patterns and exceptions in visual representations [67]. Visual representations are made from space, marks, connection and enclosures, and temporal encoding, where space is its most fundamental aspect [69]. In the last twenty years several successful visualizations have been proposed, such as starfield displays and treemaps that help to explore large information spaces and as such are an alternative type of system feedback for implementing electronic catalogs.

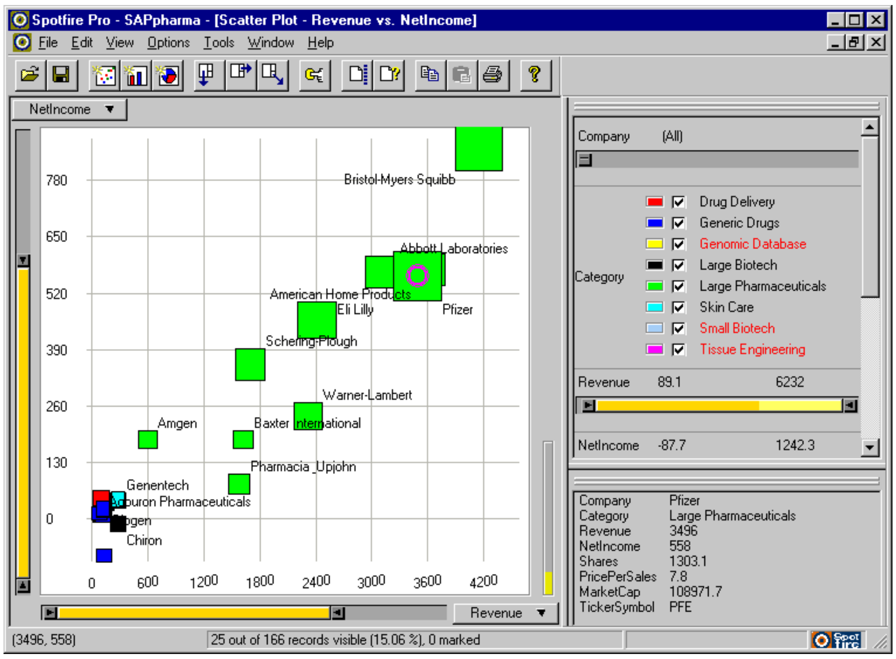
Starfield displays [67] are scatter-plot diagrams which map ordinal attributes to axes, where items of a database are positioned using markers. Although starfields use typically only two axes, the dimension of the visual representation can be augmented by adding properties to the markers, such as size, color, labels or icons. *Occlusion* is one major drawback of starfield displays, where markers in the plot that are very close may overlap thus making the diagram difficult to interpret.

Treemaps [65] are visual representations used when the items in a database have a hierarchical relationship. Items are represented by rectangles that are arranged and sized to graphically reveal underlying data patterns. As in the starfield displays, these rectangles can be colored and labeled to increase the number of dimensions being represented. Treemaps allows users to easily recognize complicated data relationships that are otherwise non-obvious.

Other visual representations exist, such as *parallel radial graphs*, *hyperbolic tree browser*, *parallel coordinates*, *fisheye* and *treeview*. For more information, the reader is referred to [69]. In a user study [36], the authors found out that their prototype InfoZoom based on the fisheye visualization would perform better in terms of search speed, accuracy, efficiency and user satisfaction than the hierarchical interface.

Dynamic querying [66] is an interaction model where users can specify queries and the visual representations are updated to show only the items matching the queries. Such queries can be rapidly adjusted by graphical widgets such as sliders, buttons and maps.

[15]This screenshot was produced using the Spotfire software. See http://www.spotfire.com for more details.



Spotfire Pro - SAPpharma - [Scatter Plot - Revenue vs. NetIncome]

Company	(All)
Category	🟥 ✓ Drug Delivery
	🟦 ✓ Generic Drugs
	🟨 ✓ Genomic Database
	⬛ ✓ Large Biotech
	🟩 ✓ Large Pharmaceuticals
	🟦 ✓ Skin Care
	🟦 ✓ Small Biotech
	🟪 ✓ Tissue Engineering
Revenue	89.1 6232
NetIncome	-87.7 1242.3

Company	Pfizer
Category	Large Pharmaceuticals
Revenue	3496
NetIncome	558
Shares	1303.1
PricePerSales	7.8
MarketCap	108971.7
TickerSymbol	PFE

(3496, 558) 25 out of 166 records visible (15.06 %), 0 marked

Figure 2.9: A screenshot of a starfield visual representation, used in the Spotfire software[15].

The visual representation used can be starfield displays, treemaps or any other, even a combination of them as in [68, 34]. As in example-critiquing, the *dynamic queries* interaction model supports the progressive refinement of the query and the continuous reformulation of the goals. Figure 2.9 shows a starfield display with dynamic querying. The user can select the attributes used in the two axes, and interactively filter the items according to her criteria.

2.4.8 Discussion

We have analyzed several DSSECs and have described a generalized model, called *query-feedback search* interaction model. The *query-feedback search* interaction model is an iterative process in which a user makes queries to the system in order to develop her domain knowledge and preferences through the system feedback, until she finds a good decision. Then, we have positioned the different DSSECs into the query-feedback search interaction model by describing the class of system feedback they provide: *matching list*, *ranked list*, *diversity*, *suggestions*, *system-proposed critiquing*, *corrective* and *visual* feedback. We have also illustrated them using the example of Section 1.3.3.

2.5 Subjective constructs

Trusting intentions (such as the intention to use, revisit or purchase from an e-vendor) on e-commerce have been studied from various viewpoints such as *search effort*, *domain knowledge*, *decision accuracy*, *ease of use*, *usefulness*, *predictability*, *competence*, *benevolence*, *integrity*, *enjoyment*, *social presence*, *confidence* and *trust* [78, 79, 81, 85, 86, 88, 64, 89]. E-commerce is far more impersonal, anonymous and automated than traditional commerce carried out between people [79] and it has not reached its full potential due to a lack of online purchase confidence [94] and consumer trust [79, 82, 83, 84]. According to [94], only 1.6% of visits to an e-commerce site result in purchases, and two-thirds of shoppers who get as far as putting items in a virtual shopping cart abandon the process before checking out. Researchers have shown the importance of taking into account decision confidence when designing user interfaces for search in electronic catalogs [80, 33, 56].

Predicting user acceptance of new technology is a mature research that started with the work of Davis in 1989 [72] with his *technology adoption model* (TAM). The author demonstrated that *perceived usefulness* and *perceived ease of use* were determinants of the intention to use a new information technology. Current theoretical models with roots in information systems, psychology and sociology can explain over 40 percent of the variance in individual intention to use information technology [87]. TAM has also been applied within e-commerce, and the authors in [86] have shown that intention to use also depends on trust towards the e-vendor, and trust is affected by perceived *integrity*, *security* and ease of use. Indeed, *trust* is a complex concept and has concisely been defined by [75] as *the belief of a customer that she can rely on the assistance of a vendor, or the willingness to depend on a vendor* [85].

Decision confidence can be defined as the certainty of the decision maker that she has made the best decision. It positively affects the *intention to buy* [90], and evidence of this in DSSECs comes from [63]. In [89] it was shown that decision confidence also positively affects the perceived usefulness. In the same study, the authors found that *decision accuracy* is a positive determinant of decision confidence, and that in turn, *search effort* and prior *domain knowledge* were determinants of decision accuracy. Domain knowledge can be defined as the knowledge that a user has about a domain, such as used cars or books, e.g. the possible objectives, the importance of such objectives and the performance of the items on such objectives. The authors in [80, 63] found out that *perceived control* positively affects *decision confidence.*

The constructs of perceived *competence*, *benevolence* and *predictability* are introduced in [85] as antecedents of trust, and define perceived predictability is the belief that the e-vendor's actions (good or bad) are consistent enough that a consumer can forecast them in a given situation. The authors also describe the benefits of trust in e-commerce as the intention to *purchase*, *cooperate* and *share information.* Also, a user study with explanation interfaces [60, 64] showed that perceived competence positively affects the *intention to return*, but not necessarily with *intention to purchase.*

Social presence is the extent to which a medium allows users to experience others as being psychologically present. The authors in [88] carried out an experiment providing the e-commerce site with different levels of human warmth and sociability and found out that social presence positively affects perceived usefulness, trust and *perceived enjoyment.* Perceived enjoyment, in turn, can influence users to revisit and trust an e-vendor.

Social influence is also a determinant of intention to use, and is defined in [87] as the degree to which an individual perceives that important other people, such as family, friends or colleagues, believe she should use the new information technology.

Table 2.3 summarizes the relationships of the different constructs.

2.6 Experimental design

To evaluate the performance of DSSECs, we have subjective measures (the ones presented in the previous Section) as well as objective measures (decision accuracy). Researchers have designed questionnaires that have shown been to have strong content validity to evaluate the previous subjective measures. Particularly, for measuring perceived ease of use and perceived usefulness, the questionnaire provided in the TAM model is widely accepted [72]. In [85] there is a questionnaire to measure trust, in [88] for enjoyment and in [35, 41, 89] for confidence. To measure perceived cognitive effort, the standard accepted evaluation is the NASA TLX test [73].

In [8] the author defines two fundamentally different ways to define the quality or accuracy of a judgment or decision: (1) A good decision is one that follows a good decision process, or (2) a good decision can be defined solely on the basis of the utility of the decision. The community has taken the second approach as it is easier to measure. Concerning this second approach, one technique is to use basic principles of coherence, such as consistency (repeating the same decision to the same decision problem in different points in time), not exhibiting intransitive preferences, or not selecting dominated items [8]. However, concerning dominated items, users state additional preferences only if they perceive this as beneficial because there is a user cost in doing so [61], and thus, it is not possible to decide whether one item is selected or not because the system may not have completely elicited the preference model of the user.

In risky choice environments, [6] defined decision accuracy as the ratio of *correct decisions* out of the total number of decisions, where a *correct decision* is the decision with highest utility. To evaluate whether a decision is correct, the authors in [35] proposed to give the opportunity to the consumer to change her mind and switch to another item after making a purchase decision, where switching would indicate an incorrect decision. The authors in [56] proposed an experimental design in which the user first makes a decision

	Decision Accuracy	**Perceived Usefulness**	**Enjoyment**	**Decision Confidence**	**Intention to use**	**Trust in e-commerce**	**Intention to buy**
Search effort	[89]						
Domain knowledge	[89]						
Decision accuracy				[89]			
Perceived ease of use		[72, 86, 89, 88]	[88]	[89]	[72, 86, 87]	[86, 88]	[89]
Perceived usefulness					[72, 86, 87]	[88]	
Perceived control				[80, 63]			
Perceived predictability						[85]	[85]
Perceived competence, benevolence, and integrity					[60, 64]	[85, 64]	
Enjoyment						[88]	
Social presence		[88]	[88]			[88]	
Social influence					[87]		
Confidence		[89]					[90, 63]
Trust		[86]			[86]		[85, 86]

Table 2.3: Relation of construct antecedents. A citation shows a row being a direct determinant of a column. For instance, search effort is a direct determinant of decision accuracy, and it is described in [89].

by selecting one item using the user interface under evaluation, and then there is a validation phase in where the user is presented with a sortable flat list of the items which the user is asked to carefully inspect all of them in order to find again the best item. If the selected items in both phases are the same, the decision is assumed to be correct.

However, this design has two main disadvantages: First, the procedure becomes less feasible as the number of the items in the catalog increases, as it would require too much effort for the participant to inspect all the items. Secondly, this validation is negatively affected by the *confirmation bias*, which is a form of selection bias in collecting evidence, wherein decision makers search evidence that confirms their hypothesis (or decision), rather than evidence that could disconfirm their hypothesis [70]. More evidence of this bias comes from [4], showing that once impressions have been formed, they exhibit remarkable perseverance. Thus, once participants make a decision, they will tend to stick to it, i.e. they will not be ready to change their previous choice. In this case, as only some of the participants will make the extra effort to carefully look again for the best decision, this validation phase tends to overestimate decision accuracy.

Note, however, that while probably it cannot be used as an absolute measure, this procedure is still a valid indicator as a relative measure for comparative user studies. It is, indeed, the one we use in Chapter 4.

2.7 Baseline system

In the basic form of a form-filling DSSEC, users are asked to completely fill in a form containing some of the most relevant attributes and then the system switches to a new window displaying the set of matching items in the catalog. Concretely, the system elicits a query by asking the user to instantiate a set of parametric constraints. The query is then the conjunction of the constraints. The user is not allowed to revise the query. If she wishes to do so, she needs to restart the process.

One common flaw in comparative user studies in our community is to compare new innovative DSSECs with rudimentary approaches such as the previous form-filling one. While form-filling indeed corresponds to the most common approach currently used on the Web, it is widely recognized as having an extremely poor design. However, simple solutions for most of these deficiencies have already been documented in the literature.

In this Section, we assemble a baseline system that corrects most of the identified drawbacks from the form-filling approach while remaining very simple. We argue that such a system can be a very acceptable baseline DSSEC, and we suggest that comparative user studies of innovative approaches using this baseline would provide more insight about the new value they bring.

In the next subsection, we describe a set of basic improvements to circumvent known deficiencies with the form-filling approach. Then we show an example implementation of such a system with the matching list type of system feedback and a second one with PWADD ranked list feedback. We then conclude with a summary.

2.7.1 Basic improvements

Query-feedback search interaction model. As we discussed earlier, psychological studies [8] suggest that decision makers begin the decision process with a vague set of preferences and refine them as they learn more about the domain. Thus, the fact that the form-filling approach does not implement an iterative process is a major drawback. While users could revise the preferences and restart the search process, they mostly do not [62]. This may be due to the cognitive effort involved in remembering the previous query and refilling in the entire form. Applying the query-feedback search interaction model described in Section 2.4 here is straightforward. The system needs to allow the user to go back to the form-filling window, display the previous query to the user and let her revise a subset of the parametric constraints. Afterwards, the user can see the updated matching items in the results window and repeat the iterative process until she makes a decision.

Merging query elicitation and feedback windows. Again, psychological studies [8] suggest that decision makers revise their preferences as they inspect items. Adding the previous improvement to the form-filling approach, users are allowed to revise preferences by going back to the previous window. However, once in this window, users need to remember what items they inspected in order to revise the preferences, which may add more cognitive effort. The other way around is also true: when users are inspecting the results, they may question why some items are or are not there and remembering the query involves some effort. The limitation of human information processing in short-term memory requires that multiple page displays be merged [95]. Therefore, a basic improvement is to merge both windows into one, where users can revise the query and inspect items without the need to switch between the two windows. Evidence in this direction comes from a user study [62], where participants that needed to switch the window between preference elicitation and inspecting the results, called *iterative form-filling*, rarely went back to revise their preferences and abandoned the search quickly, while participants that used a system merging preference elicitation and results in one window extensively revised preferences. Additional evidence in this direction is the user study in Chapter 4, where some participants commented that they would like to be able to modify a constraint even from the matching list window.

Instantaneous system feedback update. In the form-filling approach, after users revise the query using the form, they are required to confirm the changes before seeing the updated system feedback typically by clicking on a button. This confirmation can be intimidating. Evidence in this direction comes from two user studies: in the one we present in Chapter 4, changes made to the query were instantaneously reflected in the results, while in [62] users needed to confirm the changes before the results were updated. The impact was that participants in the first study revised the weights associated with the parametric constraints in the query 6.1 times per interaction on average, while in the second study they almost never did. However, a comparative user study in equal conditions other than the independent variable should be carried-out in order to assert the proposed statement.

Include all parametric constraints in the elicitation window. In [54], the authors explain that users should be allowed to express preferences in any attribute rather than a fixed subset. This is also especially important when comparing a recommender system (RC) with such a baseline approach: it is often the case that the RC makes recommendations taking into account the *rating of popularity* of the items in the catalog. Not allowing the participants using the baseline approach to filter by this rating may lead to a misleading comparison evaluation. Also, the system should allow several types of parametric constraints on an attribute, rather than allowing the most common one. For instance, it is common that systems allow users to put a maximum price, but not a minimum one.

Semantic resolution. The description of a parametric constraint may be ambiguous. Consider, for example, that a form asks for "type of car": it may be understood as the condition (used or new car), the category (sedan, limousine, off-road) or even the type of fuel. It has been shown [29] that much of the ambiguity can be eliminated by the simply presenting a sample value.

Show single attribute diversity. It has been shown that explaining the diversity of items in the catalog improves decision accuracy (See Section 2.4.3). However, this is an advanced feature and topic of research. Nevertheless, showing the diversity of the values for a single attribute is quite straightforward. In the study of Chapter 4, participants were asked, for the case of an ordered attribute such as price, whether they would prefer to select one value from a list of possible values in the catalog for that attribute (where they could see the diversity) or to manually enter the value. The results showed that 61.5% of the participants preferred to have a list of possible values. Thus, having the possibility to show the list of possible values in the catalog for a given attribute may be beneficial.

Show items from the beginning. Again, in the study of [62], the authors showed that when users are asked to state their preferences without considering the available items, many of these preferences are simply inaccurate (or *mean* preferences). Having the system with the query elicitation and results windows merged into one, the previous problem can be partially solved by displaying all (or a subset of) the items in the catalog from the very beginning.

Complete matching list. In the user study of Chapter 4 participants were asked if they prefer to see only three items (as in typical recommender systems) or if they prefer to see the list of all matching items (with a scroll bar if they do not fit on the screen). The result was that 69% of the participants preferred to have the complete list, while 15% did not care and 15% preferred to see only three items. Another study [93] showed that users rarely go beyond the first page when the matching list is split into several pages. Based on these two results, it seems beneficial to show the complete list without splitting it into several pages.

Sort by attribute. In the user study of Chapter 4, participants who were allowed to select an attribute with which to sort the matching list were asked whether that feature was useful. The result showed that 84.6% of the participants agreed while 15.4% did not care. The same result appeared in the same study, where participants that were not allowed to use this feature were asked whether they would like to have this feature.

Comparison matrix. A comparison matrix allows users to easily compare the set of items that are being considered seriously for making a decision. Adding this feature in a DSSEC is quite simple and it has been found to positively affect decision accuracy [35].

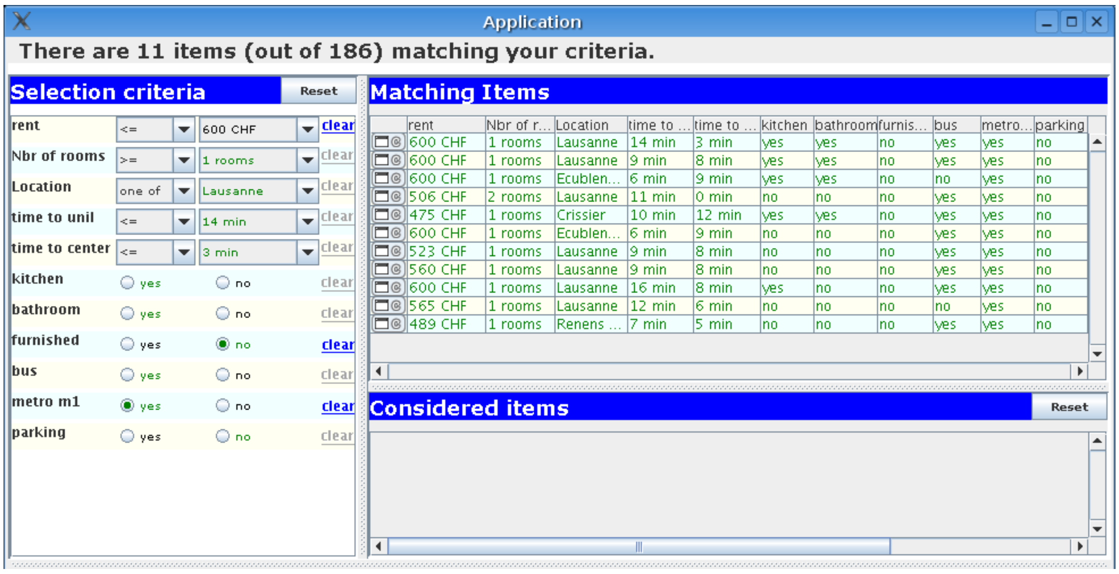
We argue that the form-filling approach with the basic improvement provided in this Section makes it a very acceptable baseline DSSEC, and we suggest that user studies of innovative approaches in comparison to this baseline would provide more insight about the new value they bring.

2.7.2 An example implementation

An implementation of the proposed baseline DSSEC with *matching list* type of system feedback is presented here. The user interface consists of 3 windows: (1) the *Selection Criteria* window, where the users can specify their criteria or *query*, (2) the *Matching Items* window, where *feedback* is displayed as the list of items in the database matching the query and (3) the *Considered Items* window, a comparison matrix interface where users can place items under consideration. It implements a *query-feedback search* interaction model, where users can state their query, see the matching items, and revise the query until they make a decision. A screenshot is available in Figure 2.10.

Also, an example implementation of the proposed baseline DSSEC, adding PWADD ranked list type of system feedback is shown in Figure 2.11. Each attribute is associated to a parametric attribute utility function and a weight. Users can select a weight from one to five, using the stars below the name of the attribute. As there is not a unique correct way to design the parametric attribute utility functions, we asked experts[16] in the domain. Ordered attributes use the parametric attribute utility function shown in Figure 2.5 at page 36 scaled from 0.0 (least desired attribute value) to 1.0 (most desired attribute value) with 0.5 in the case that it corresponded to the user selected value. Unordered attributes can be either completely satisfied or completely dissatisfied. We

[16]We would like to thank Vincent Schickel-Zuber and Li Chen for their support



There are 11 items (out of 186) matching your criteria.

Selection criteria

rent	<=	600 CHF	clear
Nbr of rooms	>=	1 rooms	clear
Location	one of	Lausanne	clear
time to unil	<=	14 min	clear
time to center	<=	3 min	clear
kitchen	yes	no	clear
bathroom	yes	no	clear
furnished	yes	[✓ no]	clear
bus	yes	no	clear
metro m1	[✓ yes]	no	clear
parking	yes	no	clear

Matching Items

rent	Nbr of r...	Location	time to ...	time to ...	kitchen	bathroom	furnis...	bus	metro...	parking
600 CHF	1 rooms	Lausanne	14 min	3 min	yes	yes	no	yes	yes	no
600 CHF	1 rooms	Lausanne	9 min	8 min	yes	yes	no	yes	yes	no
600 CHF	1 rooms	Ecublen...	6 min	9 min	yes	yes	no	no	yes	no
506 CHF	2 rooms	Lausanne	11 min	0 min	no	no	no	yes	yes	no
475 CHF	1 rooms	Crissier	10 min	12 min	yes	yes	no	yes	yes	no
600 CHF	1 rooms	Ecublen...	6 min	9 min	no	no	no	yes	yes	no
523 CHF	1 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no
560 CHF	1 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no
600 CHF	1 rooms	Lausanne	16 min	8 min	yes	no	no	yes	yes	no
565 CHF	1 rooms	Lausanne	12 min	6 min	no	no	no	no	yes	no
489 CHF	1 rooms	Renens ...	7 min	5 min	no	no	no	yes	yes	no

Considered items

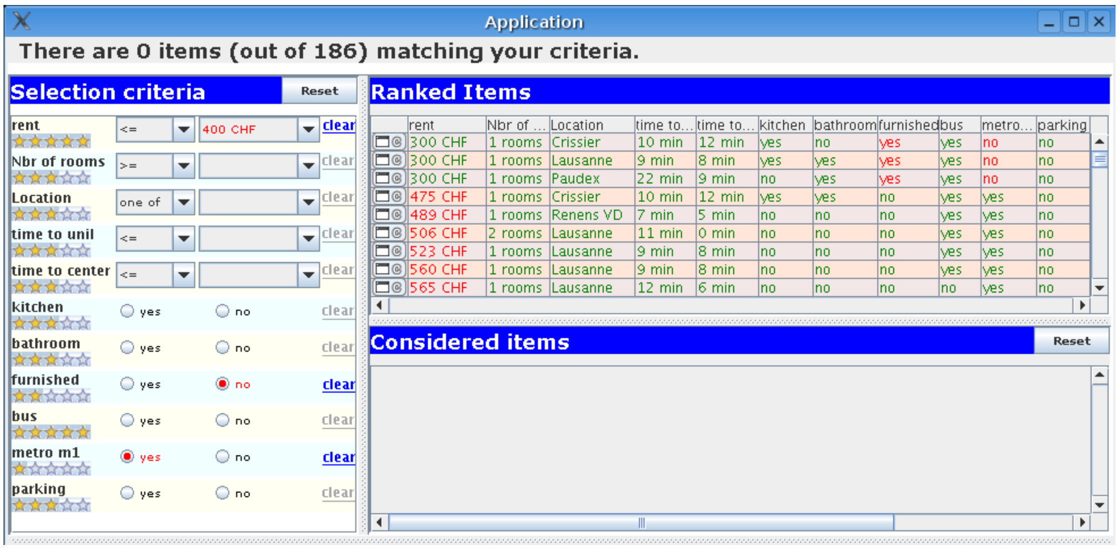
Figure 2.10: A screenshot of ECatalog, using the *matching list* type of system feedback.

would normally assign attribute utilities of 0.0 (unsatisfied) or 1.0 (satisfied). However, having an attribute utility of 1.0 (e.g. a car with the requested airbag option) would only be comparable to having an extreme value in the case of an ordered attribute (e.g. the cheapest price in the database). In order to get a more usable ranking, it was decided to set the attribute utility for satisfied values in the case of unordered attributes to 0.75, that is, between 0.5 (user selected value) and 1.0 (most desired attribute value) in the case of ordered attributes. Note that users will use the weights to balance the tradeoffs among the multiple utility functions. All the items are ranked and displayed in the *Ranked Items* window. From the users' perspective, the user interface is the same as the previous one, with the difference that users can enter a weight for each attribute and all the items are displayed using the ranking.

2.7.3 Discussion and Conclusion

We have improved some of the known deficiencies of the form-filling approach mainly with solutions from the literature while still keeping a very simple approach. We argue that such a system can be a very acceptable baseline DSSEC, and we suggest that user studies of innovative approaches in comparison to this baseline would provide more insight about the new value they bring.

In [54] there are more efficient interaction principles not considered here because they would make the system much more complex, while the purpose was to assemble a ba-


Application

There are 0 items (out of 186) matching your criteria.

Selection criteria — Reset

rent ★★★★★	<=	400 CHF	clear
Nbr of rooms ★★★☆☆	>=		clear
Location ★★★☆☆	one of		clear
time to unil ★★★☆☆	<=		clear
time to center ★★★☆☆	<=		clear
kitchen ★★★☆☆	yes	no	clear
bathroom ★★★☆☆	yes	no	clear
furnished ★★☆☆☆	yes	[✓ no]	clear
bus ★★★★★	yes	no	clear
metro m1 ★☆☆☆☆	[✓ yes]	no	clear
parking ★★★☆☆	yes	no	clear

Ranked Items

rent	Nbr of ...	Location	time to...	time to...	kitchen	bathroom	furnished	bus	metro...	parking
300 CHF	1 rooms	Crissier	10 min	12 min	yes	no	yes	yes	no	no
300 CHF	1 rooms	Lausanne	9 min	8 min	yes	yes	yes	yes	no	no
300 CHF	1 rooms	Paudex	22 min	9 min	no	yes	yes	yes	no	no
475 CHF	1 rooms	Crissier	10 min	12 min	yes	yes	no	yes	yes	no
489 CHF	1 rooms	Renens VD	7 min	5 min	no	no	no	yes	yes	no
506 CHF	2 rooms	Lausanne	11 min	0 min	no	no	no	yes	yes	no
523 CHF	1 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no
560 CHF	1 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no
565 CHF	1 rooms	Lausanne	12 min	6 min	no	no	no	no	yes	no

Considered items — Reset

Figure 2.11: A screenshot of ECatalog, using the *ranked list* type of system feedback.

sic system for fair comparative evaluations. For instance, when no items satisfy the users' query, showing partial matches would be desirable as described in [54]. However, rankings can be based on many different criteria, such as *utility*, *pareto-optimality* or *similarity*, each one with its advantages and disadvantages still under research, and thus, we suggest not to include these advanced features in a baseline approach.

We have also presented an example implementation of such as system with *matching list* feedback and a second one with *PWADD ranked list* feedback[17], which has been used for the comparative user study described in Chapter 4.

2.8 Summary

One of the goals of this Chapter was to provide definitions of the main concepts required for our research as found in the literature:

Decision Support Systems for Electronic Catalogs (DSSECs) are decision support systems, commonly used in electronic commerce (*e-commerce*), that assist *decision makers* in finding *desired items* from a set of available items, such as products or services.

An *electronic catalog* (e-catalog) is a set of *item descriptions*, such as products or services. Electronic catalogs can be implemented using databases and a particular database model, which, for the sake of simplicity and without losing applicability, we assume to be a flat

[17]The implementation is available open source for the research community. See Section 5.1.

database model.

Multi-attribute decision theory under certainty is a discipline that has its roots in operations research, psychology, economics and artificial intelligence to study how a *decision process* is done or should be done, and which is the general framework behind DSSECs. We explained that the difficulty of making a good decision mainly comes from the fact that most decision problems involve multiple *conflicting objectives*. Particularly, we have revised a prescriptive decision strategy, the *weighted additive* (WADD) strategy which confronts conflicts (*compensatory*), and a descriptive decision strategy, the *elimination-by-aspects* (EBA) strategy which avoids conflicts (*non-compensatory*) and thus requires less cognitive effort at the expense of lower decision accuracy.

Next, we have analyzed several DSSECs found in the literature and have described a generalized model, the *query-feedback search* interaction model. The *query-feedback search* interaction model is an iterative process in which a user makes queries to the system in order to develop her domain knowledge and preferences through the system feedback, until she finds a good decision. Then, we have positioned the different DSSECs into the query-feedback search interaction model by describing the class of system feedback they provide: *matching list*, *ranked list*, *diversity*, *suggestions*, *system-proposed critiquing*, *corrective* and *visualization*. We have also illustrated them using the example of Section 1.3.3.

We have discussed the importance of comparative user studies to evaluate the performance of DSSECs in terms of objective measures such as decision accuracy, and also in terms of subjective measures. Particularly, we have presented the technology acceptance model, which is used to explain user attitudes towards information systems, and extended models which explain trusting intentions in e-commerce such as the intention to use, to purchase or to revisit an e-vendor. Importantly, we have also discussed decision confidence and its benefits.

Finally, we have described a baseline DSSEC that solves most of the known deficiencies of rudimentary DSSECs, while remaining very simple. We argue that such a system can be used in user studies to achieve more meaningful comparative evaluations of innovative DSSECs.

Chapter 3

Decision confidence in query-feedback search

3.1 Introduction

In the previous Chapter we have reviewed several types of *system feedback* used in *Decision Support Systems for Electronic Catalogs* (DSSECs). The main motivation often provided in the literature for these types of system feedback is to increase the decision accuracy or sometimes more implicitly, to be beneficial "in some way". In this Chapter, we argue that increasing decision confidence, i.e. the certainty of the decision maker that she has made the best decision, is also an important property for system feedback design, that is often not (or only partially) considered in recent systems that usually focus on decision accuracy.

With this perspective, we provide a common framework in order to study the different types of system feedback in terms of potential contribution to decision confidence building. Using this framework, we study the advantages and disadvantages of several types of system feedback, and we identify the most promising ones. We then use the acquired knowledge to design a new potentially better type of system feedback, the *confidence building tradeoff explanation* (CBTE) feedback, combining *global relaxations*, the *minimal conflict set* and *local relaxations*.

This Chapter is organized as follows: In Section 3.2, we justify the benefit of decision confidence in DSSECs. In Section 3.3, we propose a framework to study decision confidence building in the *query-feedback search* interaction model, the DECOFE framework, and study several types of system feedback revised in Section 2.4 using this framework. In Section 3.4, we introduce a potential better type of system feedback called *CBTE*, and in Section 3.5, we provide a formal definition for it. In Section 3.6, we run two simulations to study the potential exponential size growth of CBTE feedback. In Section 3.7, we show an example implementation of CBTE feedback on top of the baseline DSSEC described in Section 2.7. We discuss related work in Section 3.8 and we summarize in Section 3.9.

3.2 Benefits of decision confidence in DSSECs

Usually, people need to feel confident before committing to a decision. *Decision confidence* can be defined as the certainty of the decision maker that she has made the best decision. Decision confidence can be beneficial in several ways. First, having good decision confidence helps to finally commit to the decision. Consider, for instance, the example of Section 1.3.3, where the consumer is hesitating between two cars. Even if both of them are very good choices, being unable to commit to one of them may be counterproductive for the consumer, as she might eventually never buy a car.

This problem is especially important for *e-commerce* due to the fact that e-commerce is far more impersonal, anonymous and automated than traditional commerce. For instance, in the example of Section 1.3.3, the vendor finally convinces the consumer emotionally by telling her "the owner had to sell his almost new car very quickly half price because she had to move to Japan; it is a bargain!". As noticed in Section 1.2, only 1.6% of the visits to an e-commerce site result in purchases, and two-thirds of shoppers who get as far as putting items in a virtual shopping cart abandon the process before checking out [94]. As it already has been shown in [90, 63], this low purchase rate can be significantly raised by improving decision confidence, as it positively affects the intention to buy.

Also, the consequence of a particular decision may depend on the degree of confidence held by the decision maker and communicated to others [76]. In these cases, raising the decision maker's conviction of doing the right thing can be a desirable property [77].

3.3 Decision confidence in query-feedback search (DECOFE) framework

Given a consumer and a catalog, we define here a *catalog decision problem* as the multi-attribute decision problem of finding the item in the catalog that best satisfies the preferences of the consumer[1]. Usually, decision problems involve multiple conflicting

[1]We assume decision theory under certainty, in which the consequences (items) of actions are certain, and thus an action is equivalent to simply choosing an item from the set of available items. See Section 2.3 for more about it. Note however, that we do not assume that users have a complete knowledge of their preferences. Actually, in most of the times users construct their preferences during the decision process.

objectives. Objectives conflict with each other in the sense that improving one objective (such as minimizing the price) may only be achieved at the expense of another objective (such as minimizing the mileage).

Given a catalog, a *decision support system for electronic catalogs* (DSSEC) is a decision support system that allows users to solve a catalog decision problem. The *query-feedback search* interaction model (introduced in Section 2.4) is an iterative process in which a user makes queries to the system in order to develop her domain knowledge and preferences through the system feedback, until she finds a good decision. We have seen several types of system feedback in Section 2.4.

When a user takes an item from a catalog, she ideally believes that she has found her most desired item from the ones available in the catalog, i.e. that her *decision* is the *correct decision.* However, a belief is typically associated with a degree of confidence[2] and confidence has been defined as the *subjective* judgment of one's decision [89]. We define *decision confidence* as the certainty of the decision maker that she has made the best decision.

In the following section, we extend the definition of the query-feedback search interaction model presented in Section 2.4 to take into account decision confidence. Next, we introduce four properties of the design of system feedback that can potentially help in building decision confidence. We then analyze different types of system feedback according to this framework.

3.3.1 Query-feedback search and decision confidence

Researchers have found that decision makers search for relevant information arriving first at a tentative decision and then attempt to convince themselves that it is indeed a good one. [71, 74]. In Figure 3.1 we show the decision process in the *query-feedback search* interaction model. We have deliberately split the sub-processes of developing preferences and building decision confidence for the sake of clarity. However, these two sub-processes may be mixed in the decision process.

Therefore we extend the definition of *query-feedback search.* **The *query-feedback search* interaction model is an iterative process in which a user makes *queries* to the system in order to develop her *domain knowledge*, *preferences* and *confidence* for *tentative decisions* through the system *feedback*, until she is convinced of her final decision.** The query-feedback search interaction model consists of mainly of four steps:

[2]According to the Oxford English Dictionary (1989), confidence is "the feeling sure or certain of a fact or issue; assurance, certitude; assured expectation".

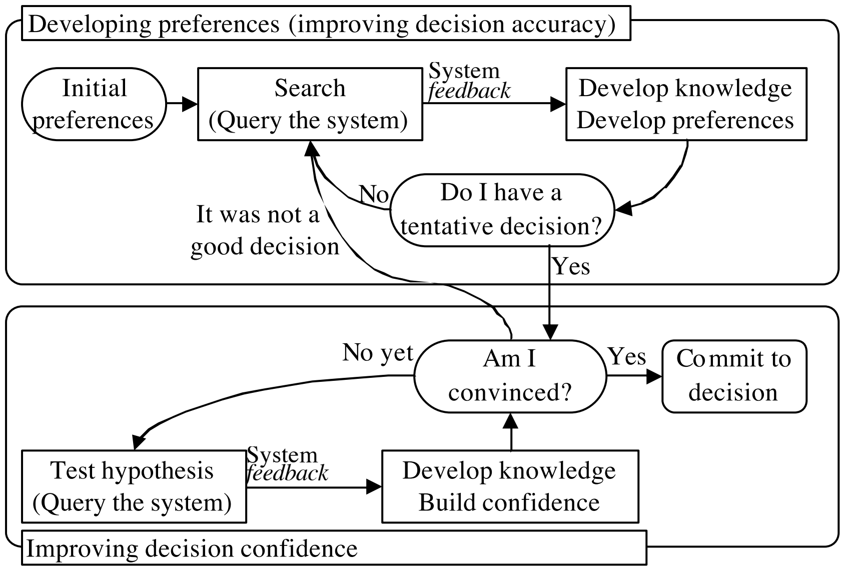

Developing preferences (improving decision accuracy)
Initial preferences
Search (Query the system)
System feedback
Develop knowledge Develop preferences
Do I have a tentative decision?
No
It was not a good decision
Yes
No yet
Am I convinced?
Yes
Commit to decision
Test hypothesis (Query the system)
System feedback
Develop knowledge Build confidence
Improving decision confidence

Figure 3.1: Decision process in the *query-feedback search* interaction model showing the construction of preferences and decision confidence building (DECOFE framework).

1. **initial query**[3]: the user has some vague preferences, and issues an initial query to the system;
2. **system feedback**: the system generates feedback (e.g. a subset of the items of the catalog or even questions) taking into account the user query;
3. **develop knowledge, preferences and confidence**: the user studies the system feedback, thus developing her domain knowledge, preferences and confidence for tentative decisions.
4. **modify query**: the user modifies the query to continue searching according to her preferences and tentative decisions.

This process continues by repeating steps 2, 3 and 4 until the user finds a justified decision.

3.3.2 System feedback properties

Intuitively, system feedback designed to improve decision accuracy should be based on *prescriptive* decision theory (i.e. how individuals should make decisions in order to find the optimal one) such as using the weighted additive decision strategy. On the other

[3]This step is optional for some systems

hand, system feedback designed to improve decision confidence should be based on *descriptive* decision theory (i.e. how individuals actually make decisions) such as using the elimination-by-aspects decision strategy. That is, users should feel more confident by using a decision strategy that they are used to.

In this way, we introduce four properties on the design of system feedback that can potentially help in building decision confidence: (1) *constraint-based*, (2) *universally valid*, (3) *negatory* and (4) *unique in its class.*

1. **Constraint-based:** We explained in Section 2.4 that depending on the type of system feedback, queries can be handled as a set of parametric constraints or as a set of parametric utility functions. Thus, **we define a type of system feedback to be *constraint-based* if it handles queries as a set of parametric constraints**. In order to build decision confidence, users are more likely to use feedback that is based on constraints rather than feedback based on utility functions. That is, as we have seen in Section 2.3.3, Tversky argues in [3] that people are reluctant to accept the principle that decisions should depend on computations based on subjective estimations of utilities in which the decision maker herself has only limited confidence, and thus decision strategies such as the *weighted additive* (WADD) strategy do not provide decision confidence (see Section 2.3.3). The author explains that people are more confident with a decision strategy that involves no numerical computations, and that is easy to explain and justify. The author argues that the use of the *elimination-by-aspect* (EBA) decision strategy motivates at the end of the decision process a *state of mind* that qualitatively justifies one's decision, achieving confidence in its rationality. This state of mind, or *mental model*, is based on a sequence of *aspects* (constraints) which leads to a unique candidate item from the catalog (see Section 2.3.2)[4]. However, the main drawback of EBA is that it heavily penalizes decision accuracy.

2. **Universally valid: We define a type of system feedback to be *universally valid* if it remains valid (for a given catalog) independent of the query for which it was produced**. For instance, given the catalog of Example 1.1 and the query "price $\leq$ 6'500 Euros AND mileage $\leq$ 50'000 km AND color = blue AND airbag = yes", the system feedback "there are not items with price $\leq$ 6'500 Euros AND mileage $\leq$ 50'000 km" remains valid for the given catalog independent of the given query. On the other hand, given the same catalog and query, the system feedback "the best item according to your query is item #10" is valid only for the given query. Therefore, universally valid feedback is potentially more reusable for the decision maker who can compile feedback in order to construct a decision justification.

3. **Negatory: We define a type of system feedback to be *negatory* if it provides sets of conjunctive constraints that *no* items in the catalog satisfy**

[4]However, the main drawback of EBA is that it heavily penalizes decision accuracy.

(or if it is straightforward to derive such sets). For instance, the feedback "There are no items with price $\leq$ 6'500 Euros AND mileage $\leq$ 50'000 km" is a set of two conjunctive constraints that none of the items in the catalog of Example 1.1 satisfies. Indeed, researchers have shown that people have the tendency to search evidence that confirms their hypothesis under study, rather than evidence that could disconfirm their hypothesis [70]. This tendency is called *confirmation bias* and it is a form of cognitive bias in collecting evidence. In our framework, the confirmation bias implies that decision makers will potentially search for evidence of why there is *not* a better choice in order to build a justification that confirms their tentative decision. Thus, in order to improve decision confidence, *negatory* feedback may potentially be more useful than feedback which helps finding better choices.

4. **Unique in its class**: In Section 2.4 we defined several classes of system feedback, such as ranked list, system-proposed critiquing, conflict and corrective feedback (See Table 2.2 on page 31). **We define a type of feedback to be *unique in its class* if there exists no other type of feedback belonging to the same class that produces contradictory results given the same query**. For instance, there are several ways to compute a ranked list (e.g. based on utility, similarity or pareto-optimality) which may produce distinct results for the same query, and thus we say that ranking feedback is not unique in its class. Feedback that is not unique in its class may not contribute to building confidence as they may lack credibility. For instance, imagine a user that visits two different websites that have the same catalog and for the same query she gets two different rankings: which one should she take into account?

Of course, a type of feedback that improves decision confidence while heavily penalizing decision accuracy would not be acceptable. Note also that there might be other such properties, such as *system transparent* and *argumentative.* However, the precise study of such properties is out of the scope of the present work.

3.3.3 Analysis of feedback types using the DECOFE framework

In this Section, we analyze several types of system feedback described in Section 2.4, in terms of the potential contribution to decision confidence building.

Ranked list feedback
The ranked list type of feedback is a compensatory strategy that balances conflicting objectives by producing a score for each item in the catalog. The score is then used to sort the list of items. Rankings can be based on different criteria, such as utility, similarity or pareto-optimality. For instance, the *weighted additive* (WADD) decision strategy achieves this by combining individual preferences into an overall utility for

each item, and it guarantees good decision accuracy (see Section 2.3.1). In this case, it involves computations based on subjective estimations of utilities which decision makers themselves have only limited confidence in [3].
Having one tentative decision in mind, this type of system feedback is difficult to use in finding evidence supporting such a decision. For instance, taking the example of Section 1.3.2, a justification like "car #10 is the best car with a score of 90 compared to a score of 88 for the second best option, given that for color I gave 100 points for blue, 90 for black, 50 for white and 25 for red, for price. . . " is intuitively not convincing. According to the DECOFE framework, such feedback is not *universally valid* as it remains valid only for the query for which it was actually produced. Neither is it *negatory* feedback, as it does not provide arguments about what is *not* in the database (a ranked list is just a reordering of the items in the catalog). As different types of ranked list feedback (e.g. based on utility, similarity or pareto-optimality) produce different rankings, it is not *unique in its class*. Thus, according to the DECOFE framework, ranked list feedback is not suitable for achieving good decision confidence.

System-proposed critiquing feedback
In system-proposed critiquing [48, 52, 49, 64], the system predicts a set of unique or compound critiques that the user may be prepared to accept in order to improve the preference model and find better items. Therefore, this type of system feedback helps to discover news items rather than to confirm the current tentative decision. Consider, for example, the system feedback "apart from your tentative decision, there are also cheaper cars with more mileage". This feedback stimulates the user to keep searching for another choice rather than committing to the current one.
According to the DECOFE framework, such feedback is not *universally valid* as it remains valid only for the query for which it was actually produced. Neither is it *negatory* feedback, as it does not provide arguments about what is *not* in the database (system-proposed critiquing compares the current item with the rest of the items in the catalog). It is not *unique in its class* as different algorithms exists (See Section 2.4.5) that produce distinct results for the same given query.

Conflict feedback
Conflict feedback can be seen as the opposite of system-proposed critiquing feedback, in the sense that it does not explain what other items are available, but rather explains why there are not items corresponding to the user's query. This type of system feedback helps to confirm one's tentative decision, rather than discover new items. Consider, for example, a user that asked for cars cheaper than 9'500 Euros, with 50'000 km mileage maximum, blue and with the airbag option. Using the catalog of Example 1.1, the system would show only one matching car, which costs 9'238 Euros and has 17'300 km. In order to make sure that is the best car, the user could ask to restrict the price to 9'000 Euros. The system would then explain that there are no cars in the catalog for 9'000 Euros and 50'000 km mileage maximum (one minimal conflict), nor cars for 9'000 Euros, blue and with the airbag option (a second minimal conflict). This explanation

gives evidence to the user that confirms that her tentative choice is a good choice[5]. According to the DECOFE framework, this type of feedback is *constraint-based.* It is also *universally valid*, as the identified conflicts remains valid independent of the query for which they were produced. That is, the previous explanation "there are not cars in the catalog for 9'000 Euros and 50'000 km" is true (given the catalog) during the whole decision process. It is also *negatory* feedback, as it provides arguments about what is *not* in the database in the form of failing sub-queries. Given a query, the minimal conflict set is unique (i.e. no distinct correct results can be produced for the same query), and thus conflict feedback is *unique in its class.* Therefore, according to the DECOFE framework, this type of system feedback potentially helps users to build decision confidence.

Corrective feedback
Conflict feedback is very useful to construct a decision justification. However, it does not explain what the options for solving the conflicting objectives are. This is the purpose of corrective feedback[6]. Consider the following example using the catalog of Example 1.1: given that the user asks for cars cheaper than 9'000 Euros, with 50'000km maximum, blue and with the airbag option, the system would explain that she needs to either relax the maximum price to 9'238 Euros (instead of 9'000 Euros), or to relax the maximum mileage to 55'700 km (instead of 50'000 km). This type of system feedback can be interpreted as a clear ad-hoc trade-off question. By deciding upon this tradeoff, the user is providing herself a justification for her decision.
As in the case of conflict feedback, corrective feedback is also *constraint-based.* It is also *negatory* feedback, as for each relaxation it is straightforward to derive a set of conjunctive constraints that no items in the catalog satisfy. That is, in the previous example, the explanation that the user needs to relax the maximum price to 9'238 Euros naturally implies that there are no cars for less than 9'238 Euros, with 50'000km maximum, blue and with the airbag option. Also, given a catalog and a query, there cannot be two contradictory corrective feedbacks (for instance, in the previous example, saying that the user needs to relax the maximum price to a value other than 9'238 Euros would be incorrect), and thus corrective feedback is *unique in its class* according to the DECOFE framework. However, it is not *universally valid*, i.e. previous feedback is invalidated when the query changes.

We summarize the given analysis in Table 3.1. Conflict and corrective feedback seem the most promising type of system feedback to potentially contribute to decision confidence building.

[5]Note however, that being confident about one choice does not necessarily imply that this choice is the best one.
[6] *Corrective* explanations are also called *query reformulations* or *query relaxations.*

	Constraint-based	**Universally valid**	**Negatory**	**Unique in its class**
Ranked list	Depends on the alg.	No	No	No
System-proposed critiquing	Depends on the alg.	No	No	No
Conflict	Yes	Yes	Yes	Yes
Corrective	Yes	No	Yes	Yes

Table 3.1: Analysis of different types of system feedback in terms of the potential contribution to decision confidence building.

3.4 Confidence Building Tradeoff Explanations (CBTE)

As reported in Section 2.4.6, previous studies about corrective and conflict feedback have focused on designing algorithms to reduce their time complexity [28, 30, 55] rather than evaluating their value in terms of decision accuracy or decision confidence. In the previous Section, we found out that conflict and corrective feedback are good starting points in designing a type of feedback that potentially contributes to decision confidence building. In this Section, we design a new type of system feedback, called *confidence building tradeoff explanations* (CBTE), combining (1) corrective feedback (or *global relaxations*), (2) conflict feedback (or the *minimal conflict set*) and (3) a new type of feedback called *local relaxations*, which combines the advantages of both global and conflict feedback. Here we present the general idea under CBTE, while a formal description is given in the next section.

Let's first introduce an example of a failing query that we are going to use throughout this chapter.

Example 3.1 A failing query (given the catalog of Example 1.1), written in informal language.

price <= 6'500 Euros AND
mileage <= 50'000 km AND
color = blue AND
airbag = yes

Minimal conflict set

Given a failing query q, the *minimal conflict set* (MCS) is the set of all the minimal subqueries of q for which there are no satisfying items in the catalog. The main advantage of the MCS is that it fulfills the four properties defined in the DECOFE framework: the MCS is based on *constraints*; it is *universally valid*, i.e. it remains valid throughout the decision process; and it explains what is not in the catalog. It is *unique in its class.* A potential drawback of MCS is that it increases exponentially with the size of the query. Below is the minimal conflict set for the query in Example 3.1:

Example 3.2 The minimal conflict set for the query in Example 3.1, written in informal language.

There are not items with:
price <= 6’500 Euros AND mileage <= 50’000 km

There are not either items with:
price <= 6’500 Euros AND color = blue AND airbag = yes

The authors in [28, 30, 55] argued that the MCS is useful because it provides a sound and minimal explanation of why a query is a failing query. While we agree with this argument, the authors did not explain in what sense it is useful (to improve decision accuracy, decision confidence or something else), nor did they confirm its usefulness with a user study. We discussed in the previous section how MCS can improve decision confidence, the validation of which is the purpose of the next chapter.

Global relaxation set

Query relaxations were already proposed in [28], although they were quite simplistic. [30, 55] proposed to provide reformulations of the query as similar as possible to the one given by the user, while guaranteeing to turn it in a successful query. We call this type of query reformulation a *global relaxation*, and a set of global relaxations as the *global relaxation set*. Below there is one example of a global relaxation set for the query of Example 3.1:

Example 3.3 A global relaxation set for the query in Example 3.1, written in informal language.

You have three options to have items matching your query:
1) change price <= 9’238 Euros (instead of 6’500 Euros)
2) change mileage <= 55’700 km (instead of 50’000 km) AND discard color
3) change mileage <= 76’000 km (instead of 50’000 km) AND discard airbag

The global relaxation set can be interpreted as a clear ad-hoc trade-off question, such as “do I prefer to pay 400 Euros more, or to accept a car that is one year older without the anti-lock braking system (ABS)?” and can be therefore be easily internalized provided that they are relatively short. However the global relaxation set size might increase even faster than the MCS depending on the size of the query. We thus propose to limit this explosion by selecting the most representative relaxations. This issue is discussed in detail in Section 3.5.3. While the global relaxation set may seem more useful than the MCS, it is not *universally valid*, e.g. the statement “there are items if you change price < 10’400 Euros in your query” is invalidated as soon as the query changes.

Local relaxation set

The main drawback of the global relaxation set is that it is not *universally valid*, and it is therefore less likely to contribute to building a decision justification. For this purpose, we introduce the concept of *local relaxation*, a relaxation that solves a particular conflict in the MCS. Given a conflict of the MCS, the *local relaxation set* (LRS) is the set of relaxations that solve this specific conflict[7]. As the MCS is universally valid, its local relaxation sets are also universally valid. Below, in Example 3.4, there is the local relaxation set for the first conflict in the MCS of Example 3.2.

Example 3.4 The local relaxation set for the first conflict in the MCS in Example 3.2, written in informal language.

There are no items with:
 price <= 6'500 Euros AND mileage <= 50'000 km

You have two options for solving this conflict:
 1) change price <= 9'238 Euros (instead of 6'500 Euros)
 2) change mileage <= 55'700 km (instead of 50'000 km)

As in the global relaxation set, the local relaxation set provides a clear ad-hoc question, such as "Do I prefer to pay 50 Euros more or to retract about BMW cars?", with the advantage that it is universally valid and therefore it has more chances to be used for the construction of a decision justification. Note that relaxations in the LRS do not generally match relaxations in the GRS. In this example, the relaxation "change price $\leq$ 9'238 Euros" is present both in the LRS and the GRS. However, while the local relaxation "change mileage $\leq$ 55'700 km" is enough to solve the first conflict, it is not enough to derive a successful query. In order to do so, the previous relaxation needs to be complemented by either discarding the constraint about color or by discarding the constraint about the airbag (See Example 3.3).

Confidence Building Tradeoff Explanations (CBTE)

We have presented so far three types of complementary system feedback with their advantages and disadvantages, which we group under the label of *confidence building tradeoff explanations* (CBTE). In typical situations where there is a mismatch between what the decision maker wants and what is it in the catalog, CBTE provides a clear trade-off question to help the user make a good decision through the *global relaxation set*, and provides evidence of building decision confidence through the *minimal conflict set* and the *local relaxation set*.

A potential drawback of CBTE is that its size can increase exponentially with the size of the query and therefore make it difficult for the user to be internalized. To evaluate the

[7]Please note the difference between a global relaxation and a local relaxation: a global relaxation solves all conflicts of a failing query at once and transforms this failing query into a successful query, while a local relaxation solves only one given conflict with the effect that other conflicts may still remain preventing the failing query from being transformed into a successful query.

effect of this potential drawback, we ran two simulations (see Section 3.6) and completed a user study (see Chapter 4).

CBTE and the parametric weighted additive (PWADD) both provide similar information about the conflicting objectives in two different forms. PWADD, on the one hand, provides this information in terms of a ranked list, and on the other hand, CBTE lists queries conflicts and query relaxations. Therefore we hypothesize that both approaches will achieve similar decision accuracy[8]:

However, CBTE is expected to have an advantage over PWADD in terms of an increase in decision confidence and perceived ease of use.

3.5 Formalization of CBTE

3.5.1 User query

A catalog is a set of *item descriptions*, such as products or services. Each item is described by a predefined set of *attributes*, where an attribute is an abstraction of a characteristic of an *item*, such as “price”, “color” or “delivery time”. Formally, we define a *catalog* as a tuple *(A, D, O)* where $A = \{A_1,\dots A_n\}$ is a finite set of attributes, each associated with a domain $D = \{D_1,\dots D_n\}$ and O is a set of available item descriptions. If the domain values of an attribute a have an order, then a is said to be an *ordered attribute*. Given an item $o \in O$, $a_i(o)$ represents the domain value of attribute A_i for item o.

We define a query here as a filtering operation over the set of items. For the sake of simplicity, we restrict a query to be a set of constraints on individual attributes, although more complex queries can be envisaged. Formally, we define a *query* as a set of constraints on individual attributes. A *constraint* defined over an attribute A_i indicates which domain values of attribute A_i are valid. Concretely, a constraint cr is a tuple *(p, a, x)* where a is an attribute with $a \in A$, x is a domain value on the domain of a defined by the user and an operator $p \in \{=, \leq, \geq\}$ which, given an item o, is satisfied if $a(o)\ p\ x$. If p is $\leq$ or $\geq$, then the a needs to be an ordered attribute. An item satisfies a query if it satisfies all the constraints of the query. The size of a query q corresponds to the number of constraints in q. The set of items in O that satisfy a query q are called the

[8]PWADD uses parametric individual utility functions to produce an overall utility. It has the advantage over WADD in that qualitative preferences are easier to elicit. We suggest that CBTE and PWADD provide similar type of tradeoff information and they should achieve similar decision accuracy. However, previous studies [62] have shown that decision accuracy in PWADD is low, at around 40%, and thus CBTE may not be very satisfactory in terms of decision accuracy either. This bad performance in decision accuracy in PWADD (also expected in CBTE) comes from the fact that users state only a very partial preference model.

matching items of q over O[9]. When no items satisfy the query, we say that it is a *failing query*[10].

A *query system* is a DSSEC that allows users to enter queries and returns the matching items.

Example 3.5 A successful query.

Given the catalog of Example 1.1, a query could be the set of constraints:
$q = \{cr_{price}, cr_{mileage}, cr_{airbag}\}$, where cr_{price} = *(price, $\leq$, 10'000 Euros), $cr_{mileage}$ = (mileage, $\leq$, 100'000 km) and cr_{airbag}=(airbag, =, yes).*

The query is the conjunction of all the constraints, so it filters the items on the catalog that satisfy the expression "price $\leq$ 10'000 Euros AND mileage $\leq$ 100'000 km AND airbag = yes". In the catalog there are 5 out of 40 items matching the query.

Example 3.6 The failing query of Example 3.1.

The query of Example 3.1, "price $\leq$ 6'500 Euros AND mileage $\leq$ 50'000 km AND color = blue AND airbag = yes" can be formally written as follows:

$q = \{cr_{price}, cr_{mileage}, cr_{color}, cr_{airbag}\}$, where cr_{price} = (price, $\leq$, 6'500 Euros), $cr_{mileage}$ = (mileage, $\leq$, 50'000 km), cr_{color}=(color, =, blue) and cr_{airbag}=(airbag, =, yes).

Example 3.7 Another failing query.

Given the catalog of Example 1.1 and the query $q = \{cr_{price}, cr_{mileage}, cr_{airbag}\}$, where cr_{price} = (price, $\leq$, 5'000 Euros), $cr_{mileage}$ = (mileage, $\leq$, 100'000 km) and cr_{airbag}=(airbag, =, yes):

q is a failing query because there are not items that satisfy this query.

A CBTE explanation consists of the *global relaxation set* (GRS), the *minimal conflict set* (MCS) and the *local relaxation set* (LRS) for each conflict in the MCS and is described in the following sections.

3.5.2 Minimal conflict set

When a query is overconstrained, the authors in [28] proposed to show an explanation of the cause of query failure in order to have a more cooperative query system.

[9] If the context is unambiguous, q over O is removed for clarity.

[10] The terms *failing query*, *unsuccessful query*, *query with empty answers* or an *over-constrained query* are used interchangeably.

Given a failing query q and the set of items O, a conflict cf of q over O is a sub-query of q ($cf \subseteq q$) for which there are no satisfying items in O. A conflict cf is a *minimal conflict* if any strict subset of cf is not a conflict. The *minimal conflict set* (MCS) is the set of all the minimal conflicts of q over O.
See a minimal conflict set in Example 3.2.

Computation complexity: Given a catalog and a failing query, the problem of computing its MCS is NP-hard [18]. Figure 3.2 shows an exhaustive search algorithm to compute the MCS requiring $O\left(\sum_{r=1}^{n}\binom{n}{r}\right) = O\left(2^n\right)$ conflict tests[11].

```
algorithm computeMCS(catalog, query)
begin
$MCS \leftarrow \emptyset$
for $k = 1$ to size(query) do
    subqueries ← getSubqueries(k, query)
    for each s in subqueries do
        if s $\notin$ MCS $\wedge$ isConflict(catalog, s)
            $MCS \leftarrow MCS \cup s$
    end
end
return MCS
end algorithm
```

getSubqueries(*k*, *query*)
gives the set of all subqueries of *query* with cardinality k.

isConflict(*catalog*, *s*)
is true if *s* is a failing query over *catalog*.

Figure 3.2: Algorithm to compute the Minimal Conflict Set in exponential time.

Example 3.8 A simple Minimal Conflict Set.

The Minimal Conflicting Set of Example 3.7 is MCS=$\{\{cr_{price}, cr_{airbag}\}\}$.

It means that there is only one conflict, represented by the expression "price $\leq$ 5'000 Euros AND airbag = yes". It shows that the criterion about mileage is not part of the conflict, and thus, relaxing this criterion would not contribute in solving the conflict.

[11]While in this thesis we are not concerned with time complexity, we note that the MCS can be computed efficiently by methods such as the QuickXPlain algorithm [25] or the use of *assumption-based CSPs* and precompiled CSPs [26]. See Section 3.8 for more information.

Example 3.9 A more complex Minimal Conflict Set.

The Minimal Conflicting Set of Example 3.6, shown informally in Example 3.2, can be formally written as follows:

MCS=$\{\{cr_{price},\ cr_{mileage}\},\ \{cr_{price},\ cr_{color},\ cr_{airbag}\}\}$.

3.5.3 Relaxations

A relaxation is an operation that turns a failing query or a minimal conflict into a successful query.

Definition: Given a catalog *q* and a set of constraints *q* (either a failing query or a minimal conflict) we define a **relaxation** *r* as a function of *q* that produces *q'* such that *q'* is a successful query on *d*. We say that *r* solves the query *q*.

Example 3.10 A query relaxation.

A relaxation for the failing query of Example 3.6 is to change the maximum acceptable mileage to 55'700 km (instead of 50'000 km) and to discard the constraint about blue color. If we apply this relaxation, the updated query becomes:

q' = $\{cr_{price},\ cr_{mileage},\ cr_{airbag}\ \}$, where cr_{price} = (price, $\leq$, 6'500 Euros), $cr_{mileage}$ = (mileage, $\leq$, 55'700 km) and cr_{airbag}=(airbag, =, yes)

Informally, *q'* is "price $\leq$ 6'500 Euros AND mileage $\leq$ 55'700 km AND airbag = yes".

Local Relaxation

We define a **Local Relaxation** as a relaxation that solves a minimal conflict. We distinguish two types of local relaxations: (1) those that remove a constraint from the query and (2) those that adjust a constraint in the query.

Given a minimal conflict, a basic type of relaxation is to remove one of the constraints of the conflict. We call this type of relaxation **Constraint Removal Relaxation** (CRR).

Definition: Given a set of constraints *q* (either a query or a minimal conflict) and a constraint cr_i = (*p*, *a*, *x*) *q*, a RCR is a *q'* such that *q'* = *q* – $cr_i$and *q'* becomes a successful query. A CCR is read as "**remove constraint** $\boldsymbol{cr_i}$".

Given a set of constraints *q* (either a query or a minimal conflict) and a constraint cr_i = (*p*, *a*, *x*) *q* where p is $\leq$, a type of more concrete relaxation is to "replace *x* by *x'* ", where that *x'* is a value such that *q'* becomes a successful query. We call this type of relaxation **Constraint Adaptation Relaxation** (CAR). Similar when p is $\geq$.

Example 3.11 A *Remove Constraint Relaxation.*

The second conflict of the MCS={{cr_{price}, $cr_{mileage}$}, {cr_{price}, cr_{color}, cr_{airbag}}} of the failing query Example 3.6 is { cr_{price}, cr_{color}, cr_{airbag} }, which explains that there are not items with price $\leq$ 6'500 Euros AND color = blue AND airbag = yes.

A remove constraint relaxation for this second conflict in the MCS is "remove constraint cr_{price}". By removing the constraint about price, the second conflict is solved, i.e. there exist at least one item (seven cars in this example) with color = blue AND airbag = yes in the catalog.

Definition: Given a set of constraints *q* (either a query or a minimal conflict) and a constraint cr_i = (*p*, *a*, *x*) *q* where p is $\leq$ (or $\geq$), a *Constraint Adaptation Relaxation* (CAR) is a q' such that q' = (*q* - cr_i) cr'_i, where cr'_i=(*p*, *a*, *x'*) and *x'* is a value in the domain of *a* such that *q'* becomes a successful query. Given *q*, cr_i = (*p*, *a*, *x*) *q*, a CAR *q'* = (*q* - cr_i) cr_i*'* where cr_i'=(*p*, *a*, *x'*) and another CAR *q"* = (*q* - cr_i) $cr"_i$ where $cr"_i$=(*p*, *a*, *x"*), CAR *q' dominates* CAR *q"* if —*x-x'*— < —*x-x"*—. A CAR is read as "**adapt constraint cr**$_i$".

Example 3.12 A *Constraint Adaptation Relaxation.*

Taking the same Example 3.11, a constraint adaptation relaxation for the second conflict in the MCS is "adapt constraint cr_{price}", where *x'* would be 9'238 Euros.

By adapting the constraint about price, the second conflict is solved, i.e. there exist at least one item (car #10) with price $\leq$ 9'238 Euros AND color = blue AND airbag = yes in the catalog.

We define a *relaxation* as being either a CRR relaxation or a CAR relaxation. By definition, we say that a CAR always dominates a CRR, in the sense that it contains more information to solve the conflict.

Definition: A relaxation r' *dominates* a relaxation *r"* if *r'* and *r"* are related to the same constraint cr_i and, if *r'* is a CAR relation then *r"* is a CRR relaxation, or if they both are CAR then *r'* dominates *r"*.

We define the Local Relaxation Set (LRS) as the set of non-dominated local relaxations that solves a minimal conflict *cf*.

Definition: Given a catalog *d* and a minimal conflict *cf* over *d*, we define the Local Relaxation Set of *cf* LRS={r_1,... r_n} as the set of all non-dominated relaxations.

Computation complexity: Given a catalog and a minimal conflict, it takes linear time to compute its LRS, i.e. the time it takes is proportional to the size of the minimal conflict. Figure 3.3 shows an algorithm to compute the Local Relaxation Set in linear time.

Example 3.13 A *Local Relaxation Set.*

The second conflict of the MCS=$\{\{cr_{price}, cr_{mileage}\}, \{cr_{price}, cr_{color}, cr_{airbag}\}\}$ of the failing query Example 3.6 is $\{ cr_{price}, cr_{color}, cr_{airbag} \}$, which explains that there are not items with price ≤ 6'500 Euros AND color = blue AND airbag = yes.

The local relaxation set for this second conflict is as follows:
LRS = $\{r_1, r_2, r_3\}$, where r_1="adapt cr_{price}", r_2="remove cr_{color}" and r_3="remove cr_{airbag}".

It states that there are three options for solving the second conflict: (1) change the maximum acceptable price to 9'238 Euros (instead of 6'500 Euros), (2) to discard the constraint about price or (3) to discard the criteria about airbag.

```
algorithm computeLRS(catalog, mincf)
begin
LRS ← ∅
For each cr = (p, a, v) in mincf do
    if p is ≤ or ≥ do
        LRS ← LRS ∪ "adapt constraint cr"
    else
        LRS ← LRS ∪ "remove constraint cr"
    end
end
return LRS
end algorithm
```

"adapt constraint *cr*",
where $cr = (p, a, x)$ and p is ≤,
is to replace x by $x' = \min(\{a(o)$, for all items o in the catalog that satisfy the query $q = mincf - cr\})$

Figure 3.3: Algorithm to compute the Local Relaxation Set, given a catalog and a minimal conflict.

Global Relaxation

We define a Global Relaxation (GR) as a set of CRR or CAR relaxations that applied together turn a failing query into a successful query. However, we limit the number of CAR relaxations to zero or one. This is because the number of all non-dominated relaxations can increase very quickly if there is more than one CAR relaxation, while showing only the extreme ones already gives a good overview. This can be better understood in the example of Figure 3.4.

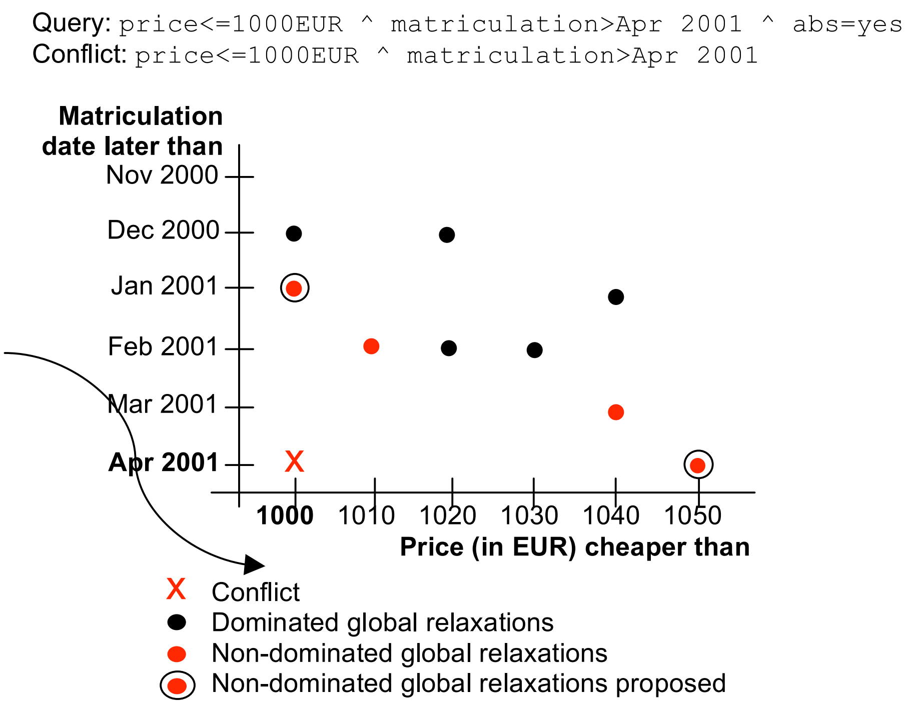


Figure 3.4: Global relaxation example.

Definition: Given a failing query q, a Global Relaxation $gr = \{r_1, \ldots r_n\}$ where r_i is either a CRR or CAR *relaxation* and gr contains at most one CAR relaxation, such that applying all the relaxation r_i to q produces a successful query q'.

Given two global relaxation *gr'* and *gr"*, *gr'* is not dominated by a *gr"* if not all the relaxations in *gr"* are in *gr'*, or at least one relaxation in *gr'* is not dominated by any of the relaxations in *gr"*.

Definition: Given a catalog d and a failing query q over d, we define the Global Relaxation Set of q, GRS=$\{r_1, \ldots r_n\}$, as the set of all non-dominated global relaxations.

Example 3.14 A *Global Relaxation Set.*

The query of Example 3.6 is:
$q = \{cr_{price}, cr_{mileage}, cr_{color}, cr_{airbag}\}$, where cr_{price} = (price, $\leq$, 6'500 Euros), $cr_{mileage}$ = (mileage, $\leq$, 50'000 km), cr_{color}=(color, =, blue) and cr_{airbag}=(airbag, =, yes).

The Global Relaxation Set is as follows:
GRS={{adapt cr$_{price}$}, {adapt cr$_{mileage}$, remove cr$_{color}$}, {adapt cr$_{mileage}$, remove cr$_{airbag}$}}.

It states that there are three options to turn q into a successful query: (1) change the maximum acceptable price to 9'238 Euros (instead of 6'500 Euros), (2) change the maximum acceptable mileage to 55'7000 km (instead of 50'000 km) and discard the criteria about color, or (3) change the maximum acceptable mileage to 76'000 km (instead of 50'000 km) and discard the criteria about airbag.

Computation complexity: Given a catalog and a failing query, the problem of computing its GRS is NP-hard [18]. Figure 3.5 shows an exhaustive search algorithm to compute the GRS requiring $O\left(\sum_{r=1}^{n}\binom{n}{r}\right) = O\left(2^n\right)$ conflict tests[12].

3.6 Simulations

We explained that the size of the CBTE feedback grows exponentially with the size of the query. Feedback that is too long would require a high cognitive effort for the user to process it and thus it would become practically useless. In this Section we study the size of CBTE feedback through two simulations. For this purpose, we used a real-life catalog of 200 used cars[13].

In the first simulation, we generate one thousand random failing queries, with query sizes uniformly distributed from one to fifteen. To do so, we first randomly selected the size of the query, n. Then, we randomly selected n attributes. Each of these attributes was associated to a parametric constraint ("equal", "less or equal than" or "more or equal than") as shown in Appendix 1, "Second-hand cars database" on page 132. For each of these parametric constraints, we randomly assigned a value from the attribute range. The query was then the conjunction of these parametric constraints. If the query was successful (i.e., there are items in the catalog satisfying the query), we dropped the query. Otherwise, we kept the query. We repeated the process until we had one thousand failing queries. We noticed that only 24.5% of the queries were successful.

[12]Similar to MCS, the GRS can be computed efficiently by methods such as the use of *assumption-based CSPs* and precompiled CSPs [26]. See Section 3.8 for more information.

[13]This catalog is used in the next chapter; see Section 4.3 for a detailed description.

```
algorithm computeGRS(catalog, query)
begin
GRS ← $\emptyset$
for $k$ = 1 to size(query) do
    subqueries ← getSubqueries(k, query)
     for each $r$ in subqueries do
          if $r \notin GRS \wedge$ isConflict(catalog, query - r)
               GRS ← GRS $\cup$ computeGRS2(catalog, query, r)
     end
end
return MCS
end algorithm

algorithm computeGRS2(catalog, query, relaxation)
where relaxation={$cr_1$,...$cr_n$}
begin
GRS2 ← $\emptyset$
For each $cr_i$= (p, a, x) in relaxation where p is $\leq$ or $\geq$ do
          GRS2 ← GRS2 $\cup$ {{$r_1$,...$r_n$} - $r_i \cup$ $r'_i$}, where
            n = sizeof(conflict) and
            $r_i$ = "remove constraint $cr_i$" and
            $r'_i$ = "adapt constraint $cr_i$"
     end
end
if GRS2 is $\emptyset$ do
          GRS2 ← GRS2 $\cup$ {{$r_1$,...$r_n$}}, where
            n = sizeof(conflict) and
            $r_i$ = "remove constraint $cr_i$"
end
return GRS2
end algorithm
```

"adapt constraint *cr*",
where $cr = (p, a, x)$ and p is $\leq$,
is to replace x by x' = min({$a(o)$, for all items o in the catalog that satisfy the query q = *query – relaxation*})

getSubqueries(*k*, *query*) and **isConflict(*catalog*, *s*)** are defined in the previous section.

Figure 3.5: Algorithm to compute the Global Relaxation Set, given a catalog and a failing query.

This simulation however, did not intuitively correspond to human behavior and let to a second simulation.

In this second simulation one thousand random failing queries were built in an incremental fashion. To do this, we first randomly selected an attribute, again associated with a parametric constraint, and we randomly assigned a value from the attribute range. We kept adding constraints to the query, until it became a failing query. We then kept this query and repeated the process until we had one thousand failing queries. Through this simulation, we noticed that 78% of the queries were successful.

The global relaxation set consists of a set of global relaxations. It can be seen as a trade-off consisting of several choices, i.e. the user needs to choose which relaxation is more appropriate, if any. We suggested that this type of trade-off information could be very valuable to the user because they represent clear ad-hoc questions, like "Do you prefer to pay 80 Euros more, or to accept a one year older car without the ABS feature?". However, if users are overwhelmed with a large global relaxation set, this information would demand a high cognitive effort to process it and it would become practically useless. Figure 3.6 shows the distribution of the size of the global relaxation set. In the case of the uniform simulation, it shows that the GRS can be quite large and thus it would not be useful for the user. On the other hand, in the case of the incremental simulation 90% of the GRSs consisted of only two or three global relaxations.

While the global relaxation set consists of a set of global relaxations, a global relaxation consists of a set of constraints to be relaxed (thus defining the size of a global relaxation). For instance, the second global relaxation of Example 3.3 on page 62 is to "change mileage to less than 55'700 km (instead of 50'000 km) and to discard the constraint about color". Figure 3.7 shows the distribution of the size of the global relaxations. Again, in the case of the uniform simulation, it shows that global relaxations can be too big to be useful. In the case of the incremental simulation, it shows that 80.4% of the global relaxations consist of only one constraint to relax, making it very attractive.

The minimal conflict set consists of a set of minimal conflicts. It shows precisely which parts of the query are causing the conflicts, and thus it clearly explains what it is not in the catalog. Figure 3.8 shows the distribution of the size of the minimal conflict set. As in the GRS, in the case of the uniform simulation, the size of the MCS can be too big to be useful. In the case of the incremental simulation, the figure shows that 68.8% of the MCSs consist of only one conflict.

While the minimal conflict set consists of a set of conflicts, a conflict consists of a set of constraints (thus defining the size of a conflict). For instance, the first conflict of Example 3.2 on page 62 is the lack of cars in the catalog with "price cheaper than 6'500 Euros and mileage of less than 50'000 km". Figure 3.9 shows the distribution of the size of the conflicts. In both cases, 95% of the conflicts consist of up to four constraints.

In summary, our work suggests that simulations using queries uniformly distributed in

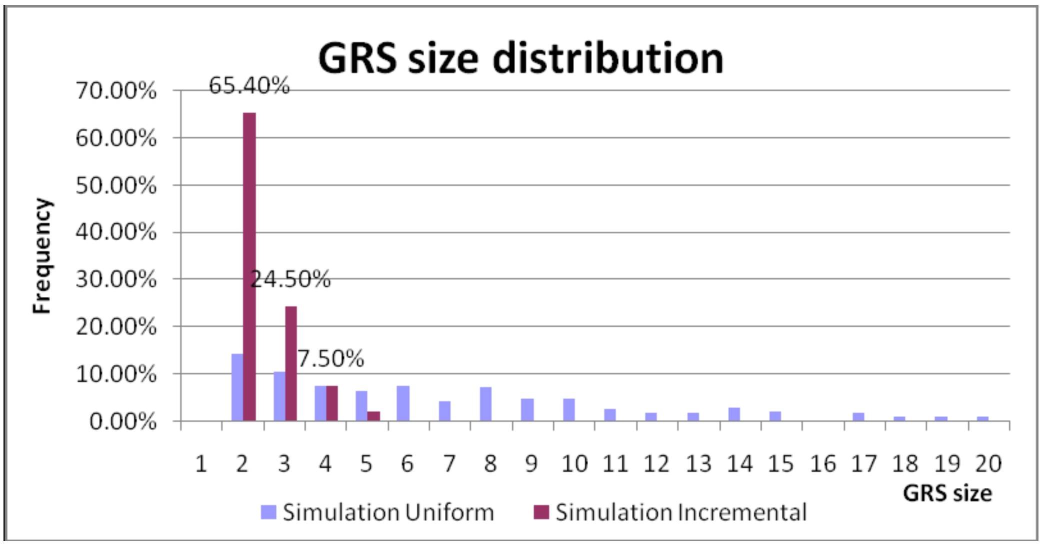


Figure 3.6: Results of the simulation. Distribution of Global Relaxation Set sizes. This graph shows 100% of the points in the case of the incremental simulation. It only shows 85% of the points in the case of the uniform simulation, where the biggest GRS has size 75.

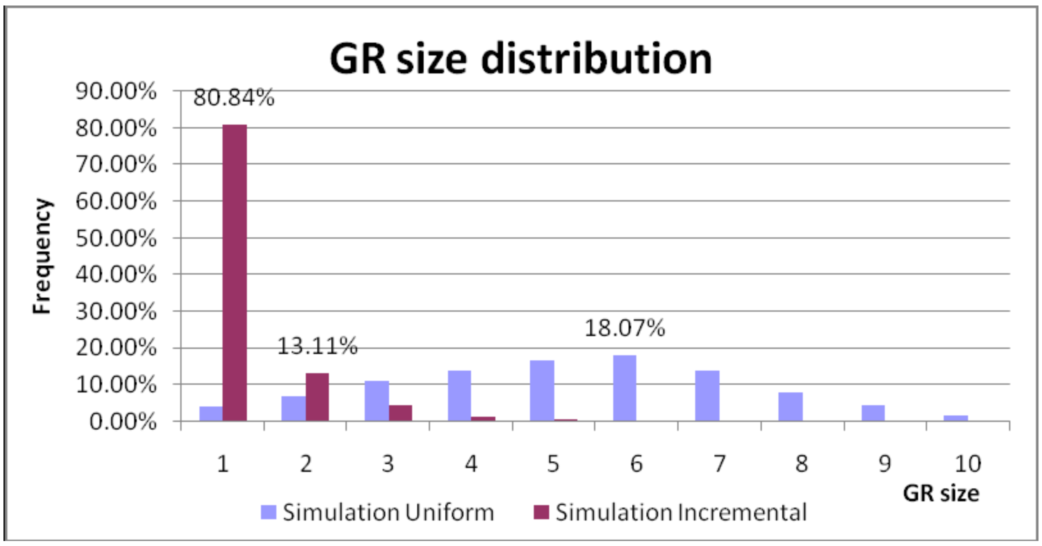


Figure 3.7: Results of the simulation. Distribution of Global Relaxation sizes.

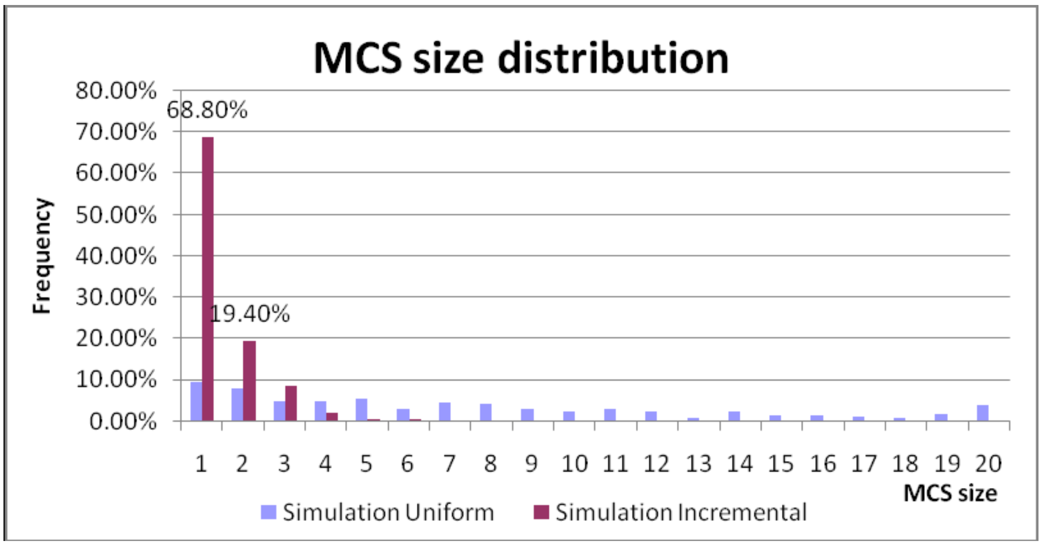


Figure 3.8: Results of the simulation. Distribution of Minimal Conflict Set sizes. This graph shows only 71% of the points in the case of the uniform distribution, where the biggest MCS has size 101.

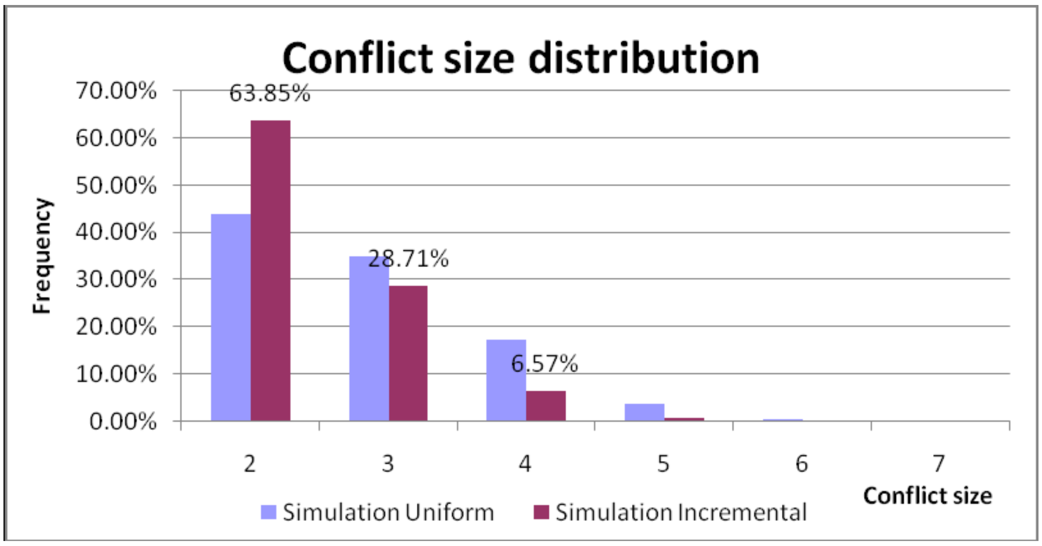


Figure 3.9: Results of the simulation. Distribution of the conflict sizes.

size leads to quite unsatisfactory results with respect to the size of the system feedback, while simulations using queries built in an incremental fashion lead to much more promising results. This study therefore illustrates that results obtained through simulation may heavily depend on how queries are constructed. This confirms the fact that the use of simulations for the study of human behavior need to be considered with care and that empirical studies, although quite expensive, are necessary for a reliable evaluation of human-machine interaction systems.

We carried out a pilot study with several common every-day business-to-consumer catalogs: movies, apartments and university lectures. We noticed that the behavior of real users in stating queries was much closer to the incremental simulation, with encouraging results. However, as we said previously, a correct evaluation requires a user study, which is the purpose of Chapter 4.

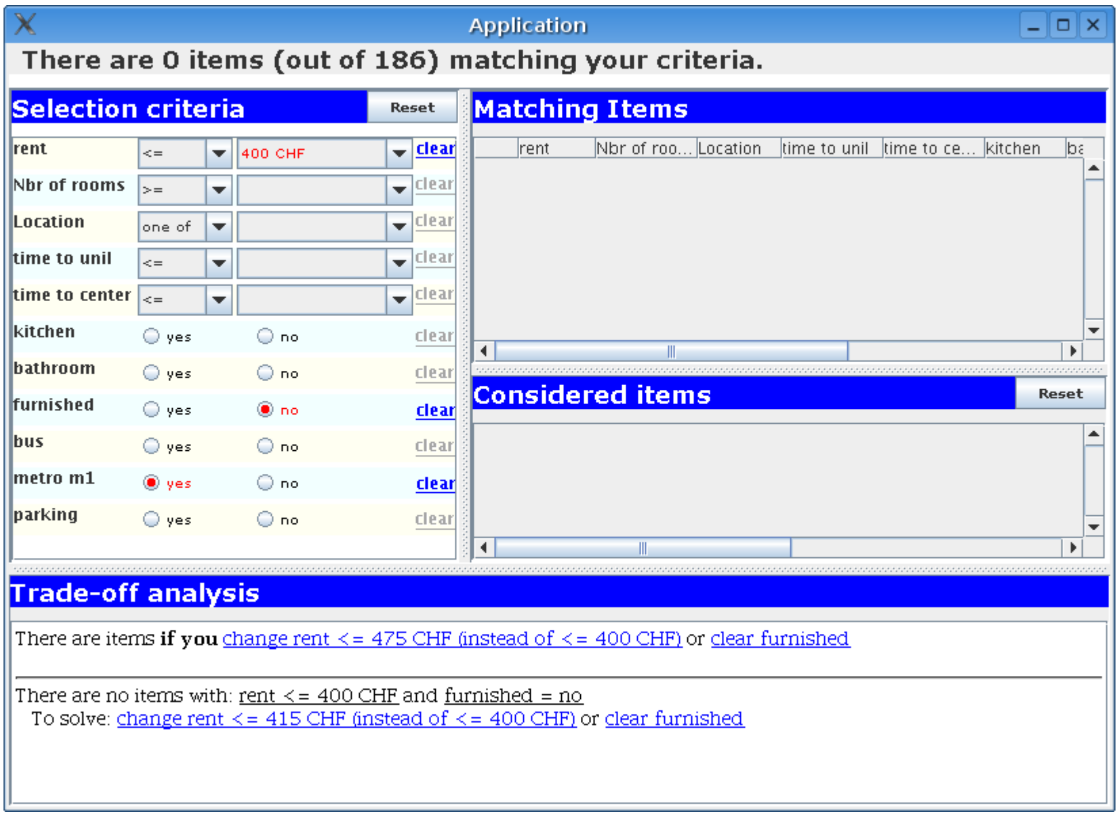
3.7 An example implementation of CBTE

An implementation of the *confidence building tradeoff* (CBTE) system feedback is shown in Figure 3.10. It has exactly the same interface as the baseline system (see Section 2.7.2) while adding a window at the bottom called *Trade-off analysis*. When the query is successful, this window does not show any information. When the query is over-constrained, it shows (1) the global relaxations (GRS), (2) the minimal conflict set (MCS) and (3) the local relaxation set (LRS) for each conflict in the minimal conflict set.

Global Relaxation Set
The global relaxation set is a set of global relaxations, and a global relaxation consists of a set of constraints to be relaxed. We display the global relaxations ordered by their size, i.e. global relaxations that involve only one constraint are presented before the ones involving two constraints, and so on. We tuned the system to show only six relaxations maximum and to exclude the ones involving more than three constraints because of usability issues, i.e. it becomes more difficult to read with the number of constraints involved. Note that effect of this pruning is studied in Section 4.5.7. By reading the displayed global relaxations, users are given a set of options for solving all the conflicting objectives in a query at once, and thus it represents a clear tradeoff question. Users have then two possibilities to go for one of the given options: (1) manually changing the query using the *selection criteria* window or (2) accepting one of the relaxations by clicking on it, with the effect that the system will automatically revise the query in the *selection criteria* window accordingly.

Minimal Conflict Set and Local Relaxation Set
The minimal conflict set is a set of conflicts, and a conflict is a set of constraints that no item in the catalog satisfies. We tuned the system to again show only five conflicts

maximum for usability issues. By reading the conflicts, users learn which parts of the query are causing the conflicts. They can then use the selection criteria window to relax one or several constraints. For each conflict, its local relaxation set is displayed. The local relaxation set is a set of local relaxations that solves the given conflict, and a local relaxation consists of a single constraint to be relaxed. By reading the local relaxation set, users learn how much they need to relax a particular constraint in order to solve a given conflict. As in global relaxations, users can manually change the query using the selection criteria window, or click one of the relaxations for automatic application.



There are 0 items (out of 186) matching your criteria.

rent	<=	400 CHF	clear
Nbr of rooms	>=		clear
Location	one of		clear
time to unil	<=		clear
time to center	<=		clear
kitchen	yes	no	clear
bathroom	yes	no	clear
furnished	yes	[✓ no]	clear
bus	yes	no	clear
metro m1	[✓ yes]	no	clear
parking	yes	no	clear

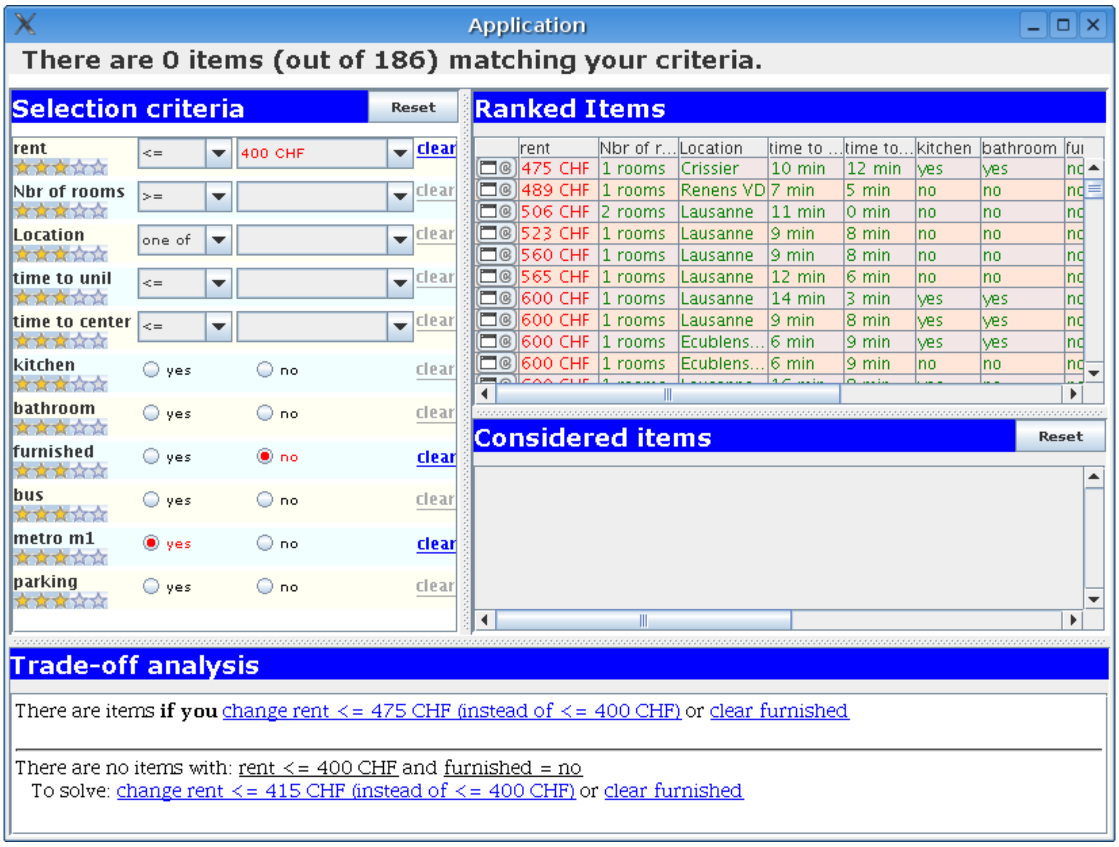
rent	Nbr of roo...	Location	time to unil	time to ce...	kitchen	ba

There are items **if you** change rent <= 475 CHF (instead of <= 400 CHF) or clear furnished

There are no items with: rent <= 400 CHF and furnished = no
To solve: change rent <= 415 CHF (instead of <= 400 CHF) or clear furnished

Figure 3.10: A screenshot of ECatalog, using the *confidence building tradeoff explanations* (CBTE) type of system feedback.

An implementation of CBTE+PWADD, i.e. the combination of both approaches, is shown in Figure 3.11. It has exactly the same interface as the baseline system using *ranked list* feedback described in Section 2.7.2 while adding the *Trade-off analysis* window at the bottom. These two interfaces will be used in a comparative user study in the next Chapter.



Application

There are 0 items (out of 186) matching your criteria.

Selection criteria Reset

rent	<=	400 CHF	clear
Nbr of rooms	>=		clear
Location	one of		clear
time to unil	<=		clear
time to center	<=		clear
kitchen	yes	no	clear
bathroom	yes	no	clear
furnished	yes	[✓ no]	clear
bus	yes	no	clear
metro m1	[✓ yes]	no	clear
parking	yes	no	clear

Ranked Items

rent	Nbr of r...	Location	time to ...	time to...	kitchen	bathroom	fu
475 CHF	1 rooms	Crissier	10 min	12 min	yes	yes	nc
489 CHF	1 rooms	Renens VD	7 min	5 min	no	no	nc
506 CHF	2 rooms	Lausanne	11 min	0 min	no	no	nc
523 CHF	1 rooms	Lausanne	9 min	8 min	no	no	nc
560 CHF	1 rooms	Lausanne	9 min	8 min	no	no	nc
565 CHF	1 rooms	Lausanne	12 min	6 min	no	no	nc
600 CHF	1 rooms	Lausanne	14 min	3 min	yes	yes	nc
600 CHF	1 rooms	Lausanne	9 min	8 min	yes	yes	nc
600 CHF	1 rooms	Ecublens...	6 min	9 min	yes	yes	nc
600 CHF	1 rooms	Ecublens...	6 min	9 min	no	no	nc

Considered items Reset

Trade-off analysis

There are items **if you** change rent <= 475 CHF (instead of <= 400 CHF) or clear furnished

There are no items with: rent <= 400 CHF and furnished = no
To solve: change rent <= 415 CHF (instead of <= 400 CHF) or clear furnished

Figure 3.11: A screenshot of ECatalog, using *ranked list* and *CBTE* type of system feedback.

3.8 Related work

The research on decision confidence for DSSECs is quite recent (with some preliminary experiments in 2000 [35]) and, with the notable exception of Pearl Pu et al since 2004 [53, 56, 63], has mainly focused on carrying out experiments in order to understand the antecedents and benefits of decision confidence, rather than understanding the direct effect of different types of system feedback on decision confidence.

Antecedents of decision confidence
In [89] the authors carry out a laboratory experiment to understand the effect of search effort and domain knowledge on decision accuracy, decision confidence, perceived ease of use and perceived usefulness. For this purpose, participants of the experiment are exposed to four DSSECs. As the result of their study, the authors provided a *partial least square* (PLS) model showing that the effects of search effort and domain knowledge are mediated through decision accuracy and decision confidence to positively impact perceived ease of use and perceived usefulness. Particularly interesting for our work, the authors found out that decision accuracy positively affects the decision confidence, and decision confidence positively affects perceived usefulness. However, they did not provide information about which type of DSSECs are better in terms of decision accuracy or decision confidence, i.e., the use of the four DSSECs was only meant to provide sufficient diversity to construct a PLS model.

Decision confidence on Collaborative Filtering
Collaborative filtering systems (also called recommender systems) attempt to predict items that a user may be interested in based on the similarity of her *profile* with the profiles of many other users. Collaborative filtering differs from DSSECs (content-based systems) in that they don't necessarily have a description of the products in terms of attributes, and make recommendations such as "people that bought this product also bought product X".

One main problem of collaborative filtering is that they are black boxes, providing no transparency of how they work. In this perspective, there is quite a lot of work on understanding which types of explanations about how recommendations are constructed (such as rating histograms [81]) are more effective in order to improve decision confidence. Note that in this thesis we focus on content-based systems only. For a survey on collaborative methods, see [59].

Decision confidence on DSSECs
The authors in [35] found out that allowing users to do a first filtering of a large catalog improves decision confidence. They also found out that adding a comparison matrix (a user interface that allows users to easily compare the set of items that are being considered seriously for making the decision; see Section 2.7) in a DSSEC positively affects decision accuracy, while it does not have an impact on decision confidence.

In [63], the authors compare two types of DSSECs, *user-motivated critiquing* and *system-proposed critiquing*, in terms of decision accuracy, effort and confidence. They show that *user-motivated critiquing* achieved better decision confidence than *system-proposed critiquing*. Notice however that this does not necessarily imply that the first approach was specifically designed to improve decision confidence as the observed increase might be the sole consequence of the increase in decision accuracy that has been observed as well. In fact, while the authors explain the difference between the two systems, they do not explain if their system was designed in order to improve confidence, nor they do provide an explanation of why their system outperformed in terms of confidence.

Instrumented tasks

The performance of several DSSECs when users face a concrete tradeoff has been studied in [53]. In this work, a user evaluation was carried out where the participants were asked to find their best choice in a catalog of fifty apartments. Then, they were asked to find another apartment in the same catalog, 100 CHF[14] cheaper than the one previously found. Thus, the study was artificially creating concrete tradeoffs to the participants. Using this experimental set-up, the authors compared two DSSECS, a baseline system[15] (filtering by constraints, plus sorting by one attribute) and their *TweakingUI* system. With the *TweakingUI* system, participants were allowed to indicate how much (if at all) the system could compromise for each of the criterions. This approach is quite similar to the PWADD system described in Section 2.4.2, with the difference that the importance of criterions were elicited in form of level of compromise (the more compromisable, the lower the weight of importance) and users could ask to filter out from the ranked list the items in the catalog that do not fulfill some of the criteria by defining them as non-compromisable. They found out that while the *TweakingUI* system (preference-based, similar to PWADD) performed better in terms of decision accuracy, the baseline system had better decision confidence. In fact, a recurrent comment was that the TweakingUI system was "hiding something from me", and that they could "see everything in the other system". Thus, it provides positive evidence that constraint-based systems improve decision confidence as we suggest with our DECOFE framework. However in the user study, the participants responded positively to the question of whether they would prefer the *TweakingUI* interface if the catalog would be larger, and thus the authors concluded that preference-based system are better. While we agree that preference-based systems are better in terms of decision accuracy, in our DECOFE framework we argue that constraint-based systems outperform them in terms of decision confidence and we further study this empirically in the next Chapter.

In a similar study [56], the authors also measured decision confidence. First, they asked participants to find their best choice in a catalog of 100 apartments and asked them how confident they were with their choice. Then they were asked to perform four additional tradeoff tasks: (1) "find a cheaper apartment", (2) "find a bigger apartment", (3) "find an apartment which is 100 CHF cheaper" and (4) "find an apartment which is 5 square

[14]100 CHF (Swiss francs) is approximately 63 Euros, on 15th October 2006. .

[15]The authors call *RankedList* interface at the *Baseline* system we presented in Section 2.7.

meters bigger". At the end of this process, the users were allowed to potentially change their first choice, and were asked again how confident they were with their new choice. The experimental results showed that decision confidence increased from 68.6% to 77.1%, the difference being statistically significant. The authors then concluded that example-critiquing with tradeoff support leads to a significant increase in decision confidence.

Trust
In [60] the authors conducted a survey and found out that system-proposed critiquing feedback would perform better in terms of users' trust formation compared to a simple k-best interface even after the "why" enhancement. They also found out that the recommender agent's competence is positively correlated with user's intention to return, but not necessarily with their intention to purchase. This study was then validated in [64] through a user study involving a catalog of 25 notebooks. Trust is defined as the long term relationship between a user and the organization that the recommender system represents, and thus this user study does not exactly deal with improving decision confidence, which is the certainty of the decision maker that she has made the best decision.

Explanations in case-based reasoning
The use of system feedback or *explanations* to improve user confidence was already proposed in the 80's for *case-based reasoning* (CBR). In CBR, the system typically applies abductive or deductive reasoning based on rules defined in the system to provide an explanation for a given solution (the best choice). However, if we knew the solution, we would not need the system anymore. Instead, DSSECs provide explanations (or *system feedback* as we defined in Section 3.4) to users in order to help them find the solution. Several types of system feedback have been studied for DSSECs: *ranking*, *diversity*, *suggestions*, *system-proposed critiquing*, *conflict* and *corrective* feedback.

SmartClients
In [34, 42, 43] the authors present SmartClients, a system combining starfield displays [12] (which shows two-dimensional tradeoffs; See Section 2.4.7), parallel coordinates (which show a subset of the items in a graphical form) and PWADD rankings [67] (See Section 2.4.2). In [42] the authors dismiss the use of minimal conflict sets because of their potential exponential size growth and propose solving over-constrained queries by allowing the user to chronologically retract her constraints. In [34, 43] the authors also provide corrective feedback and discuss its benefit in terms of decision accuracy, but not in terms of decision confidence. Corrective feedback was already proposed in [28, 30], and we discuss it in detail in Section 3.3.3.

Efficient computation
Showing conflicts and relaxations has also been proposed for *interactive problem solving* (such as *complex product configuration*) or *diagnosis* [23] tasks using the *constraint satisfaction problem* (CSP) framework, as in the case of the ILOG configurator [38]. In the cases where computing one minimal conflict is enough (i.e. not requiring to com-

pute the minimal conflict set), the QuickXPlain algorithm [25] improves the efficiency of finding a preferred minimal conflict compared to existing non-intrusive algorithms by recursively partitioning the conflict detection problem into sub-problems and by skipping sub-problems that are not part of the conflict. In the cases where the complete minimal conflict set and the global relaxation set are needed, the authors in [26] proposed first compiling the configuration problem in an automaton representing the set of solutions of the CSP, and then usin this data structure to provide corrective feedback efficiently. However, research in this direction once again does not deal with improving decision confidence.

3.9 Summary

The research on decision confidence for DSSECs is quite recent (with some preliminary experiments in 2000 [35]) and, with the notable exception of Pearl Pu et al since 2004 [53, 56, 63], has mainly focused on carrying out experiments in order to understand the antecedents and benefits of decision confidence, rather than understanding the direct effect of different types of *system feedback* on decision confidence.

We have indentified that current research is mainly focused on novel types of system feedback based on preferences rather than constraints. In the DECOFE framework, we argue that system feedback based on constraints, even if not novel as such, can be combined in order to improve decision confidence.

In this direction, we have proposed a framework called DECOFE to study the building of decision confidence in the *query-feedback search* interaction model. Within this framework, we extended the definition of the *query-feedback search* interaction model as an iterative process in which a user makes queries to the system in order to develop her *domain knowledge*, *preferences* and *confidence* for *tentative decisions* through the *system feedback*, until she is *convinced* of her final decision. We identified four properties for the design of system feedback that can potentially contribute to decision confidence building: (1) *constraint-based*, (2) *universally valid*, (3) *negatory* and (4) *unique in its class.*

Using the DECOFE framework, we have studied the potential contribution to decision confidence building of several types of system feedback proposed in the literature. Within this framework, we have argued that *conflict* and *corrective* feedback (combined with either *matching list* or *ranked list* feedback) were the most promising types of feedback for decision confidence building. As it will be shown in the next Chapter, this claim is further validated by experimental evaluation in the case of ranked list feedback compared to conflict and corrective feedback.

Based on this analysis, we proposed the *confidence building tradeoff explanation* (CBTEs) feedback which explains the query tradeoffs in terms of conflicts and relaxations that can potentially contribute to decision confidence building, and it is meant to be combined with either *matching list* or *ranked list* system feedback. In typical situations where there is a mismatch between what the decision maker wants and what is it in the catalog, CBTE provides a clear trade-off question to help the user make a good decision through the *global relaxation set*, and provides evidence to build decision confidence through the *minimal conflict set* and the *local relaxation set*. While global relaxations and the minimal conflict set had already been proposed for interactive problem solving in the literature, we explain their contribution to decision confidence building based on the DECOFE framework and we propose local relaxations to complement the previous two types of feedback.

In addition, we argued that CBTE provides tradeoff information comparable to the one produced by PWADD and therefore we also hypothesize that both approaches will have a similar decision accuracy. However, the advantage of CBTE is a potential increase in decision confidence.

The use of minimal conflict sets has been dismissed by some researchers [42] because of their potential exponential size growth. In order to study in practice the size of such type of feedback, we ran two simulations: simulations using queries uniformly distributed in size which produced fairly bad results, while simulations using queries built in an incremental fashion had very attractive results. This study shows that the results of simulations depend very much on how queries are constructed, and it just confirms that results on simulation of human behavior are in general quite doubtful. However, as we said previously, a correct evaluation requires a user study.

The various claims made within the DECOFE framework and the usefulness of CBTE will be experimentally evaluated in the next chapter.

Chapter 4

Comparative user study

4.1 Introduction

In Chapter 3, we introduced the *confidence building tradeoff explanations* (CBTE) type of system feedback aiming at improving decision confidence in *decision support systems for electronic catalogs* (DSSEC). As explained in the previous Chapter, the main idea of the *decision confidence in query-feedback search* (DECOFE) framework is to design a DSSEC that gives to the users complete query control during the decision process, while helping them to build decision confidence. This is achieved by means of *system feedback* that fulfills four properties given in the DECOFE framework: (1) *constraint-based*, (2) *universally valid*, (3) *negatory* and (4) *unique in its class.* While we are mainly interested in improving decision confidence, a system which neglects the other important DSSECs features such as decision accuracy is not of course admissible.

The performance of DSSECs can be measured by decision accuracy, decision confidence, search effort and trust, among other constructs. In the following section, we introduce a behavioral model to evaluate different DSSECs in terms of the objective and subjective measures introduced in Section 2.6. An ideal objective would be to compare several DSSECs using this research model. As time and resources are limited, we will compare one of the most used types of system feedback in DSSECs, the *parametric weighted additive* (PWADD) type of *ranked list* feedback, with *confidence building tradeoff explanations* (CBTE) feedback introduced in the previous chapter. The DSSEC described in Section 2.7 using *matching list* feedback is used as a baseline.

We set up a between-group experimental design to evaluate the performance improvement (with respect to the baseline system) of CBTE, PWADD and of a combination of CBTE and PWADD (CBTE+PWADD). In Section 4.2, we present the research model. In Section 4.3 and 4.4, we describe the methodology and the experimental results respectively. While results need to be taken with caution as described in Section 4.6, the experimental study shows that CBTE outperforms PWADD in all the subjective constructs presented in the research model, and that decision accuracy is not degraded. In Section 4.5, we interpret these experimental results. We present limitations of the study in Section 4.6, related work in Section 4.7 and conclude in Section 4.8.

4.2 Research model

In Section 2.5 we have seen three subjective constructs used in this study that positively affect trust intentions as reported in [89, 90, 63, 85, 86]: *perceived ease of use* of the e-vendor's system, *decision confidence* (user confidence in their choices) and user *trust* in the e-vendor. We have also seen that *perceived usefulness* and *ease of use*, *enjoyment* and *perceived competence* positively affects *trust* [86, 88]. In Section 2.8 we have seen that most research in DSSECs has been concerned with improving *decision accuracy*, which is an objective measure. Recently it has been found that [89] *decision accuracy* also has the positive impact of improving *decision confidence. Domain knowledge* has also been found a determinant of *decision accuracy.* The authors in [80, 63] found out that *perceived control* positively affects *decision confidence.* In this thesis we are interested in designing DSSECs that improve decision confidence, without sensibly penalizing these other important constructs. In Figure 4.1 we show the proposed research model.

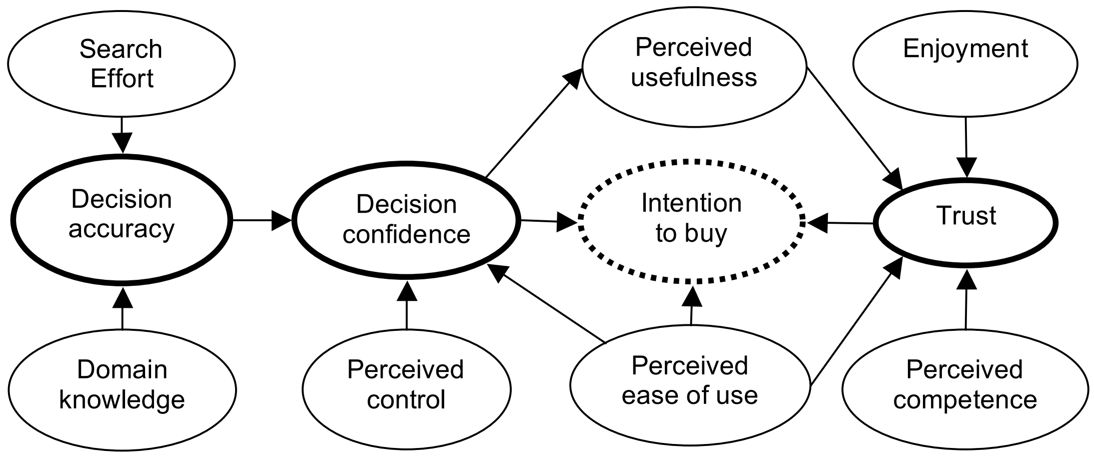


Figure 4.1: User Study. Research model.

In this user study, we compare the *baseline* system described in Section 2.7 with two types of (expected) improvements: using the *parametric weighted additive* (PWADD) decision strategy, which is used in most DSSECs, and (2) *confidence building tradeoff explanations* (CBTE), which is our approach in this thesis. These systems are evaluated using the constructs cited in the given research model. Given that they are subjective measures (except for decision accuracy), we are going to use questionnaires that have shown been to have strong content validity (again, see Section 2.5).

Also, in Section 3.6 we saw that the size of CBTE could grow exponentially with the size of the query, making this type of system feedback useless in such cases. We ran two simulations, one using queries uniformly distributed in size which had quite bad results, and another one using queries build in an incremental fashion which had very attractive results (small CBTE sizes). The study showed that the results of simulations depend very much on how queries were constructed, and it just confirmed that results on simulation of human behavior are in general quite doubtful. In this sense, we are also evaluating experimentally the size of CBTE feedback in this user study to understand whether the potential exponential size growth is in fact a drawback or not in practical business-to-consumer situations.

We also discussed in Section 3.4 that CBTE and the parametric weighted additive (PWADD) both provide similar information about the conflicting objectives in two different forms. PWADD, in one hand, provides this information in terms of a ranked list, and on the other hand, CBTE lists queries conflicts and query relaxations. Therefore we hypothesized that both approaches will achieve similar decision accuracy, which will be also verified in this user study.

We next explain the methodology for this comparative user study, followed by results and conclusions.

4.3 Methodology

In order to test the effects of applying the *explicit trade-off analysis* (CBTE) and/or the *parametric weighted additive* (PWADD) approaches, a common user interface was created for the four possible systems (shown in Table 4.1) resulting from the two given independent variables. 39 persons participated in the user study and were randomly assigned to three groups. Each participant was introduced to the baseline system and was then asked to search their best option in a catalog of used cars. Afterwards, the participant was asked to search again for a better option in the same catalog of cars using one of the other three systems. The next step consisted of a validation phase, where the participant was given a flat list of all the cars and asked to carefully inspect all of them to find their correct best choice. We then evaluated the increase of performance of using CBTE after the baseline system compared to the use of PWADD after the baseline system. The increase of performance for the combination of CBTE+PWADD is also evaluated. Performance is based on objective measures, such as decision accuracy, as well as subjective measures extracted from questionnaires.

Subjects
It is foreseen there will be with-in group variations as participants will not have (1) a common expertise in the use of electronic catalogs, (2) a common expertise of the

	-CBTE	**+CBTE**
-PWADD	Baseline	CBTE
+PWADD	PWADD	CBTE + PWADD

Table 4.1: Two indepedent variables user study.

catalog domain being used and (3) common criteria flexibility. An experimental design to cancel these effects is the *within-subjects* (or "repeated measures") design, where each participant is exposed to the four systems. This brings about the problem of the necessity of counterbalancing the *carry-over* effects of practice and fatigue: given a participant and one of the interfaces, the results would be influenced by the interfaces that she used previously. In other words, the results of a participant where she uses the four interfaces in some order would not be equal to the same participant using the four interfaces in a different order. To counterbalance this effect, it is necessary to create a group of participants for each possible permutation in ordering the four interfaces. This makes 4! = 24 groups. If we say that the minimum of participants per group should be 10, then this study would need at least 240 participants. Since it is hard to recruit participants we needed to adopt another experimental design.

For this study, we managed to involve only 40 participants, which it is much less than the 240 participants needed in the previous set-up. In order to cancel the 3 effects previously mentioned, we decided to design a *between-groups* comparative design, i.e. the baseline system is taken as a common denominator and participants are asked to evaluate a system compared to the baseline. This set-up changes absolute questions to relative ones and thus potentially cancels the mentioned variance between participants. For instance, instead of being asked "Is it easy to use?" they are asked "Is it easier to use than the baseline system?". The price to pay for this change is the necessity of converting a two independent variable setup to a one independent variable setup, which limits the type of statistical tools that can be applied afterwards to analyze the results.

Concretely, 39 participants were involved in the user study. They were promised an award of 20 CHF at the end of the experiment. They were randomly assigned to three groups of thirteen participants each. Each participant first used the baseline system and thenone of the other three systems according to their group. This organization is shown in Table 4.2.

		First system	**Second system**		
Group	**# participants**	**Baseline**	**CBTE**	**PWADD**	**CBTE + PWADD**
G1	13	Yes	Yes		
G2	13	Yes		Yes	
G3	13	Yes			Yes

Table 4.2: One independent variable user study.

In the demographics questionnaire, participants were asked about their gender, age, occupation, familiarity with computers and decision making. They were asked also about their expertise in the catalog domain but this question could not be used in the analysis of the results because it turned out to be incorrectly formulated. The list of questions is available in Table A.1 at page 121. Table A.5 at page 141 shows the results of the demographic questionnaires. According to an ANOVA statistical test, no demographic characteristics favored one system over the others.

Stimuli and task
In order to create a more realistic scenario, we contacted a company that aggregates the inventory of used cars from several concessionaries in Switzerland. They provided us with a catalog of 61443 cars, from which 7924 are described with the following 35 attributes[1]: brand, model, model type, current price, original price, date of the 1^{st} matriculation, mileage, category, fuel type, transmission, number of seas, number of doors, performance in hp, cubic capacity, number of cylinders, color and the boolean features metallic color, ABS, airbag, air conditioning, alarm, cd player, central locking, electrical seats, electrical windows, fog lamps, heating seats, leather interior, navigation system, power steering, speed control, sport chassis, sun roof, theft prevention, traction control and xenon lights.

However the procedure used to evaluate decision accuracy, described below, needs to limit the number of items, and based on similar studies [56, 58], 200 items seemed a reasonable choice, which were randomly selected from the 7924 available. We called this catalog *Comparis200*[2]. To introduce the systems to the participants, we used a catalog of 186 apartments[3] modeled by 11 attributes.

The objective of the participants was to find their most preferred used car from the ones available in the catalog. Rather than designing a controlled laboratory experiment with concrete objectives given to the participants such as "finding a cheap car with few mileage and a minimum of three features such as airbag", this open task allowed the systems to be tested in a more ecological setting, where typically participants do not have a fixed set of preferences.

Procedure
The timeline for each participant, shown in Table 4.4, was as follows[4]: First there was a typical demographics questionnaire about the participant. Afterwards, the participant used the baseline system followed by the second system according to her group. A validation phase to measure the decision accuracy followed and finally there was a

[1]We called this catalog *Comparis8000.* We would like to acknowledge Comparis.ch for sharing this catalog to the research community. It is available to download: See Appendix 2.

[2] *Comparis200* is a catalog of 200 used-cars, taken from *Comparis8000*, modeled by 35 attributes. It is available to download: See Appendix 2.

[3]We would like to acknowledge Paolo Viappiani for sharing a database of apartments

[4]A Dell Optiplex SX260 (Pentium IV, 2'8 GHz, 512Mb RAM) with a 19" color screen was used in this user study.

questionnaire specific to each group.

The objective of these four systems is the same: to assist a user in finding her preferred item from the set of available items. A common user interface for the four systems is needed in order to achieve a fair evaluation between them. However, as they differ in the elicitation process and the type of system feedback provided to the user, the user interface cannot be the exactly same. The baseline system described in Section 2.7 has been implemented and extended to integrate the support for the intrinsic features of the two improved approaches, as described in Sections 2.7.2 and 3.7. This ensures that they are both practically identical except for the one aspect whose effect is being tested. In Table 4.3 we show the complete inventory of the differences between the four different systems.

System	**Description & Screenshot**	**Differences**
Baseline	Section 2.7.2 See Figure 2.10 at page 50	1) The *Items* window shows all the items matching the query, the query being interpreted as a set of parametric constraints.
CBTE	Section 3.7 See Figure 3.10 at page 77	1) idem. 2) There is a new window, called *Trade-off analysis*, which shows CBTE feedback according to the query, the query being interpreted as a set of parametric constraints.
PWADD	Section 2.7.2 See Figure 2.11 at page 51	3) The *Items* window show all the items sorted according to the query, the query being interpreted as a set of weighted parametric utility functions (PWADD feedback). 4) The *Selection criteria* window allows the user to also specify a weight for each parametric utility function.
CBTE+ PWADD	Section 3.7 See Figure 3.11 at page 78	2) idem. 3) idem. 4) idem.

Table 4.3: Inventory of the differences between the different systems.

Participants were asked to use the baseline system. It is a common practice to provide training to teach participants to use a user interface in order to further help in canceling the variations in experience between participants [92]. In this study, the training was provided in the form of a tutorial for each one of the interfaces. Concretely, there was a printed tutorial that the participants needed to follow while using the real system with a catalog of apartments. The tutorial is available in Appendix 1, under the label "User study. Baseline tutorial" at page 127. In Table 4.5 there is the number of pages of the tutorial for each system and the average time that the participants took to follow it. After the training, we asked the participant to look at the printed description of the catalog of second-hand cars. It is available at Appendix 1, under the label "User Study. Second-hand cars database". Afterwards, the participant was asked to use the system to find her preferred second-hand car. They were asked to limit themselves to five minutes although it was not a strict constraint. The system monitored the total time of the interaction, the number of criteria revisions, the number of distinct attributes used and the number of considered items among others. Finally, the system presented a questionnaire with measures related to the constructs confidence, ease of use, usefulness, enjoyment, trust which is listed at Table A.2 in page 123. These measures have been adapted from the questionnaires reported in Section 2.6 which were shown to have strong content validity. The questionnaire also included the NASA TLX test which measures cognitive load in performing a task [73].

The next step in the user study was to use one of the other three systems, which we hereafter call the *second system*. As before, the participant was introduced to the system with the help of a printed tutorial that explained the additional features. The tutorial is available in Appendix 1, under the label "User study. ECatalog+T" on page 136 for group 1, "User study. ECatalog+W" on page 134 for group 2 and the combination of those two used for group 3. Afterwards she was asked to find her preferred second-hand car using the second interface. The catalog used was the same as in the previous step. Thus, the task, finding the preferred item from the catalog, is not exactly the same as in the previous step because the participants already had some knowledge of the set of items available in the catalog from the previous step. However, remember that this comparative study does not evaluate whether one of the second systems is better than the baseline one, but rather which of the three systems is greater improvement on the baseline system, and thus it is not an issue if the task in the second system is not the same as the first one. The system monitored the same measures as in the previous step plus specific measures intrinsic to the second system. Finally, the system again presented a questionnaire with questions related to the same constructs as in the previous step. However, in this questionnaire, there were measures of the second system in comparison to the baseline one, i.e. instead of asking whether the system was useful, it was asking whether the second system was more useful than the baseline system. The list of questions is available at Table A.3 on page 124. The NASA TLX test was also included in this step.

After using the two systems, we asked the participants to validate their choice. The procedure for this validation phase was taken from [56, 35] and in this user study was implemented by simply taking the same interface used in the baseline system and removing the "Selection Criteria" window. Concretely, the system presented (1) a flat list of all the items and (2) a space where the user could temporally put some items for further inspection, called the "Considered Items" window. The two items that the user selected in the baseline and second system were added to the "Considered Items" window automatically by the system. The user could sort the flat list of items by one column at a time. Using this interface, the participant was asked to carefully inspect all the items and to switch to a better one if she could find one or to stay with one of the previous two choices. The decision accuracy of one group of participants was measured as the fraction of the number of participants that using the second system selected the same item as in the validation phase, divided by the total number of participants of the group.

To finish the user study, the participants were asked to fill up a questionnaire specific to their group in order to better interpret the results later.

For each participant, the same experimenter introduced the experiment to the participant, provided her with the printed tutorials and started the software. From this point on, the user study was autonomous in order to avoid any interference from the experimenter to the participant. Questionnaires and the interactions using the systems were all recorded in log files for later analysis. At the end of the study, the participant was awarded with 20 CHF independently of the performance.

Timeline for a participant:

- Demographics survey
- Baseline System
    - Tutorial
    - Perform task
    - Survey
- Second system
    - Tutorial
    - Perform task
    - Survey
- Validation
    - Switching rate
    - Survey
- Specific survey

Table 4.4: Timeline for a participant in the user study.

System	# pages	Average time
Baseline	5	12.0 min
CBTE	5	7.3 min
PWADD	2	9.2 min
CBTE + PWADD	8	13.3 min

Table 4.5: Number of pages of the tutorial of each system and the average time that the participants took to complete it.

Notice that the selected experimental setup guarantees that the only noticeable differences between the compared systems are the ones indicated in Table 4.3. Therefore, these differences should be the ones used for the result interpretations that are given in Section 4.5.

4.4 Results

Each participant took on average 1h17min to take part in the user study. During the study, we collected objective measures (total interaction time, number of criteria revisions, number of distinct attributes used and number of considered items among others) and subjective measures related to the constructs of the research model discussed in Section 4.2. These results are available in Table A.6 – Table A.8 on page 142[5]. We first apply factor analysis on the subjective measures to identify the relevant ones and group them into constructs in section 4.4.1 and we then analyzed the results using statistical significance tests in section 4.4.2.

4.4.1 Factor analysis

As explained previously, each participant answered 34 questions after using the second system and four questions after the validation. All items were measured on a seven-point Likert strongly disagree/strongly agree scale. We applied statistical tools over these questions to find out the most relevant factors. Concretely, we applied principal component analysis (PCA) to these questions, which suggested eight factors according to the latent root criterion. Then we computed the table of iterated varimax rotated principal factor analysis, plus the minimum, maximum, mean and standard deviation for each measure which is shown in Table 4.6. A measure is associated to a factor if the measure has a high loading with this factor and low loadings with all the other factors. Then this factor is considered to exhibit convergent and discriminant validity. We used 0.50 as a threshold between a high and a low loading. Questions Q1, Q6, Q7, Q9, Q10, Q12, and Q13 were removed due to a high loading in more than one factor or in none of them.

Table 4.7 shows these factors and their associated measures. In this table we can easily identify that factor 1 is related to perceived ease of use and perceived usefulness. Factor F2 is mainly associated with decision confidence after validation, that is, how confidence a participant was that she had found the best or at least a very good choice using the second interface. Q15 about perceived usefulness was also associated to factor F2 but it was eliminated because it didn't belong to the same construct and had a much lower loading. Factor F3 is related to decision confidence before validation, that is, how confidence a participant was that she had found her preferred choice after using the second system. Factor F4 is clearly related to the trust construct. Factor F5 cannot

[5]The complete interaction log files are also available to download. See Appendix 2.

	F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**min**	**max**	**mean**	**Std**
~~**Q1**~~	-0.26	-0.17	0.36	0.00	-0.38	0.15	-0.20	0.41	3	7	5.10	1.32
Q2	-0.44	-0.16	**0.69**	-0.15	-0.04	0.24	-0.13	0.34	2	7	5.28	1.54
Q3	-0.41	-0.14	**0.63**	-0.03	-0.06	0.21	-0.11	0.45	2	7	5.33	1.38
Q4	-0.24	-0.13	0.29	-0.26	-0.14	**0.72**	-0.03	0.01	1	7	4.69	2.08
Q5	0.19	0.22	-0.18	0.04	0.37	-0.16	**0.54**	0.02	1	6	2.59	1.63
~~**Q6**~~	0.28	0.09	**-0.58**	0.20	0.24	-0.04	**0.54**	-0.10	1	6	2.18	1.34
~~**Q7**~~	0.21	0.14	-0.08	0.09	0.46	-0.10	0.29	-0.40	1	6	2.82	1.78
Q8	-0.02	0.09	0.09	-0.20	0.00	-0.01	-0.03	**0.67**	2	7	4.51	1.69
~~**Q9**~~	0.10	0.16	-0.35	-0.08	0.40	0.00	-0.11	-0.45	1	6	2.23	1.21
~~**Q10**~~	**-0.52**	-0.24	0.09	-0.16	-0.26	0.05	-0.08	**0.50**	2	7	5.00	1.48
Q11	0.12	0.11	-0.03	-0.08	-0.03	-0.05	**0.64**	-0.06	1	7	2.74	1.78
~~**Q12**~~	-0.24	-0.09	0.12	0.02	-0.11	0.49	-0.25	0.47	2	7	4.05	1.50
~~**Q13**~~	0.43	0.03	-0.06	-0.10	0.33	0.08	0.45	-0.15	1	6	2.90	1.41
Q14	-0.27	-0.18	-0.21	-0.12	**-0.60**	0.23	-0.20	0.18	1	7	4.21	1.56
Q15	-0.39	**-0.55**	0.10	-0.47	-0.38	-0.06	-0.03	0.12	2	7	4.87	1.73
Q16	-0.35	-0.18	0.13	-0.16	**-0.61**	0.04	-0.16	0.05	2	7	4.54	1.53
Q17	**-0.55**	-0.11	0.26	-0.03	-0.41	0.40	-0.25	0.03	2	7	4.87	1.54
Q18	**-0.73**	-0.11	0.12	-0.06	-0.25	0.25	-0.13	-0.01	2	7	5.05	1.45
Q19	**-0.77**	-0.10	0.01	-0.15	-0.09	0.11	-0.14	0.23	2	7	4.90	1.41
Q20	**-0.78**	-0.12	0.29	-0.26	-0.17	0.13	-0.11	0.31	2	7	5.38	1.44
Q21	**-0.67**	-0.22	0.28	-0.23	-0.33	0.12	-0.25	0.09	2	7	5.00	1.45
Q22	**-0.51**	-0.24	0.24	-0.28	-0.35	0.31	-0.07	-0.03	2	7	5.10	1.60
Q23	-0.45	-0.04	-0.03	-0.24	-0.07	**0.76**	-0.10	0.04	2	7	5.03	1.61
Q24	**-0.67**	-0.21	0.04	-0.16	-0.26	0.33	-0.20	0.04	2	7	5.13	1.38
Q25	**-0.64**	-0.01	-0.15	-0.14	-0.13	0.40	-0.40	-0.01	2	7	4.72	1.36
Q26	**-0.82**	-0.08	0.26	-0.22	-0.17	0.07	-0.19	0.21	2	7	5.44	1.41
Q27	**-0.75**	-0.06	0.20	-0.20	-0.25	0.14	-0.08	0.12	1	7	5.21	1.71
Q28	**-0.80**	-0.11	0.13	-0.18	-0.31	0.21	-0.11	0.11	1	7	5.13	1.60
Q29	**-0.54**	0.02	0.09	-0.15	-0.44	0.39	0.20	0.22	2	7	5.69	1.16
Q30	-0.40	-0.08	0.09	-0.17	**-0.53**	0.49	0.06	0.33	1	7	4.87	1.51
Q31	-0.24	0.07	0.09	**-0.79**	-0.03	0.17	0.11	0.19	2	7	4.62	1.44
Q32	-0.29	-0.13	-0.03	**-0.81**	-0.28	0.26	-0.07	0.11	2	7	4.51	1.32
Q33	-0.46	-0.14	0.14	-0.25	**-0.69**	0.09	-0.09	-0.05	2	7	5.15	1.44
Q34	-0.39	0.08	0.22	**-0.50**	-0.47	0.15	0.12	0.10	2	7	5.54	1.34
Q35	-0.16	**-0.95**	0.07	-0.01	-0.07	0.01	-0.08	-0.02	1	7	5.51	1.71
Q36	-0.05	**-0.93**	-0.02	-0.04	-0.15	-0.01	-0.10	0.12	1	7	5.10	1.71
Q37	-0.18	**-0.78**	0.21	0.07	-0.09	0.30	-0.18	-0.01	2	7	5.51	1.43
Q38	**-0.70**	-0.38	0.21	0.01	-0.19	0.33	-0.02	-0.08	2	7	5.44	1.58

Table 4.6: Results of the user study. Factor analysis applied to the questions presented in Table A.3 and Table A.4 on page 124.

be associated to a construct of the research model because its associated measures are unrelated. Factor F6 is associated to domain knowledge, as defined in Section 2.5. Factor F7 and F8 describe decision confidence, perceived ease of use and perceived usefulness and can be discarded because these constructs are already taken into account by previous (and so more important) factors. Table 4.8 shows the relevant factors, now associated to a construct of the research model and its Cronbach Alpha levels above 0.7 probing construct reliability.

Factor	**Question**	**Description**
F1	Q17	I can more quickly find the information I need on this system
	Q18	The system was easier to use for product assessment
	Q19	The system was easier to use to compare products
	Q20	The system was easier to use to accomplish the task
	Q21	Overall, I found the system easier to use
	Q22	This system provided better quality information
	Q24	This system was more useful for product assessment
	Q25	This system was more useful for comparing products
	Q26	This system was more useful for accomplishing the task
	Q27	This system improved my performance in accomplishing the task
	Q28	This system increased my effectiveness for accomplishing the task
	Q29	I found my interaction with this system more interesting
	Q38	After the validation, I think that the Improved System was more useful than the Baseline System
F2	~~Q15~~	~~The system provided more helpful guidance to interact with it~~
	Q35	After the validation, I am satisfied with the choice that I took using the Improved System
	Q36	After the validation, I am confident that using the Improved System I found the best possible choice from the catalog
	Q37	After the validation, I think that I found a very good choice using the Improved System
F3	Q2	I am more satisfied with my choice
	Q3	I am more confident that I have found my optimal choice from the ones in the catalog
F4	Q31	I would feel that this online vendor is more honest
	Q32	I would feel that this online vendor is more trustworthy
	Q34	I would feel that this online vendor cares more about customers
F5	Q14	My interaction with the system was easier for me to understand
	Q16	This system is more user-friendly
	Q30	I found my interaction with this system more enjoyable
	Q33	I would feel that this online vendor is more competent
F6	Q37	I have found a better choice
	Q23	About the products, I got a better "picture/understanding" of the different possibilities for my region of interest
F7	Q5	I became more confused when I used the system
	Q11	The system behaved more in unexpected ways
F8	Q8	I found it easier to recover from errors encountered while using the system

Table 4.7: Results of the user study. Factors and its associated measures.

Factor	Associated construct	Construct reliability (cronbach alpha)
F1	**Perceived ease of use and usefulness**	0.96
F2	**Confidence after validation**	0.91
F3	**Confidence before validation**	0.93
F4	**Trust**	0.86
F6	**Domain knowledge**	0.80

Table 4.8: Results of the user study. Factors, associated construct s and construct reliability.

From these results, we see that the statistical analysis identified four factors that nicely correspond to the constructs in the research model shown at Section 4.1. It is not surprising that **the perceived ease of use construct and the perceived usefulness construct have been merged together**, as it is shown in the research model that the former interacts directly with the later. Also, in accordance with other studies [41], satisfaction and confidence have proved to be correlated. It is interesting to see that the statistical analysis has **split the decision confidence construct in two constructs, decision confidence before validation and after validation**. In practice, users that buy in an e-commerce site do not take part in a validation phase and so this construct might be argued to be irrelevant. However, we think that this validation phase takes place in a real situation although in a different form. For instance, if a user buys a product and one week later she discovers that the vendor had a better product, she would acknowledge at this point that she did not buy the right product and her decision confidence after this process of validation would decrease. This, in turn, would potentially affect user trust in following interactions. The **trust construct** has also been identified by the statistical analysis.

The *perceived control* construct has not initially been captured by the factor analysis. Perceived control is represented by question Q12 in the Table A.2: "I felt more in control of the system". Although it cannot contribute to the validation of the research model, it can be very well employed in the interpretation of the results and so we include it in the following statistical tests.

4.4.2 Statistical significance tests

The first result to note is that the decision accuracy for the three groups was quite similar (69.2% for G1, 61.5% for G2 and 69.2% for G3) and the difference was not statistical significant ($p > 0.68$). See Table 4.9. This is quite an impressive result which suggests that although CBTE and PWADD provide a different type of assistance to the user, their value is quite similar from the perspective of the user seeking the target item.

Mean			**Chi-square p-value**		
G1	**G2**	**G3**	**G1-G2**	**G1-G3**	**G2-G3**
69.2%	61.5%	69.2%	0.68	0.68	1

Table 4.9: Results of the user study. Decision accuracy for G1 (CBTE), G2 (PWADD) and G3 (CBTE+PWADD).

We also have four objective measures (interaction time, number of criteria revisions, number of distinct attributes used and number of considered items) monitored by the system during the interaction of the participant and the four constructs identified in the previous section coming from the questionnaire answered by the participant. As explained in section 4.3 "User study methodology", we divided 39 participants into three groups. This makes three groups, thirteen participants in each group and eight variables per participant. In order to eliminate out-liers, we applied a median filter by eliminating the lowest and the highest value for each group and variable. We could not use the ANOVA statistical significance test because the data did not exhibit a normal distribution, which is a prerequisite of such a test. We used the Wilcoxon test, which is a more robust non-parametric test [91].

Participants took an interaction time of 4 min 15 seconds using the second system, revised 14.5 times the criteria, used 10.2 distinct attributes and considered 1.7 items in average. In Table 4.10 we see the ratio between the second system and the baseline system for these measures. For instance, we see that the number of criteria revisions while using the second system increased for group 1 (47%) and group 2 (35%) while they decreased for group 3 (68 - 100 = -32%) compared to when the baseline system was used.

Ratio Measure	**Mean**			**Wilcoxon p-value**		
	G1	**G2**	**G3**	**G1-G2**	**G1-G3**	**G2-G3**
Interaction time	93.30%	106.48%	80.14%	0.213	0.003	0.016
Nbr of criteria revisions	147.73%	135.35%	68.38%	0.021	0.003	0.003
Nbr distinct attributes used	202.93%	230.90%	103.60%	0.441	0.005	0.018
Nbr considered items	69.24%	106.82%	115.15%	0.012	0.028	0.500

Table 4.10: Results of the user study. Ratio between the second system and the baseline system for objectives measures.

Table 4.11 shows the mean and significance test for the constructs identified in the previous section. These constructs consist of unweighted averages of their associated measures shown previously at Table 4.8 as done in similar studies [89, 88]. Remember from section 4.3 "User study methodology" that these measures are seven-point Likert scale, where 1 is strongly disagree, 4 is neutral and 7 is strongly agree. For instance, we see there was an improvement of perceived ease of use and usefulness while using the second system compared to the baseline system, and that this improvement was bigger for group 1 (5.40 in the seven-point Likert strongly disagree/strongly agree scale) than for group 2 (4.71). This difference is statistical significant (p = 0.003).

Construct	**Mean**			**Wilcoxon p-value**		
	G1	**G2**	**G3**	**G1-G2**	**G1-G3**	**G2-G3**
+ Perceived ease of use and usefulness	5.40	4.71	5.66	0.003	0.045	0.003
+ Confidence after validation	5.94	5.06	5.67	0.004	0.028	0.006
+ Confidence before validation	5.73	4.95	5.50	0.015	0.116	0.126
+ Trust	5.00	4.33	5.39	0.008	0.047	0.003
+ Domain knowledge	4.73	4.90	5.23	0.554	0.043	0.050
+ Perceived control	4.09	3.45	4.55	0.018	0.091	0.008

Table 4.11: Results of the user study. Constructs about the second system in comparison to the baseline system.

The NASA TLX test which measures cognitive load in performing a task did not shown any statistically significant difference between the three groups.

4.5 Interpretations of the results

4.5.1 Decision confidence

In Table 4.11 we can see that CBTE (group G1) outperforms PWADD (group G2) in terms of perceived ease of use and usefulness (p = 0.003), confidence after validation (p = 0.004), confidence before validation (p = 0.0015), trust (p = 0.008) and perceived control (p = 0.018).

In Section 3.3 we argued that CBTE and PWADD both provide similar tradeoff information. However, in the case of CBTE, users can use this information to also build decision confidence. CBTE achieves this by providing feedback according to the user's current query that fulfills four given properties: (1) *constraint-based*, (2) *universally valid*, (3) *negatory* and (4) *unique in its class.* For instance, consider this system feedback:

Example 4.1 Example CBTE feedback.

Given your query:
price <= 6'500 Euros AND mileage <= 50'000 km AND color = blue AND airbag = yes

There are not items with:
price <= 6'500 Euros AND mileage <= 50'000 km

You have two options for solving this conflict:
1) change price <= 9'238 Euros (instead of 6'500 Euros)
2) change mileage <= 55'700 km (instead of 50'000 km)

It explains concretely which the tradeoff between price and mileage is, given that the user desires a blue car with airbag. If the decision maker decides to take a car based on option 2 and a friend ask him why she took a car with more mileage than agreed (50'000 km), the decision maker can provide a simple justification: there is not such an option unless you agree to pay 9'200 Euros (instead of the 6'500 Euros we foresaw), while 5'700 km more mileage is not such a big deal. Moreover, the user can also easily verify the correctness of this information.

On the other hand, PWADD presents this type of information in terms of a ranked list. Explaining to a friend that she took one car because the system told him that this car had the highest score of 90% is intuitively not a very convincing justification. Moreover, the decision maker has not the means to partially verify the information provided by PWADD.

In the following sections, the results for other constructs are explained, some of which further help in interpreting the performance of CBTE in terms of decision confidence.

4.5.2 Perceived usefulness, perceived ease of use and trust

Table 4.11 shows that CBTE outperforms PWADD in terms of perceived usefulness and ease of use. This can be explained by noting that decision confidence positively affects perceived usefulness. That is, a system that helps you arrive at a more confident decision is inevitable a more useful system [89].

Table 4.11 also shows that CBTE+PWADD (group 3) is a better choice than CBTE ($p < 0.05$) and PWADD ($p < 0.01$) in terms of perceived ease of use and usefulness and trust. This result can be explained by suggesting that a system combining two distinct approaches provides more functionality and thus is more useful. According to the concept of *individual heterogeneity* described in [80], some individuals may prefer to use the PWADD approach while others may prefer the CBTE approach, and allowing the user to select the approach to use is more beneficial. The concept of *dynamic heterogeneity* also described in [80] as "the changing needs for information during the information acquisition process itself" may explain why CBTE+PWADD is perceived as being more useful (and improve trust, since perceived usefulness is one of its antecedents) because, as the authors found, user control on which type of information to use at any given time improved understanding of a specific task.

Regarding CBTE, participants found this type of system feedback useful in accomplishing the task (85% of the participants) and easy to internalize (62%). Details on each part of CBTE system feedback are shown in Table 4.12. These results show that although the usefulness of CBTE could be compromised by the potential growth in size of the CBTE, that turned not to be the case in this practical situation. In turn, it provides additional evidence that decision justification positively affects the decision confidence.

	Useful			**Easy to internalize**		
	No	**Neutral**	**Yes**	**No**	**Neutral**	**Yes**
GRS	0.00	0.15	0.85	0.08	0.00	0.92
MCS	0.08	0.15	0.77	0.23	0.15	0.62
LCS	0.08	0.23	0.69	0.23	0.23	0.54

Table 4.12: Results of the user study. Perceived usefulness and ease to internalize for participants using the CBTE system. GRS: Global Relaxation Set, MCS: Minimal Conflict set, LCS: Local Relaxation Set

When combining CBTE and PWADD, less people found the CBTE system feedback useful, although the ratio is still positive (62% of the participants), and 54% of the participants found it easy to internalize. Details on each part of the CBTE system feedback for participants using CBTE+PWADD are shown in Table 4.13.

	Useful			**Easy to internalize**		
	No	**Neutral**	**Yes**	**No**	**Neutral**	**Yes**
GRS	0.31	0.23	0.46	0.31	0.31	0.38
MCS	0.31	0.23	0.46	0.23	0.23	0.54
LCS	0.46	0.08	0.46	0.23	0.15	0.62

Table 4.13: Results of the user study. Perceived usefulness and ease to internalize for participants using the CBTE+PWADD system.

We explained in Section 3.4 that CBTE system feedback provides tradeoff information comparable to the way PWADD does. This can explain the fact that the participants using CBTE agreed on 85% that the system was useful for solving the task, while that was the case for only 62% of the participants using CBTE+PWADD. As similar functionality is provided by two approaches, the usefulness of each approach is decreased (users take advantage of only one approach at a given moment in time).

As for trust, four antecedents are described: perceived usefulness, perceived ease of use, enjoyment and perceived competence. The last two antecedents were discarded by the factor analysis. The improvement in perceived usefulness and ease of use of CBTE in comparison to PWADD explains the same increase in trust.

4.5.3 Decision accuracy

The three groups first used the baseline system in equal conditions. As the participants were randomly assigned to each of the three groups, the decision accuracy achieved for the three groups should have been the same. The results show 38%, 38% and 23% average decision accuracy for group G1, G2 and G3 respectively. While 23% seems much lower than 38%, the difference is not statistical significant, with p = 0.40. While different experimental set-ups are typically not comparable, it is worth noting that another experimental study [62] using a similar system as our baseline system, named *iterated form-filling* approach, got very similar results: 35% decision accuracy for participants searching in a catalog of apartments.

In Section 2.4.2 and 3.4 we explained that PWADD and CBTE respectively provide added value information to the baseline approach by which users can make more informative judgments. This added value should be transformed into better decision accuracy. We have seen that group G1 and G2 achieved much higher decision accuracy, of 69% and 61% respectively. However, this comparison can be misleading because of the *carry-over* effect, i.e., a participant could feel more comfortable with the second system only because she had gained practice while using the baseline system.

In Section 3.4 we argued that CBTE and PWADD both provide similar tradeoff information: while PWADD presents this information in terms of a ranked list, CBTE lists the conflicts in the query and presents global and local query relaxations. We therefore hypothesized that both approaches will achieve similar decision accuracy. The results of the experiment show that the difference between decision accuracy, 69% in the case of CBTE and 61% in the case of PWADD, is not statistical significant (p = 0.68). Thus, we can say that the amount of added value is similar.

While in this experiment the amount of added value provided by CBTE and PWADD turns out to be similar in terms of decision accuracy, it could very well be the case that the type of added value is not comparable. In this case, combining CBTE and PWADD in one system could yet increase decision accuracy compared to CBTE or PWADD alone (as we have seen in the previous section, this was the case for perceived usefulness and ease of use). But that was not the case for decision accuracy: combining CBTE and PWADD into one system did not increase (nor decrease) the decision accuracy compared to CBTE or PWADD alone (69% decision accuracy, not statistically significant different to CBTE nor PWADD). This is quite an impressive result which suggests that **not only the amount of added value but also the type of value conveyed by both CBTE and PWADD approaches seems be quite comparable in terms of decision accuracy**. This result validates our hypothesis of Section 3.4.

4.5.4 Domain knowledge

We were intuitively expecting that participants using the CBTE system would achieve more domain knowledge at the end of the decision process. This was not the case and in fact CBTE achieved worse domain knowledge than PWADD (4.73 against 4.90 in the 7-point Likert scale), although the difference is not statistically significant (p=0.55).

However this result is consistent with research model described in Section 4.2 based on the literature review presented in Section 2.5. The research model states that domain knowledge is a determinant of decision accuracy only. We have seen that CBTE and PWADD achieve similar decision accuracy and so it is fair that the level of domain knowledge is also similar.

4.5.5 Perceived control

The *perceived control* construct was tested with the question "I felt more in control of the system", using a 7 point Likert scale, where 1 means strongly disagree, 4 is neutral and 7 stands for strongly agree. Participants in the first group responded that CBTE had the same degree of control as the baseline (4.07). This result is coherent since the CBTE interface is exactly the same as the baseline system, with the only addition of the tradeoff analysis window. However, participants in the second group responded that PWADD gave them fewer degree of control (3.45). This difference is statistically significant (p = 0.018). This result can be explained by the fact that the PWADD interface does not allow users to sort the list of items by an attribute of their choice, as they are ranked by the system using a utility function. Thus, participants felt in less control of the system.

Participants using the PWADD system were asked whether they would like to have the possibility to sort the items by a column, and surprisingly 11 of the 13 participants (84.6%) answered positively, while the other two participants remained neutral. This result adds additional evidence of the lower degree of control.

Participants in the third group using the CBTE+PWADD system answered the question about perceived control with 4.54 but it reached a low statistical significance level (p=0.09) compared to 4.07 in the second group.

Other studies have shown evidences that perceived control positively affects decision confidence [80, 63]. Unfortunately, we cannot verify this hypothesis in our user study because we asked participants whether they felt *more* control in the second system compared to the baseline system, rather than how much degree of control the felt in the second system.

4.5.6 Behavior in stating queries

The three groups exhibited different patterns in their queries, showing that the type of system feedback influenced their behavior. For this purpose, it is instructive to observe the distribution of successful and failing queries for each group.

The interaction logs show that 30.8% of the participants using PWADD never performed a failing query. Taking into account that the major value of PWADD (and WADD) is its compensatory property, i.e. to rank the items when the decision maker has conflicting preferences, this is a quite surprising result because it means that almost one third of the users could never take profit of the major value of PWADD. On the other hand, all the participants using CBTE performed at least one failing query during the whole interaction, and therefore they could take profit of CBTE at least once. In the case of CBTE+PWADD, 23.1% of the participants (as opposed to 30.8% in PWADD and 0% in CBTE) never performed a failing query. **In conclusion, the use of CBTE induced more users to make challenging queries**.

What is the cause for this difference in behavior? As we said in the Section 2.7, people are habituated to use *matching list* system feedback which does not provide any type of help in the case of failing queries, and thus users may be annoyed by this situation. We think that this type of system has influenced users to make less challenging queries first in order to avoid this annoying situation. On the other hand, we suggest that participants using CBTE have learned, through the tutorial, that making a challenging query provides immediate explicit feedback about the conflict that can be easily internalized, as discussed in Section 3.4, and therefore **they were not intimidated by having empty answers**. On the other hand, the fact that feedback provided by PWADD is not explicit but implicitly given in the ranking, maybe the reason of why 30.8% of the participants were not yet encouraged to make challenging queries.

Also, we suggest that users using CBTE learned to revise their criteria according to the items available in the catalog. To support this fact, it is instructive to see the distribution of successful and failing queries taking into account only the participants that issued at least one failing query, i.e. those that made challenging queries. Using PWADD, 51.7% of the queries were failing queries, while this percentage was reduced to 35.5% in the case of CBTE+PWADD and to 21% in the case of CBTE. Figure 4.2 shows the distribution of successful and failing queries for each system given the query size. It is amazing to see that in the case of PWADD, almost all the queries involving more than six attributes were failing queries, whereas in the case of CBTE, participants managed to issue successful queries involving a large number of attributes. Taking into account that participants revised their criteria 14.5 times in average during the whole interaction and that participants using CBTE clicked on average only two relaxations, it is fair to say that **users using CBTE learn better what is available in the catalog, the trade-offs they are facing and can then revise their preferences accordingly, which consequently potentially increases decision confidence**.

Participants using CBTE clicked, on average, 1.3 global relaxations and 0.7 local relaxations and, in the case of CBTE+PWADD, 0.3 global relaxations and 0.6 local relaxations.

4.5.7 CBTE feedback size

In the CBTE and CBTE+PWADD interface, the system displayed CBTE system feedback when the query was over-constrained. We explained in section 3.5 that the size of the global relaxation set, minimal conflict set and local relaxations grows exponentially with the size of the query. Here we show the experimental results regarding this issue in our user study[6].

The global relaxation set (GRS) consists of a set of global relaxations. It can be seen as a trade-off consisting of several choices, i.e. the user needs to choose which relaxation is more appropriate, if any. We suggested that this type of trade-off information could be very valuable to the user because they represent clear ad-hoc questions, like "Do you prefer to pay 80 Euros more, or to accept a one year older car without the ABS feature?". However, if users are overwhelmed with a large global relaxation set, this information would demand a high cognitive effort to process and becomes practically useless. Figure 4.3 shows the distribution of the size of the global relaxation set. In the case of CBTE, it shows that the 84.1% of the GRSs consist of only five or less global relaxations (or in the case of CBTE+PWADD, the 81.1% of the GRSs consisted of six or less global relaxations).

While the global relaxation set consists of a set of global relaxations, a global relaxation consists of a set of constraints to be relaxed (thus defining the size of a global relaxation). For instance, the second global relaxation of Example 3.3 on page 62 is to "change mileage to less than 55'700 km (instead of 50'000 km) and to discard the constraint about color". Figure 4.4 shows the distribution of the size of the global relaxations. In the case of CBTE, it shows that 68.7% of the global relaxations consisted of only one constraint to relax, making it very attractive. In the case of CBTE+PWADD, this quality is degraded, with 70% of the global relaxations consisting of three or less constraints to be relaxed. This difference between CBTE and CBTE+PWADD can be explained by

[6]Note that, as in the previous subsection, two participants were removed for computing the graphs in order to show a clear comparison because they had query sizes of 28 and 31 while the rest of participants had query sizes of 17 or less.

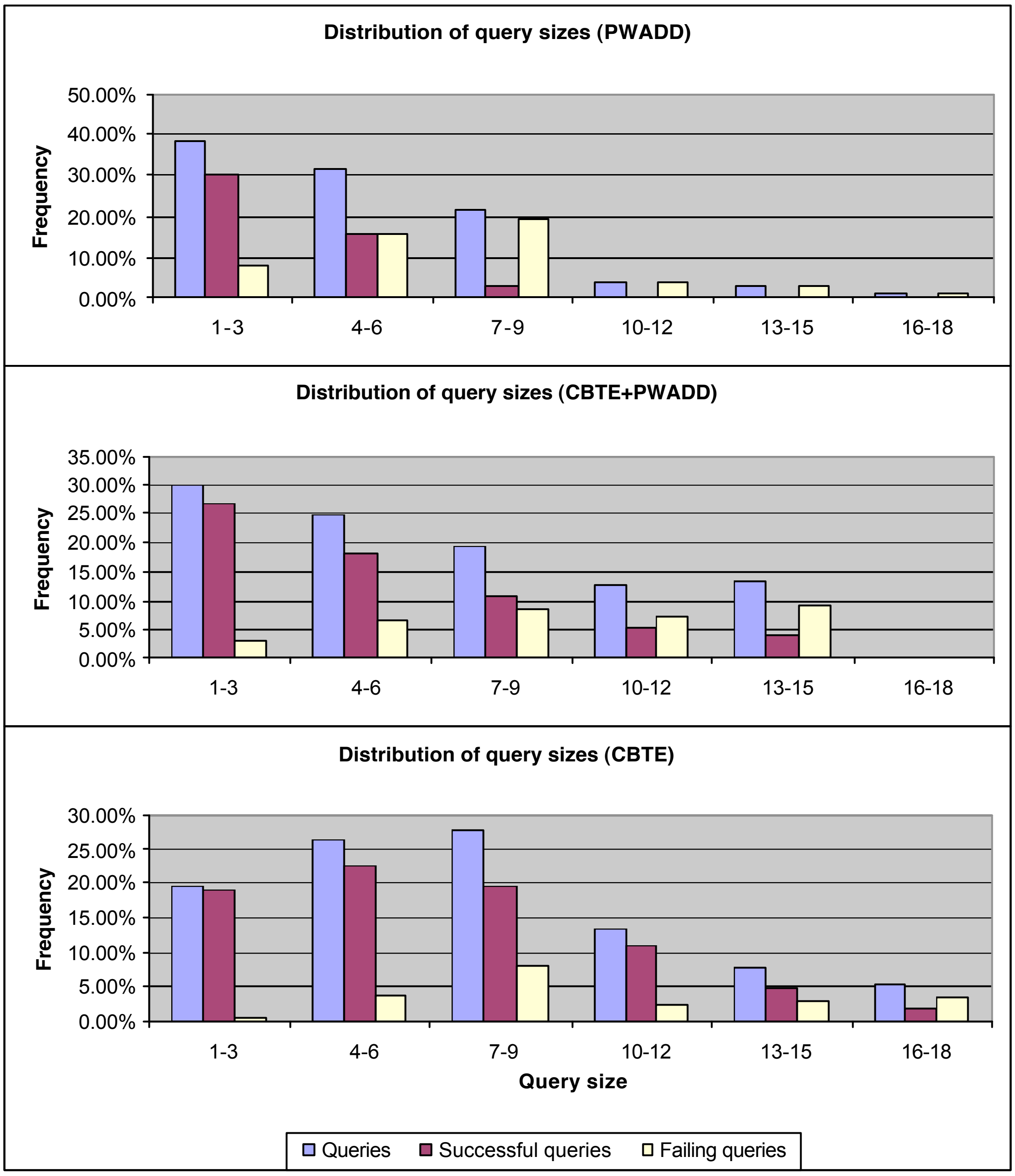


Figure 4.2: Results of the User study. Distribution of query sizes. Participants who never performed a failing query were removed from the distribution. Also, two participants were removed in order to show a clear comparison because they had query sizes of 28 and 31 while the rest of participants had query sizes of 17 or less.

the fact that participants using CBTE adjusted more their queries according to what is actually present in the catalog, as shown in the last subsection.

This is a very interesting result, which suggests that, even if the size of the global relaxation set grows exponentially, this is not a major concern in practical business-to-consumer catalogs and CBTE normally remains usable.

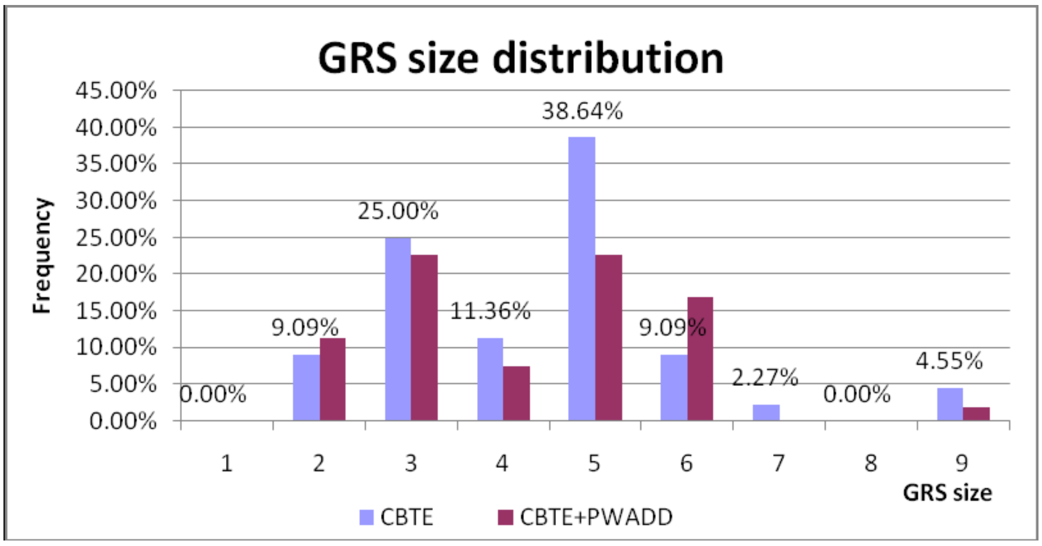


Figure 4.3: Results of the User Study. Distribution of Global Relaxation Set sizes. This graph shows 100% of the points in the case of CBTE. It only shows 83% of the points in the case of CBTE+PWADD, where the biggest GRS has size 21.

The minimal conflict set (MCS) consists of a set of minimal conflicts. It shows precisely which parts of the query are causing the conflicts, and thus it clearly explains what it is not in the catalog. Figure 4.5 shows the distribution of the size of the minimal conflict set. For CBTE, it shows that 50% of the MCSs consisted of only one conflict. As with the GRS, in the case of CBTE+PWADD the size of MCS tends to be more problematic.

While the minimal conflict set consists of a set of conflicts, a conflict consists of a set of constraints (thus defining the size of a conflict). For instance, the first conflict of Example 3.2 on page 62 is the lack of cars in the catalog with a “price cheaper than 6’500 Euros and a mileage of less than 50’000 km”. Figure 4.6 shows the distribution of the size of the conflicts. In both cases, 97% of the conflicts consisted of up to five constraints. Thus, the GRS outperforms the MCS in shortness. This result supports evidence of the subjective measure presented in Section 4.5.2, where 92% of the participants answered that the GRS was easy to internalize, compared to only 62% of the participants in the case of MCS.

In Section 3.6 we ran two simulations in order to also study the size of CBTE feedback.

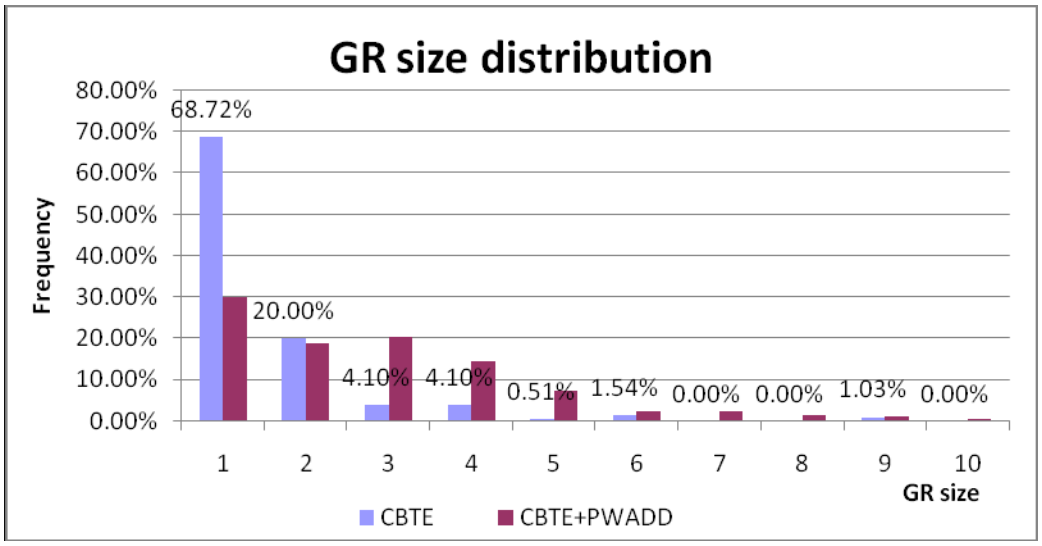


Figure 4.4: Results of the User Study. Distribution of Global Relaxation sizes.

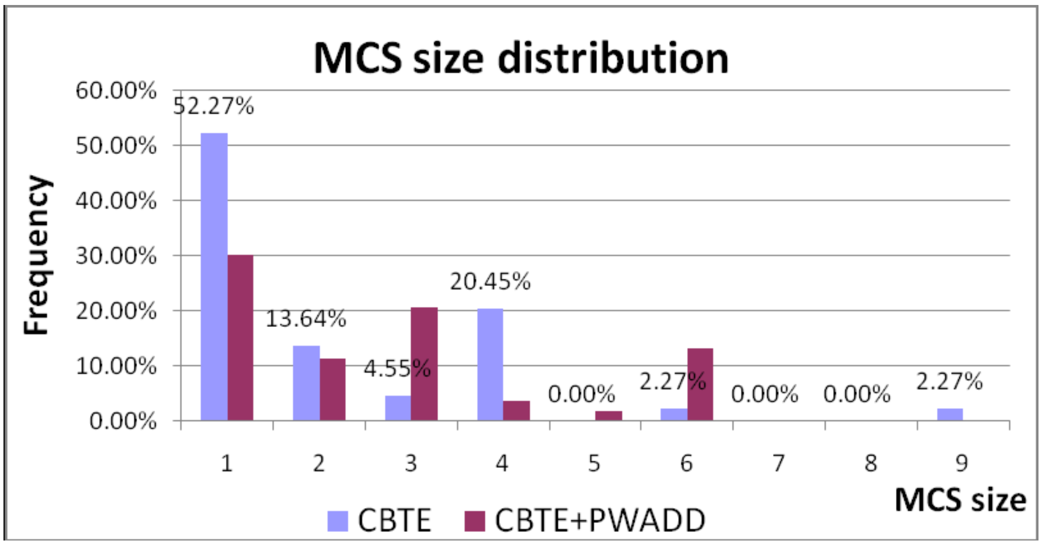


Figure 4.5: Results of the User Study. Distribtion of Minimal Conflict Set sizes. This graph shows 95.4% of the points in the case of CBTE, where the biggest GRS has size 13. It shows 81.13% of the points in the case of CBTE+PWADD, where the biggest GRS has size 48.

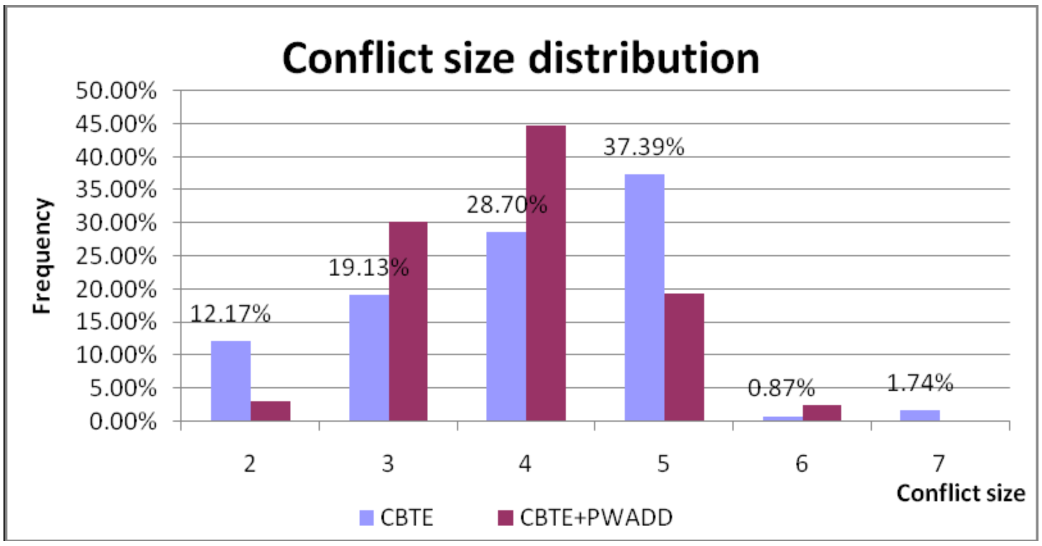


Figure 4.6: Results of the User Study. Distribution of the conflict sizes.

First, a simulation using queries uniformly distributed in size which indeed had quite bad results, and a simulation using queries build in an incremental fashion which had very attractive results. We observed that the experimental results in this study were much closer to the incremental simulation. As users formulate queries incrementally, it seems reasonable that the incremental simulation obtains results that are closer to our experiment than the uniform simulation.

Time to compute the global relaxation set was 0.26 seconds in average (0.36 in CBTE and 0.17 in CBTE+PWADD), while time to compute the minimal conflict set and the local relaxations was 1.6 seconds (2.4 in CBTE and 0.96 in CBTE+PWADD). The participants did not complain about this response time.

4.5.8 Baseline system

As explained in Section 2.7, the user could select the type of predicate associated with each attribute constraint. For instance, a user could ask for cars with a maximum, minimum or exactly a given price. By default, we, as the designers of the DSSEC, pre-defined the choice of the predicates according to standard user profiles, e.g. we know that generally users prefer to indicate a maximum price rather than a minimum price. In this sense, it was quite surprising to see that 33% of the participants changed at least one of the pre-defined predicates for ordered attributes (attributes which domain is a set of ordered values). 33% is not a negligible percentage, and it shows how risky it is to make assumptions that fit everybody. Remember that PWADD makes at least one assumption for each ordered attribute in order to design its associated parametric utility function.

Some DSSECs limit the number of displayed items to a threshold. That was not our case: the system displayed all the matching items (or even all of them, ranked, in the case of PWADD). We saw that while 77% of the participants selected (either as consideration or final selection) an item from the first top four, 23% of the participants selected one item between the fifth and the twentieth position. This suggests that users like and appreciate having the possibility to inspect the comprehensive list of available items. This is confirmed by the final questionnaire, where 77% of the participants said that they preferred to see the comprehensive list of items, while 13% preferred to have only the first top three (10% of the participants did not care).

PWADD and CBTE+PWADD
While participants revised the value of a parametric utility function 14.5 times on average, they only revised the weights 6.1 times, in the case of PWADD, and 3.6 times in the case of CBTE+PWADD (in CBTE the participants could not set weights). This is quite surprising, as in other similar DSSECs participants tend to not change the weights, and it adds evidence that the guidelines described in Section 2.7 solve some of the common well known usability drawbacks in DSSECs.

4.6 Limitations of the user study

Sensitivity
In our comparative between-groups experimental design we have lost the sensitivity inherent to a with-in groups design. For instance, we could have asked a participant about the usefulness of CBTE in comparison to PWADD and average afterwards this measure with all the participants, instead of measuring the average usefulness of CBTE on a group of participants in comparison with the average usefulness of PWADD on another group.

Measures after study
We asked the participants in the study to imagine that they had the intention to buy a used car in the very near future, but actually this was not the case for most of them. In this situation we could not assess the *purchase intention* measure because users in the study were participating in a hypothetical situation with no real intention of purchasing a car. Being able to assess this measure would have allowed us to provide evidence about the existence of a relationship between confidence and purchase intention.

Catalog domain
The test domain for this experimental study was a catalog of used cars. We cannot generalize these results for other type of domains. Moreover, the size of the catalog was rather small, 200 cars and it is dangerous to extrapolate the results for bigger catalogs. Even more significant, the number of participants was limited to 39, which represents a

rather small study.

Tutorial
When visiting an e-vendor website, users do not generally learn to use their website through a tutorial before actually using it. However, in our user study we provided participants with this training. Therefore we cannot foresee which system, CBTE or PWADD, would perform best for the first time an individual is using the system.

Validation phase
Results show between 61% and 69% decision accuracy using CBTE and PWADD systems. However, we think that these scores are inflated. There are two reasons for believing this. Firstly, decision accuracy for PWADD in our study is much higher than the reported in a similar study [62] using a similar system named *example-critiquing*, 61% compared to 40% (The study in [62] is using a similar set-up as the one we discussed in section 4.5.3 in which decision accuracy matched for our baseline system and their equivalent approach, named *iterated form-filling* approach). Even if both systems are in principle similar (query-feedback search interaction model using PWADD system feedback), it is true that our system had some basic interface usability improvements as described in Section 2.7 that may *partially* justify the difference in performance (see for instance Section 4.5.8 in where we show that our interface users revised their weights on average 6.1 times during a process, while in other systems such in [62] participants barely revised the weights simply because of, in our opinion, usability issues). A second reason to doubt the accuracy of these results is that this validation phase is negatively affected by the *confirmation bias.* The confirmation bias is the tendency to search for or interpret information in a way that confirms one's preconceptions, whereas decision makers search for evidence that confirms their hypothesis (or decision), rather than evidence that could disconfirm it [70]. More evidence of this bias comes from [4], showing that once impressions have been formed, they exhibit remarkable perseverance. Thus, once participants make a decision, they will tend to stick to it, i.e. they will not be ready to change their previous choice. In this case, as only some of the participants will make the extra effort to carefully look again for the best decision, this validation phase tends to overestimate decision accuracy. In this sense, the results for decision accuracy are not to be taken as absolute measures, but only as relative ones in order to compare CBTE against PWADD.

4.7 Discussion and related work

The objective of the *weighted additive* decision strategy (and PWADD) is to achieve good decision accuracy, and it is not concerned with decision confidence. However, we can see that as predicted by [89] and described in the research model (see Section 2.5 and Section 4.2), decision accuracy has a positive impact on decision confidence: participants using PWADD increased their decision accuracy from 38% to 61.5% and at the same time they were more confident of their choices (measured as 4.95 in a 7-point Likert scale).

In [63], the authors compare two types of DSSECs, *user-motivated critiquing* and *system-proposed critiquing*, in terms of decision accuracy, effort and confidence. They show that *user-motivated critiquing* achieved better decision confidence than *system-proposed.* Thus, we may be tempted to say that the first approach is designed with the objective to improve decision confidence. However the first system also was better in terms of decision accuracy. Thus, the improvement in terms of decision confidence could be based solely on the positive effect of the improvement in decision accuracy as discussed earlier.

In our study, we have seen that while CBTE and PWADD achieve similar decision accuracy, CBTE results in more decision confidence. Thus, we can conclude that the positive increment of decision confidence in CBTE is not only based on the positive effect of the improvement in decision accuracy, but also to the compliance with the properties given in the *decision confidence in query-feedback search* (DECOFE) framework. This result adds some evidence to our claim in Section 3.1 that improving decision accuracy and decision confidence are two fundamentally different objectives requiring each a different approach, and gives some support to the DECOFE framework in this regard.

4.8 Conclusions

We have set-up a between-group experimental design to evaluate the performance improvement (with respect to the baseline system) of CBTE, PWADD and of a combination of CBTE and PWADD (CBTE+PWADD), and we carried out an experimental evaluation that produced positive evidences for the validity of the DECOFE framework introduced in Section 3.3. However, results need to be interpreted with the caveat that the size of the experiment was limited.

The experimental study shows that CBTE outperforms PWADD in decision confidence while decision accuracy is not being degraded. The study also provides evidence that the positive increment of decision confidence in CBTE is not only based on the positive effect of the improvement in decision accuracy, but also on the compliance with the properties given in the *decision confidence in query-feedback search* (DECOFE) framework. This result adds some evidence to our claim in Section 3.1 that improving decision accuracy and decision confidence are two fundamentally different objectives each requiring a different approach.

Participants in our experiment found that CBTE system feedback was useful to accomplish the task (85% of the participants) and easy to internalize (62%). As an objective measure, the size of the *minimal conflict set* (and CBTE in general) was reasonably small, and thus the potential exponential growth of CBTE was not a concern for this study. These results are consistent with the incremental simulation explained in Section

3.6. Also, computational time was not much of a concern even with the use of a basic exhaustive search algorithm, except for few cases.

An interesting result concerning the behavior of participants in the experiment is that CBTE system feedback incited more participants to make more challenging queries, and thus to state more accurate objectives rather than mean objectives (see the example of Section 1.3.2 where decision makers tend to state mean objectives). In particular, participants were not worried by failing queries because they were supported by the CBTE system feedback. This is a very beneficial behavior because DSSECs can better help users if they state precise objectives.

Another interesting result is the empirical evidence that CBTE provides tradeoff information comparable to PWADD and therefore both approaches have similar decision accuracy, while the advantage of CBTE is an increase in decision confidence and perceived ease of use.

The experiment showed that CBTE and PWADD can be effectively combined, leading to improvements in terms of perceived usefulness and ease of use, trust, domain knowledge and perceived control.

Chapter 5

Conclusions

The research on decision confidence for *decision support systems for electronic catalogs* (DSSECs) is quite recent (with some preliminary experiments in 2000 [35]) and, with the notable exception of Pearl Pu et al since 2004 [53, 56, 63], has mainly focused on carrying out experiments in order to understand the antecedents and benefits of decision confidence, rather than understanding the direct effect of different types of *system feedback* on decision confidence. We have seen that current research mainly focuses on novel types of *system feedback* based on preferences rather than constraints. We proposed a framework to study decision confidence building, and argued that system feedback based on constraints, though not novel as such, can be combined in order to improve decision confidence. Based on this analysis, we proposed a new type of system feedback, the *confidence building tradeoff explanations* (CBTE) feedback which explains the query tradeoffs in terms of conflicts and relaxations that can potentially contribute to decision confidence building. In a comparative user study, CBTE outperformed the *parametric weighted additive* (PWADD) ranked list feedback by directly improving decision confidence and stimulating users to make more challenging queries. These results, however, need to be accepted with the caveat of limited experiment size.

We summarize the key results of our work and discuss the remaining open issues below.

5.1 Summary of main results

Query-feedback search
We have analyzed several DSSECs found in the literature and described a generalized model, the *query-feedback search* interaction model, an iterative process in which a user makes *queries* to the system in order to develop her *domain knowledge* and *preferences* through the *system feedback*, until she finds a good decision. Then, we have positioned the different DSSECs into the query-feedback search interaction model by describing the

class of system feedback they provide: *matching list*, *ranked list*, *diversity*, *suggestions*, *system-proposed critiquing*, *conflict*, *corrective* and *visualization.*

A baseline system
We have described a baseline DSSEC that solves most of the known deficiencies of rudimentary DSSECs, while remaining very simple. We argue that such a system can be used in user studies to achieve more meaningful comparative evaluations of innovative DSSECs.

Query-feedback search and decision confidence
We integrated decision confidence building into the *query-feedback search* interaction model by extending its definition as an iterative process in which a user makes queries to the system in order to develop her *domain knowledge*, *preferences* and *confidence* for *tentative decisions* through the *system feedback*, until she is *convinced* of her final decision.

System feedback properties to improve decision confidence
Intuitively, system feedback designed to improve decision accuracy should be based on *prescriptive* decision theory (i.e. how individuals should make decisions in order to find the optimal one) such as using the weighted additive decision strategy. On the other hand, system feedback designed to improve decision confidence should be based on *descriptive* decision theory (i.e. how individuals actually make decisions) such as using the elimination-by-aspects decision strategy. That is, users should feel more confident by using a decision strategy that they are used to. In this direction, we identified four properties for the design of system feedback that can potentially contribute to decision confidence building: (1) *constraint-based*, (2) *universally valid*, (3) *negatory* and (4) *unique in its class.* An experimental evaluation produced positive evidence in this direction.

Conflict and corrective feedback, good candidates to improve decision confidence
Using the DECOFE framework, we have studied the potential contribution to decision confidence building of several types of system feedback proposed in the literature. Within this framework, we have argued that *conflict* and *corrective* feedback (combined with either *matching list* or *ranked list* feedback) were the most promising types of feedback for decision confidence building. This claim is supported by experimental evaluation in the case of ranked list feedback compared to conflict and corrective feedback.

CBTE improves decision confidence and stimulates making more challenging questions
Based on this analysis, we proposed the *confidence building tradeoff explanation* (CBTEs) feedback. We argued that system feedback based on constraints, even if not novel as such, can be combined in order to directly improve decision confidence. This claim is supported by an experimental study comparing CBTE and PWADD. In typical situations where there is a mismatch between what the decision maker wants and what is it in the catalog,

CBTE provides a clear trade-off question to help the user make a good decision through the *global relaxation set*, and provides evidence for building decision confidence through the *minimal conflict set* and the *local relaxation set.*

In the same experimental evaluation, we saw that CBTE system feedback incited more participants to make more challenging queries, and thus to state more accurate objectives rather than mean objectives. This is a very beneficial behavior because DSSECs can better help them if users state precise objectives.

***Minimal Conflict Set* and *Global Relaxation Set* exponential sizes not a concern**
The size of MCS and GRS increases exponentially with the size of the query. A large feedback may be difficult to internalize and is thus useless. In our user study, the size of MCS and GRS feedback was reasonably small, even for queries of size 18. These results are consistent with the incremental simulation run in Section 3.6. This result suggests that the potential problem growth of MCS and GRS may not be a concern for practical B2C catalogs.

5.2 Discussion and future work

Further evaluation of the DECOFE framework
We carried out a user study which provided some positive feedback to the DECOFE framework. However, this work is far from being complete. In particular, experiments involving other types of system feedback reviewed in the framework would give more insight about the DECOFE validity. It is also important to find a procedure to evaluate the quality of a decision justification.

A step in this direction would be to ask a participant to convince another participant about her choice (after she has chosen an item with a given interface). This procedure would most probably suffer from a strong non-systematic error, but this should be canceled out over a sufficient number of observations. This *sufficient* number is a critical parameter for the usability of the procedure, since it is very hard and costly to recruit participants. Another possibility to evaluate the quality of decision justification could be to ask the participant to write down the justification, and then count the number of correct statements.

More generally, while we have reviewed some basic literature on cognitive science in this thesis thanks to disinterested help from researchers in the cognitive science domain, we think that a much stronger collaboration with cognitive scientist is necessary for designing new metrics and, in general, for designing DSSECs.

Generalization of the DECOFE framework and CBTE feedback
In this work, we assumed decision theory under certainty. While this is the most typical

scenario in e-commerce, it is not necessarily true in the general case: the consequences of actions are generally uncertain. For instance, a consumer may decide between buying a photo camera either in a traditional store or in some not fully-reliable auction website for a cheaper price, with the uncertainty that she may not receive the product. Thus, the decision maker needs to take this uncertainty into account when making a decision.

We also assumed that we have a homogeneous catalog, that is, a catalog with a predefined set of fix attributes. However, sometimes we need to choose among a set of non homogeneous products. For instance, there are many diverse options to choose a present for a child, such as consumable craft supplies, outdoor toys or board games, which cannot be directly compared.

A possible direction would be to derive a set of homogeneous attributes from heterogeneous products. For instance, for any type of product it is possible to say whether the product is convenient for a child or not (e.g. it cannot have sharp edges). Thus, we can derive a boolean attribute called "convenient for a child" for every product. We can then base our decision using a set of such attributes. The difficulty relies on deriving such homogeneous catalog from a heterogeneous one. Techniques such as clustering and information extraction could be used.

Computationally efficient confidence building feedback
In Chapter 3 we designed a type of system feedback that can potentially build decision confidence, while ignoring the algorithmic complexity of this process. Although this is not a problem for the user study in Chapter 4 during which we observed that, for two real-life business-to-consumer catalogs (with respectively 10 and 35 attributes), the computation leads to short enough response times, this may not be the case for other domains. Moreover, if such feedback would have to be computed for several users simultaneously, as is the case in web e-commerce, the task would become intractable. Thus, it would be interesting to study how to pre-compute feedback for all the possible situations and encode them in a compact form for later quick retrieval. The method should be able to incorporate the catalogue changes into the encoded form without needing to start the whole analysis from scratch. Current work in this direction is the use of *assumption-based CSPs* and precompiled CSPs [26], and algorithms such as QuickXPlain [25].

Adaptive feedback
Decision makers search for relevant information, arrive at a tentative decision and then attempt to justify that decision [71, 74]. The first step can be guided by system feedback that improves decision accuracy by stimulating the expression of preferences, while the second step needs feedback providing evidence for the correctness of the user's tentative decision, in order to improve decision confidence. It could be interesting to study how the system can detect in an unobtrusive way that the second step has began, so that it can adapt the type of feedback provided to users.

A possible direction comes from the observation that adding an item to a comparison matrix may be a strong evidence for the start of this second phase. However, as more complex interactions may arise (for instance, a user considering more than one item before committing to a decision), the use of pattern recognition in the query updates could be used.

New types of system feedback
In Section 3.3 we proposed four properties for the design of *system feedback* that potentially contribute to the construction of the *decision justification*. We then proposed a new type of feedback, the confidence building tradeoff analysis, fulfilling these properties. It would be of course interesting to study other types of system feedback.

For instance, Simonson et al. introduced in [7] the hypothesis of *tradeoff contrast*, which states that the tendency to prefer an item is enhanced or hindered depending on whether the tradeoffs within the set of items under consideration are favorable or unfavorable to that item. Let's illustrate this concept with an adapted example also proposed in [7]: a consumer is evaluating two cars, where the first costs 10'000 Euros and has 50'000 km mileage and the second costs 10'200 Euros and has 45'000 km. The choice between the first and second car depends on whether the consumer is willing to pay 200 Euros more for having 5'000 km mileage less. The tradeoff contrast hypothesis predicts that the consumer is more likely to select the second car if the set of cars under consideration includes pairs of cars for which the cost for lower mileage is greater than that implied by the comparison between the two cars under evaluation.

It would be interesting to exploit the concept of *tradeoff contrasts* for the production of more convincing feedback. The main difficulty would be to select the most interesting tradeoff contrasts to show, as the number of possibilities increases exponentially with the number of attributes.

A possibility for overcoming this difficulty arises from the observation that users prefer feedback which is short (as in our user study participants used relaxations with one or two constraints only). Therefore, selecting those tradeoff constrasts which can be explained concisely would be preferred.

Another question would be whether it is better to consider all the items in the catalog or only those under consideration. In our study, the average size of the considered item set was two, and therefore this would not be enough for such comparison.

Integrating collaborative recommendation systems to content-based search
Recommender systems (RS), often implemented using a collaborative filtering algorithm, attempt to predict products that a user may be interested in given some information about the user's profile. This profile may be, for example, the purchase history of the user, or a set of ratings about some products. RSs differ from content-based search in the sense that they help to *discover* products that the users may even not know about. However, two of the main drawbacks of RSs are the cold start problem (users are required to rate several products the first time they use the system) and the latency problem (new products are not recommended because they are not yet rated by enough

users). We think that RSs provide very valuable information (such as the ratings or the purchase history) that could be exploited in content-based search in such a way that the two above-mentioned drawbacks could be eliminated.

Consider, for example, a user that enjoyed the movies “Toy Story” and “Finding Nemo”. She could enter a DSSEC system for the first time and ask to be shown only the movies that other people who enjoyed these same two films, also enjoyed. While this would have an effect similar to using an RS by rating these two movies, using a content-based search would give an immediate benefit and much more control to the user. For instance, the user could ask to filter the previous results by showing only new movies that appeared in the previous week, and thus the latency problem would be reduced, as the few ratings of the new movie would not be compared with numerous ratings of the old movies. Similarly, in the case where the user would be interested in old rather than new movies, it would be straightforward to filter using the content-based system. Therefore, we believe that it could be very interesting to study how the privileged information that RCs provide could be queried in a DSSEC. Applying current types of system feedback used in DSSEC (designed to increase decision accuracy and confidence) to this type of information could prove to be very useful.

Eliciting an accurate utility function from natural interactions
We have seen in the example of Section 1.3.2 that people are reluctant to answer unnatural questions such as the ones generated when the WADD strategy is used, even if they are told that they will accurately find the best item. However, within more natural conversations, such as in our example of traditional commerce or in EBA or PWADD-like decision strategies, some of the elicited preferences can be used to construct a normative WADD utility function. If this utility function would be accurate enough to provide an initial ranking that incites users to trust the system, they may be ready to complete the elicitation of the current utility function to increase decision accuracy. The open issue however is how to transfer EBA and PWADD-like utilities to a normative (generalized) additive utility function.

Take the following example as a potential solution: assuming additive independence, consider that the user asks for cars for 6’500 Euros maximum, blue color and airbag, and there are no matching cars with this query. By discarding the constraint about color, we could make an initial guess that blue color has a score of 100 and the other colors have a score of 50. When being presented with this guess, the user could realize the benefit of fine-tuning this individual utility as in the example of Section 1.3.2. Also, based on the user consideration set, the system could guess some initial weights and present the swing-weight assessment table to the user for fine-tuning. The difficulty of such an approach however, strongly increases when the additive independence, which is commonly not satisfied, is not assumed anymore.

A way around this could be to use a normative method that guarantees achieving good decision accuracy. However, the main drawback of such an approach is the resulting low decision confidence arising from the fact that the decision maker had not internalized a decision justification. Although there is no a straightforward and obvious way to solve this, it would be interesting to investigate how, once a complete preference model have been elicited, a decision aid could create a set of explanations which are potentially useful for the decision maker to internalize a decision justification and achieve better decision confidence.

Appendix A

User study files

In this appendix there are the surveys and tutorials used in the experiment, as well as the responses from the 39 participants.

Table A.1: Demographics questionnaire.

Page 1: Personal information

- Gender: ◯ male ◯ female
- Age: ◯ <18 ◯ 18-24 ◯ 25-35 ◯ 36-45 ◯ 46-55 ◯ >55
- Occupation: ....
- Is English your native language? ◯ yes ◯ no

Page 2: Familiarity with computers

- Approximately how many hours per day do you use a computer?
  - ◯ every day
  - ◯ a few times a week
  - ◯ a few times a month
  - ◯ a few times a year
  - ◯ never
- Computer abilities:
  - ◯ I rarely use computers
  - ◯ I am a normal user (use MS Office, surf the web)
  - ◯ I know how to configure my operating system

- In what computer language is this query written in? (ignore the question if you don't know it)

  ```
  SELECT shape, COUNT(*) FROM info GROUP BY shape;
  ```

  ....
  ....

Page 3: Decision making

- Do you use Internet to compare products/services? ◯ yes ◯ no
- Do you use specialized websites to compare products (other that search engines such as Google)?

  ◯ yes

  Which ones? (for instance: http://www.resto-rang.ch to compare restaurants)..
  ....
  ....

  ◯ no
- Do you use computer tools to compare decision/making decisions?

  ◯ yes

  Which ones? (for instance, "MS Excel", "Logical Decisions for Windows")..
  ....
  ....

  ◯ no

Page 4: Relevant abilities for the scenario

- Select the option which most describes you car abilities:

  ◯ I rarely drive cars
  ◯ I am a normal car user (I use it to go to work or during week-ends)
  ◯ I know what are the typical values for car performance and cubic capacity
  ◯ I read car magazines regularly

Id	Question
1	I understood the task I was asked to do
2	The task was a little ambiguous
3	It was easy to complete this task
4	I felt comfortable with this task
5	I knew enough about cars to accomplish this task
6	I am satisfied with my choice
7	I am confident that I have found the best possible choice from the catalog
8	I think that I have found a very good choice
9	I often became confused when I used the system
10	I made errors frequently when using the system
11	Interacting with the system was often frustrating
12	I found it easy to recover from errors encountered while using the system
13	The system was rigid and inflexible to interact with
14	I found it easy to get the system to do what I wanted it to do
15	The system often behaved in unexpected ways
16	I felt in control of the system
17	I found it cumbersome to use the system
18	My interaction with the system was easy for me to understand
19	The system provided helpful guidance to interact with it
20	This is a user friendly system
21	I can quickly find the information I need on this system
22	The system was easy to use for product assessment
23	The system was easy to use to compare products
24	The system was easy to use to accomplish the task
25	Overall, I found the system easy to use
26	This system provided good quality information
27	About the products, I got a good ”picture/understanding” of the different possibilities for my region of interest
28	This system was useful for product assessment
29	This system was useful for comparing products
30	This system was useful for accomplishing the task
31	I found my interaction with this system interesting
32	I found my interaction with this system entertaining
33	I found my interaction with this system enjoyable
34	I found my interaction with this system pleasant
35	I would feel that this online vendor is honest
36	I would feel that this online vendor is trustworthy
37	I would feel that this online vendor cares about customers
38	I would feel that this online vendor would provide me with good service

Table A.2: Questions asked after the use of the baseline system.All items were measured on a seven-point Likert strongly disagree/strongly agree scale.

Id	**Question**
Q1	With the improved system, it was easier to accomplish the task
Q2	I am more satisfied with my choice
Q3	I am more confident that I have found my optimal choice from the ones in the catalog
Q4	I have found a better choice
Q5	I became more confused when I used the system
Q6	I made more errors when using the system
Q7	Interacting with the system was more frustrating
Q8	I found it easier to recover from errors encountered while using the system
Q9	The system was more rigid and inflexible to interact with
Q10	I found it easier to get the system to do what I wanted it to do
Q11	The system behaved more in unexpected ways
Q12	I felt more in control of the system
Q13	I found it more cumbersome to use the system
Q14	My interaction with the system was easier for me to understand
Q15	The system provided more helpful guidance to interact with it
Q16	This system is more user-friendly
Q17	I can more quickly find the information I need on this system
Q18	The system was easier to use for product assessment
Q19	The system was easier to use to compare products
Q20	The system was easier to use to accomplish the task
Q21	Overall, I found the system easier to use
Q22	This system provided better quality information
Q23	About the products, I got a better ”picture/understanding” of the different possibilites for my region of interest
Q24	This system was more useful for product assessment
Q25	This system was more useful for comparing products
Q26	This system was more useful for accomplishing the task
Q27	This system improved my performance in accomplishing the task
Q28	This system increased my effectivenes for accomplishing the task
Q29	I found my interaction with this system more interesting
Q30	I found my interaction with this system more enjoyable
Q31	I would feel that this online vendor is more honest
Q32	I would feel that this online vendor is more trustworthy
Q33	I would feel that this online vendor is more competent
Q34	I would feel that this online vendor cares more about customers

Table A.3: Questions asked after the use of the second system.

Id	**Question**
Q35	After the validation, I am satisfied with the choice that I took using the Improved System
Q36	After the validation, I am confident that using the Improved System I found the best possible choice from the catalog
Q37	After the validation, I think that I found a very good choice using the Improved System
Q38	After the validation, I think that the Improved System was more useful than the Baseline System

Table A.4: Questions asked after the validation phase

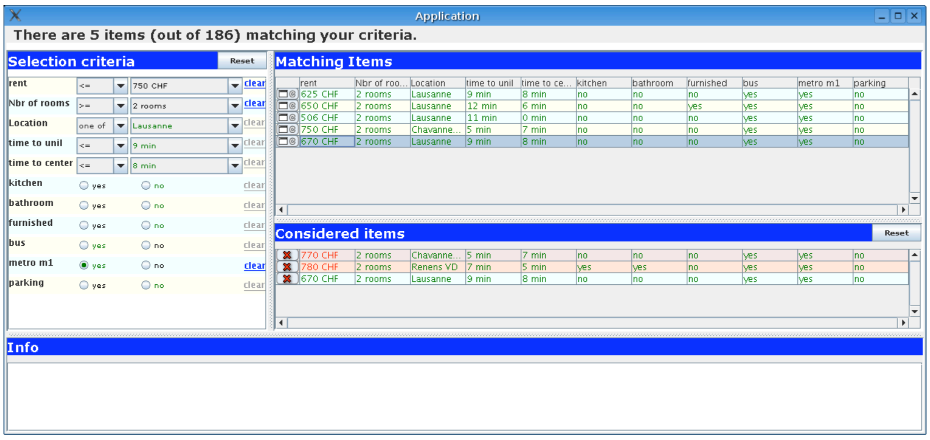
User Study. Baseline Tutorial. Page 1 / 5.

Baseline System: ECatalog. Tutorial, David Portabella Clotet, June 2006

Baseline System: ECatalog. Tutorial

Through this tutorial, you are going to learn to use the baseline system, called "ECatalog".

Here you have a screenshot of the user interface.



There are 5 items (out of 186) matching your criteria.

Selection criteria

rent	<=	750 CHF
Nbr of rooms	>=	2 rooms
Location	one of	Lausanne
time to unil	<=	9 min
time to center	<=	8 min
kitchen	yes	no
bathroom	yes	no
furnished	yes	no
bus	yes	no
metro m1	[✓ yes]	no
parking	yes	no

Matching Items

rent	Nbr of roo...	Location	time to unil	time to ce...	kitchen	bathroom	furnished	bus	metro m1	parking
625 CHF	2 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no
650 CHF	2 rooms	Lausanne	12 min	6 min	no	no	yes	yes	yes	no
506 CHF	2 rooms	Lausanne	11 min	0 min	no	no	no	yes	yes	no
750 CHF	2 rooms	Chavanne...	5 min	7 min	no	no	no	yes	yes	no
670 CHF	2 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no

Considered items

770 CHF	2 rooms	Chavanne...	5 min	7 min	no	no	no	yes	yes	no
780 CHF	2 rooms	Renens VD	7 min	5 min	yes	yes	no	yes	yes	no
670 CHF	2 rooms	Lausanne	9 min	8 min	no	no	no	yes	yes	no

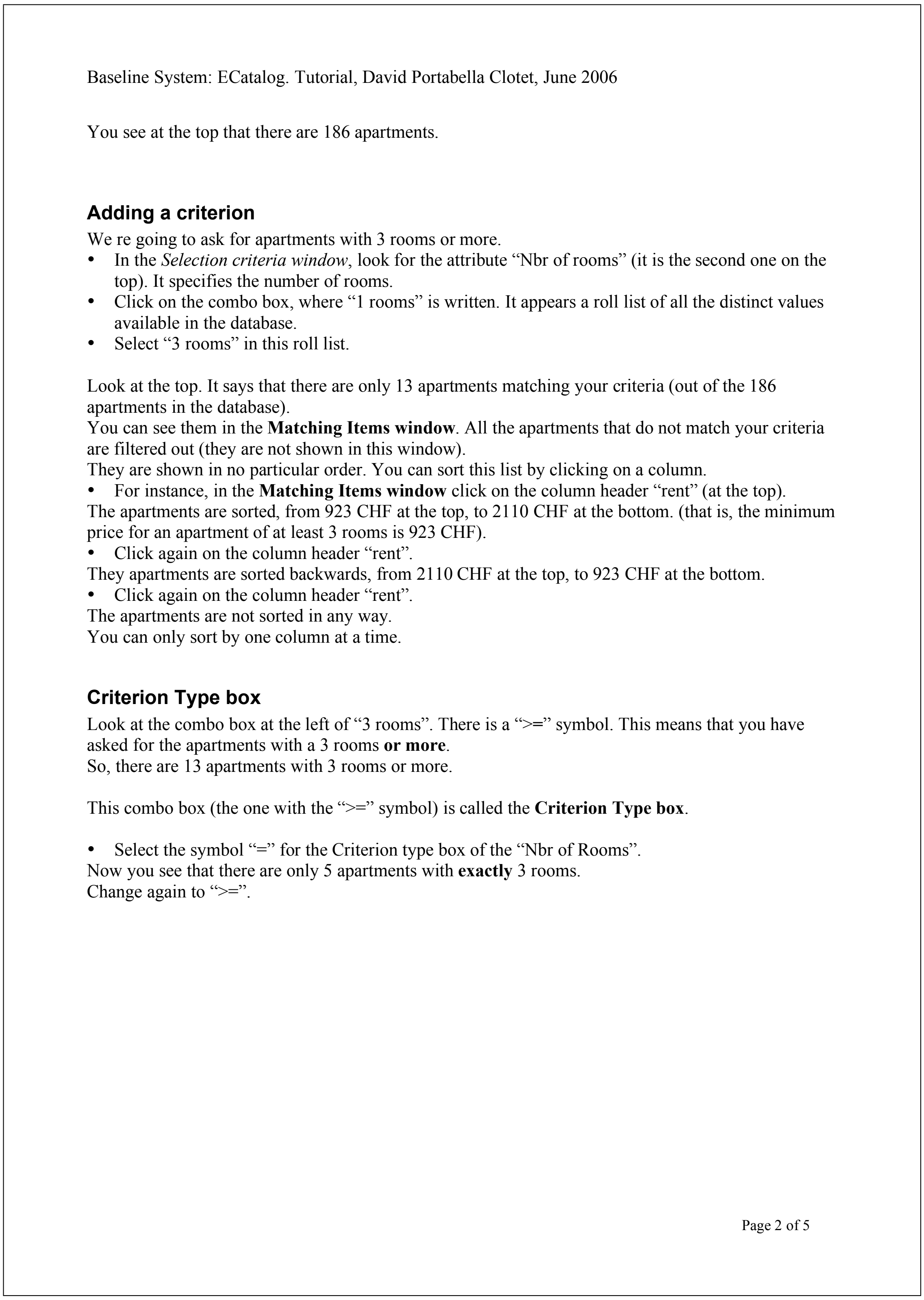
There are four main windows:

- On the left there is the **Selection Criteria window**.
  You can specify several criterions to filter out items that you are not interested in.
- On the top right there is the **Matching Items window**.
  Here you see the items that match your criteria
- On the bottom right there is the **Considered Items window**.
  Here you can put the items that you are considering. It allows you to compare items.
- At the bottom there is a window that it is not used at all in this system.

This database contains 186 apartments to rent taken from a database of UNIL.

Please follow this tutorial to learn to use the Advanced Catalog System.

User Study. Baseline Tutorial. Page 2 / 5.



Baseline System: ECatalog. Tutorial, David Portabella Clotet, June 2006

You see at the top that there are 186 apartments.

Adding a criterion

We re going to ask for apartments with 3 rooms or more.

- In the *Selection criteria window*, look for the attribute "Nbr of rooms" (it is the second one on the top). It specifies the number of rooms.
- Click on the combo box, where "1 rooms" is written. It appears a roll list of all the distinct values available in the database.
- Select "3 rooms" in this roll list.

Look at the top. It says that there are only 13 apartments matching your criteria (out of the 186 apartments in the database).
You can see them in the **Matching Items window**. All the apartments that do not match your criteria are filtered out (they are not shown in this window).
They are shown in no particular order. You can sort this list by clicking on a column.

- For instance, in the **Matching Items window** click on the column header "rent" (at the top).

The apartments are sorted, from 923 CHF at the top, to 2110 CHF at the bottom. (that is, the minimum price for an apartment of at least 3 rooms is 923 CHF).

- Click again on the column header "rent".

They apartments are sorted backwards, from 2110 CHF at the top, to 923 CHF at the bottom.

- Click again on the column header "rent".

The apartments are not sorted in any way.
You can only sort by one column at a time.

Criterion Type box

Look at the combo box at the left of "3 rooms". There is a ">=" symbol. This means that you have asked for the apartments with a 3 rooms **or more**.
So, there are 13 apartments with 3 rooms or more.

This combo box (the one with the ">=" symbol) is called the **Criterion Type box**.

- Select the symbol "=" for the Criterion type box of the "Nbr of Rooms".

Now you see that there are only 5 apartments with **exactly** 3 rooms.
Change again to ">=".

User Study. Baseline Tutorial. Page 3 / 5.

Baseline System: ECatalog. Tutorial, David Portabella Clotet, June 2006

Selection Criteria window

Your current criteria is “Nbr of Rooms >= 3”. You can see this in the Selectrion Criteria window because the value “3 rooms” is displayed in black, while the other values (rent, time to unil and time to center) are displayed in green.
For instance, the value for the “rent” attribute is “1380 CHF”. As it is shown in green, it means that it is only an **example** (actually, it corresponds to the apartment selected in the “Matching Items” window, or the first one if none is selected). By “**example**” we mean that this value is not used to filter out apartments: you can see in the “Matching Items” window that there are apartments of more than 1380 CHF.

So, if you would like apartments of 1380 CHF or less, you need to add this criterion:
- In the *Selection criteria window*, look for the attribute “rent” (it is the first one on the top). It specifies the monthly rent of the apartment.
- Click on the combo box, where “1380 CHF” is written. It appears a roll list of all the distinct values available in the database.
- Select “1380 CHF” in this roll list.

Now you can see that the values “1380 CHF” and “3 rooms” are in black.
There are only 3 apartments that fulfill your criteria (1380 CHF or less, AND 3 rooms or more).

Remove the last criterion that you added:
- On the right of the combobox where you set the value “1380 CHF” there is a “clear” button. Click on there.

Now, you have again the 13 apartments, with 3 rooms or more.

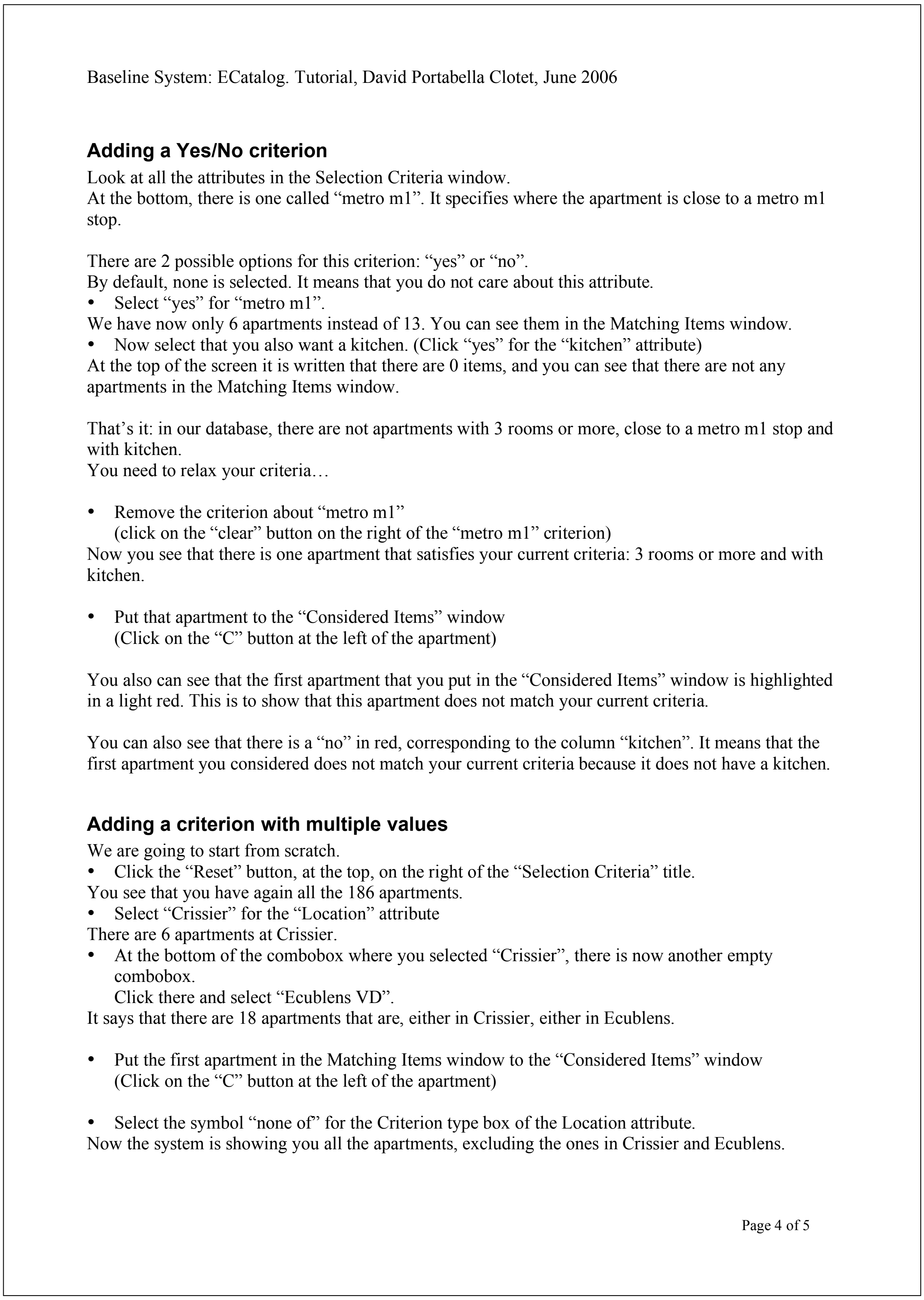
Considered Items window

At the right bottom you can temporarily place your items for latter comparison.
- Click on the second apartment present in the Matching Items window (anywhere in the line, for instance, where it is written “1600 CHF”.

That entire apartment is now highlighted.
- Click on the “C” button at the left of the apartment.

It has been placed in the “Considered Items” window.

User Study. Baseline Tutorial. Page 4 / 5.



Baseline System: ECatalog. Tutorial, David Portabella Clotet, June 2006

Adding a Yes/No criterion

Look at all the attributes in the Selection Criteria window.
At the bottom, there is one called “metro m1”. It specifies where the apartment is close to a metro m1 stop.

There are 2 possible options for this criterion: “yes” or “no”.
By default, none is selected. It means that you do not care about this attribute.

- Select “yes” for “metro m1”.

We have now only 6 apartments instead of 13. You can see them in the Matching Items window.

- Now select that you also want a kitchen. (Click “yes” for the “kitchen” attribute)

At the top of the screen it is written that there are 0 items, and you can see that there are not any apartments in the Matching Items window.

That’s it: in our database, there are not apartments with 3 rooms or more, close to a metro m1 stop and with kitchen.
You need to relax your criteria…

- Remove the criterion about “metro m1”
  (click on the “clear” button on the right of the “metro m1” criterion)

Now you see that there is one apartment that satisfies your current criteria: 3 rooms or more and with kitchen.

- Put that apartment to the “Considered Items” window
  (Click on the “C” button at the left of the apartment)

You also can see that the first apartment that you put in the “Considered Items” window is highlighted in a light red. This is to show that this apartment does not match your current criteria.

You can also see that there is a “no” in red, corresponding to the column “kitchen”. It means that the first apartment you considered does not match your current criteria because it does not have a kitchen.

Adding a criterion with multiple values

We are going to start from scratch.

- Click the “Reset” button, at the top, on the right of the “Selection Criteria” title.

You see that you have again all the 186 apartments.

- Select “Crissier” for the “Location” attribute

There are 6 apartments at Crissier.

- At the bottom of the combobox where you selected “Crissier”, there is now another empty combobox.
  Click there and select “Ecublens VD”.

It says that there are 18 apartments that are, either in Crissier, either in Ecublens.

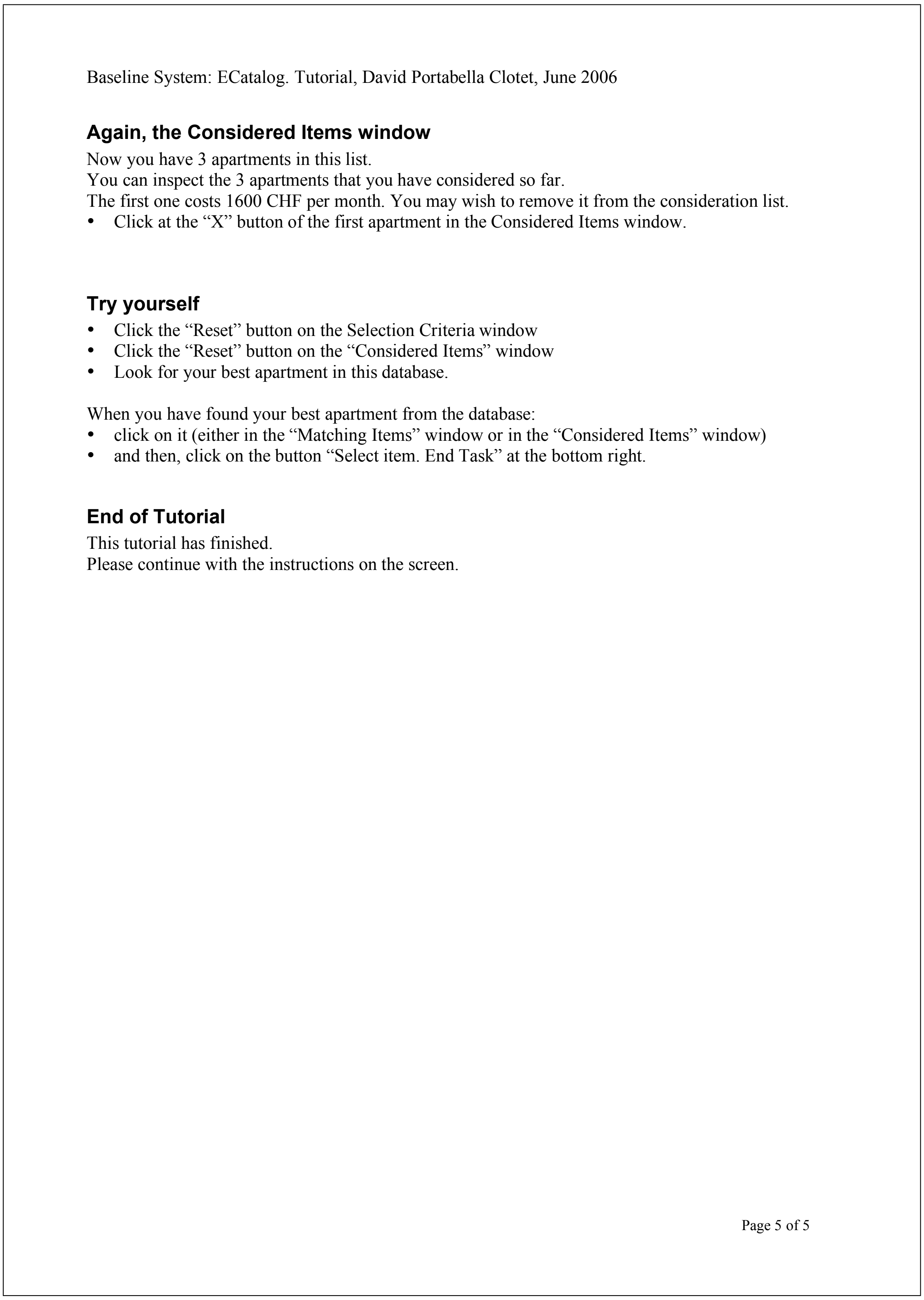
- Put the first apartment in the Matching Items window to the “Considered Items” window
  (Click on the “C” button at the left of the apartment)

- Select the symbol “none of” for the Criterion type box of the Location attribute.

Now the system is showing you all the apartments, excluding the ones in Crissier and Ecublens.

Page 4 of 5

User Study. Baseline Tutorial. Page 5 / 5.



Baseline System: ECatalog. Tutorial, David Portabella Clotet, June 2006

Again, the Considered Items window

Now you have 3 apartments in this list.
You can inspect the 3 apartments that you have considered so far.
The first one costs 1600 CHF per month. You may wish to remove it from the consideration list.
- Click at the "X" button of the first apartment in the Considered Items window.

Try yourself

- Click the "Reset" button on the Selection Criteria window
- Click the "Reset" button on the "Considered Items" window
- Look for your best apartment in this database.

When you have found your best apartment from the database:
- click on it (either in the "Matching Items" window or in the "Considered Items" window)
- and then, click on the button "Select item. End Task" at the bottom right.

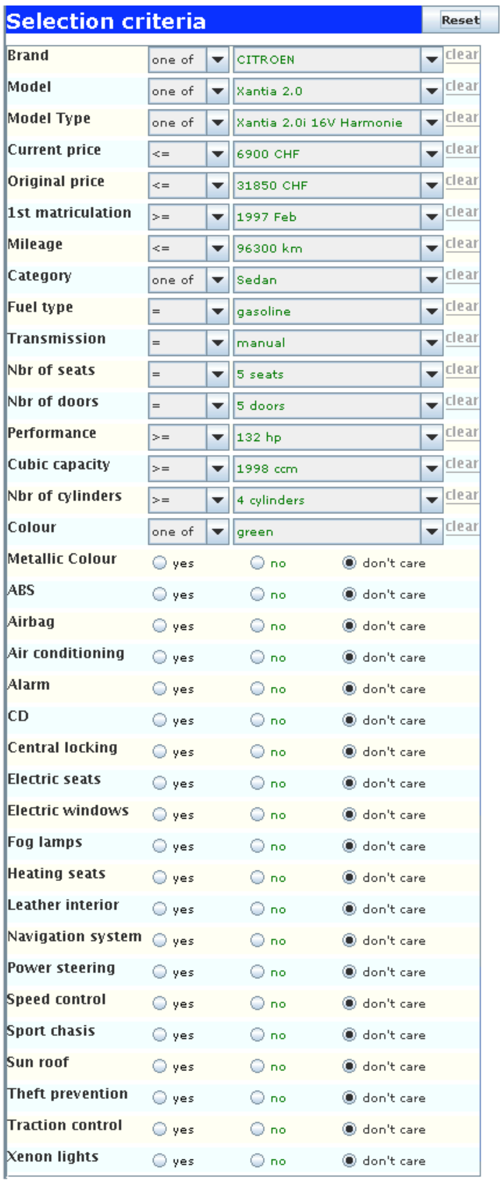
End of Tutorial

This tutorial has finished.
Please continue with the instructions on the screen.

Page 5 of 5

User Study. Second-hand cars database. Page 1 / 2.

Second-hand cars database.
Here you can see the list of attributes in the "Selection Criteria" window.



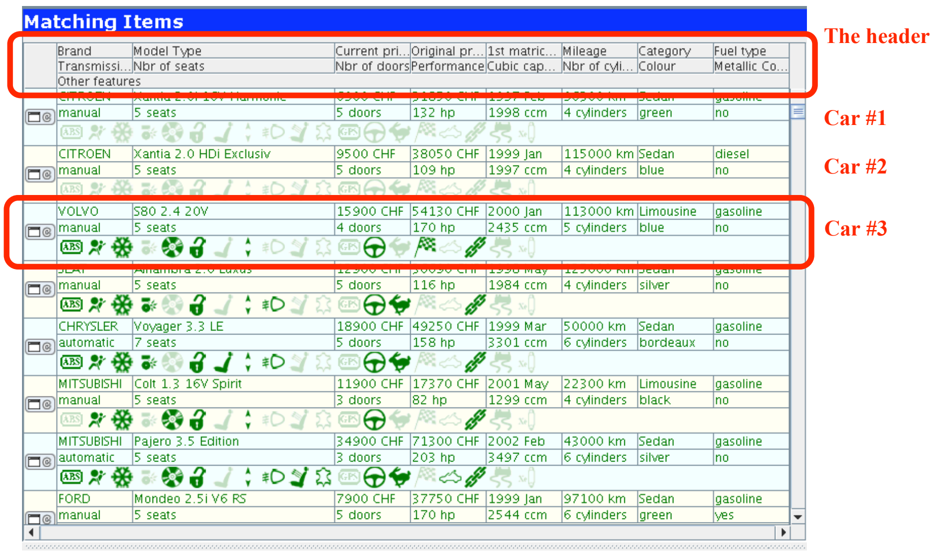
Selection criteria Reset

Brand	one of	CITROEN	clear
Model	one of	Xantia 2.0	clear
Model Type	one of	Xantia 2.0i 16V Harmonie	clear
Current price	<=	6900 CHF	clear
Original price	<=	31850 CHF	clear
1st matriculation	>=	1997 Feb	clear
Mileage	<=	96300 km	clear
Category	one of	Sedan	clear
Fuel type	=	gasoline	clear
Transmission	=	manual	clear
Nbr of seats	=	5 seats	clear
Nbr of doors	=	5 doors	clear
Performance	>=	132 hp	clear
Cubic capacity	>=	1998 ccm	clear
Nbr of cylinders	>=	4 cylinders	clear
Colour	one of	green	clear
Metallic Colour	yes	no	[✓ don't care]
ABS	yes	no	[✓ don't care]
Airbag	yes	no	[✓ don't care]
Air conditioning	yes	no	[✓ don't care]
Alarm	yes	no	[✓ don't care]
CD	yes	no	[✓ don't care]
Central locking	yes	no	[✓ don't care]
Electric seats	yes	no	[✓ don't care]
Electric windows	yes	no	[✓ don't care]
Fog lamps	yes	no	[✓ don't care]
Heating seats	yes	no	[✓ don't care]
Leather interior	yes	no	[✓ don't care]
Navigation system	yes	no	[✓ don't care]
Power steering	yes	no	[✓ don't care]
Speed control	yes	no	[✓ don't care]
Sport chasis	yes	no	[✓ don't care]
Sun roof	yes	no	[✓ don't care]
Theft prevention	yes	no	[✓ don't care]
Traction control	yes	no	[✓ don't care]
Xenon lights	yes	no	[✓ don't care]

User Study. Second-hand cars database. Page 2 / 2.

Second-hand cars database.

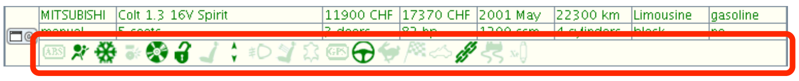
Here you can see how a car is displayed in the "Matching Items" window.



Brand	Model Type	Current pri...	Original pr...	1st matric...	Mileage	Category	Fuel type
Transmissi...	Nbr of seats	Nbr of doors	Performance	Cubic cap...	Nbr of cyli...	Colour	Metallic Co...
Other features							
CITROEN	Xantia 2.0i 16V Harmonie	[illegible]	[illegible]	1997 Feb	[illegible]	Sedan	gasoline
manual	5 seats	5 doors	132 hp	1998 ccm	4 cylinders	green	no
CITROEN	Xantia 2.0 HDi Exclusiv	9500 CHF	38050 CHF	1999 Jan	115000 km	Sedan	diesel
manual	5 seats	5 doors	109 hp	1997 ccm	4 cylinders	blue	no
VOLVO	S80 2.4 20V	15900 CHF	54130 CHF	2000 Jan	113000 km	Limousine	gasoline
manual	5 seats	4 doors	170 hp	2435 ccm	5 cylinders	blue	no
SEAT	Alhambra 2.0 Luxus	[illegible]	[illegible]	1998 May	[illegible]	Sedan	gasoline
manual	5 seats	5 doors	116 hp	1984 ccm	4 cylinders	silver	no
CHRYSLER	Voyager 3.3 LE	18900 CHF	49250 CHF	1999 Mar	50000 km	Sedan	gasoline
automatic	7 seats	5 doors	158 hp	3301 ccm	6 cylinders	bordeaux	no
MITSUBISHI	Colt 1.3 16V Spirit	11900 CHF	17370 CHF	2001 May	22300 km	Limousine	gasoline
manual	5 seats	3 doors	82 hp	1299 ccm	4 cylinders	black	no
MITSUBISHI	Pajero 3.5 Edition	34900 CHF	71300 CHF	2002 Feb	43000 km	Sedan	gasoline
automatic	5 seats	3 doors	203 hp	3497 ccm	6 cylinders	silver	no
FORD	Mondeo 2.5i V6 RS	7900 CHF	37750 CHF	1999 Jan	97100 km	Sedan	gasoline
manual	5 seats	5 doors	170 hp	2544 ccm	6 cylinders	green	yes

All the attributes in the Selection Criteria window are shown in this compact representation.

The icons



MITSUBISHI	Colt 1.3 16V Spirit	11900 CHF	17370 CHF	2001 May	22300 km	Limousine	gasoline

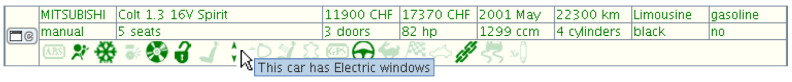
These icons correspond to the attributes ABS, Airbag, Air conditioning… until Xenon lights, in the same order than in the "Selection Criteria" window.

If the color of the icon is dark green, it means that the car has this attribute. For instance, the second icon corresponds to Airbag. As it is in dark green, it means that this car has Airbag.

If the color of the icon is light green, it means that the car has not this attribute. For instance, the first icon corresponds to ABS. As it is in light green, it means that this car has not ABS.

Tooltip for the icons

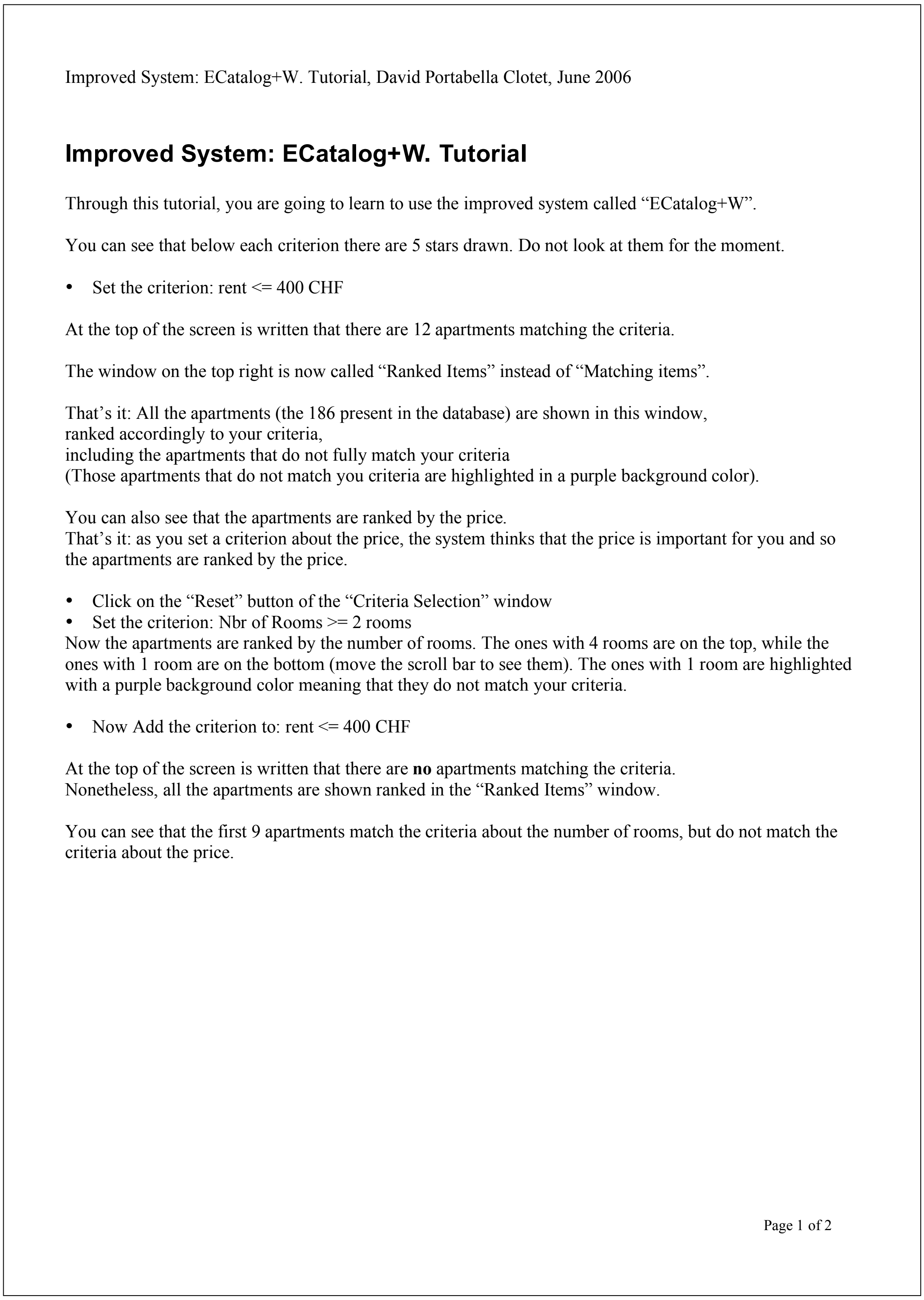
If you move your mouse over the icon, it will display a tool tip showing the name of the attribute.



MITSUBISHI	Colt 1.3 16V Spirit	11900 CHF	17370 CHF	2001 May	22300 km	Limousine	gasoline
manual	5 seats	3 doors	82 hp	1299 ccm	4 cylinders	black	no

This car has Electric windows

User Study. Ecatalog+W Tutorial. Page 1 / 2.



Improved System: ECatalog+W. Tutorial

Through this tutorial, you are going to learn to use the improved system called “ECatalog+W”.

You can see that below each criterion there are 5 stars drawn. Do not look at them for the moment.

- Set the criterion: rent <= 400 CHF

At the top of the screen is written that there are 12 apartments matching the criteria.

The window on the top right is now called “Ranked Items” instead of “Matching items”.

That’s it: All the apartments (the 186 present in the database) are shown in this window,
ranked accordingly to your criteria,
including the apartments that do not fully match your criteria
(Those apartments that do not match you criteria are highlighted in a purple background color).

You can also see that the apartments are ranked by the price.
That’s it: as you set a criterion about the price, the system thinks that the price is important for you and so the apartments are ranked by the price.

- Click on the “Reset” button of the “Criteria Selection” window
- Set the criterion: Nbr of Rooms >= 2 rooms

Now the apartments are ranked by the number of rooms. The ones with 4 rooms are on the top, while the ones with 1 room are on the bottom (move the scroll bar to see them). The ones with 1 room are highlighted with a purple background color meaning that they do not match your criteria.

- Now Add the criterion to: rent <= 400 CHF

At the top of the screen is written that there are **no** apartments matching the criteria.
Nonetheless, all the apartments are shown ranked in the “Ranked Items” window.

You can see that the first 9 apartments match the criteria about the number of rooms, but do not match the criteria about the price.

User Study. Ecatalog+W Tutorial. Page 2 / 2.

Improved System: ECatalog+W. Tutorial, David Portabella Clotet, June 2006

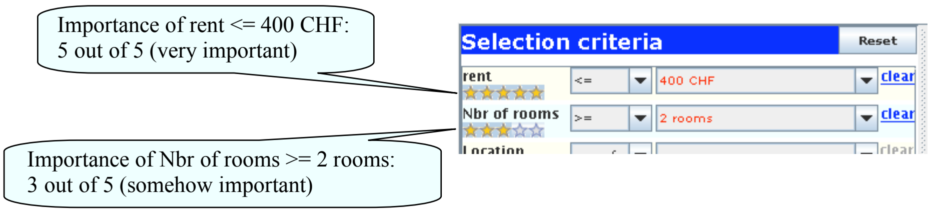
Setting the importance of a criterion

Let's say that you want to explain to the system that the price is more important than the number of rooms. How to do this?

In the "Selection Criteria" window, below the "rent" criterion, there are five stars. The first three are in yellow. This means that the importance of this criterion is 3 out of 5.

- Click at the star number five

You see? Now the five starts are in yellow. This means that you have set the importance of this criterion 5 out of 5, i.e. the maximum.


Importance of rent <= 400 CHF: 5 out of 5 (very important)
Importance of Nbr of rooms >= 2 rooms: 3 out of 5 (somehow important)
Selection criteria
Reset
rent
<=
400 CHF
clear
Nbr of rooms
>=
2 rooms
clear
Location
clear

You can see in the "Ranked Items" window that now the apartments at the top match the criterion about the price.

By default, the importance of the criterion is 3 out of 5.

- Change the importance of the criterion "Nbr of Rooms" to 1 out of 5, i.e. the minimum.

Now there are more apartments at the top matching the criterion about the price.

Note: In this system you cannot sort the list of items by clicking on a column.

Summarizing

Setting the importance of a criterion to "5 out 5" means that that criterion is very important.
Setting the importance of a criterion to "1 out 5" means that that criterion is important, but not very much.

Try yourself

- Click the "Reset" button on the Selection Criteria window
- Click the "Reset" button on the "Considered Items" window
- Look for your best apartment in this database.

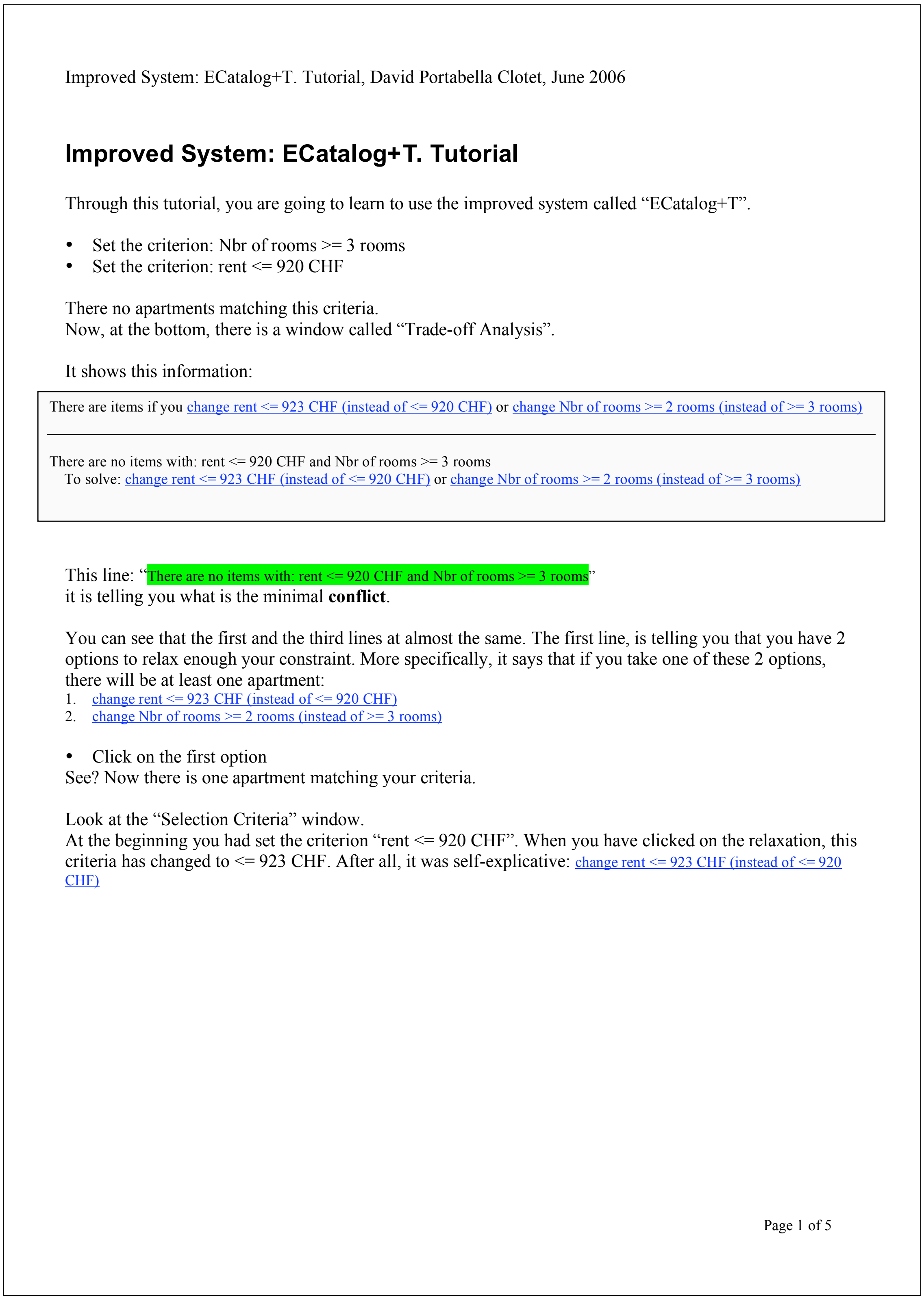
When you have found your best apartment from the database:

- click on it (either in the "Matching Items" window or in the "Considered Items" window)
- and then, click on the button "Select item. End Task" at the bottom right.

End of Tutorial

This tutorial has finished.
Please continue with the instructions on the screen.

User Study. Ecatalog+T Tutorial. Page 1 / 5.



Improved System: ECatalog+T. Tutorial, David Portabella Clotet, June 2006

Improved System: ECatalog+T. Tutorial

Through this tutorial, you are going to learn to use the improved system called "ECatalog+T".

- Set the criterion: Nbr of rooms >= 3 rooms
- Set the criterion: rent <= 920 CHF

There no apartments matching this criteria.
Now, at the bottom, there is a window called "Trade-off Analysis".

It shows this information:

There are items if you change rent <= 923 CHF (instead of <= 920 CHF) or change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)

There are no items with: rent <= 920 CHF and Nbr of rooms >= 3 rooms
To solve: change rent <= 923 CHF (instead of <= 920 CHF) or change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)

This line: "There are no items with: rent <= 920 CHF and Nbr of rooms >= 3 rooms"
it is telling you what is the minimal **conflict**.

You can see that the first and the third lines at almost the same. The first line, is telling you that you have 2 options to relax enough your constraint. More specifically, it says that if you take one of these 2 options, there will be at least one apartment:
1. change rent <= 923 CHF (instead of <= 920 CHF)
2. change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)

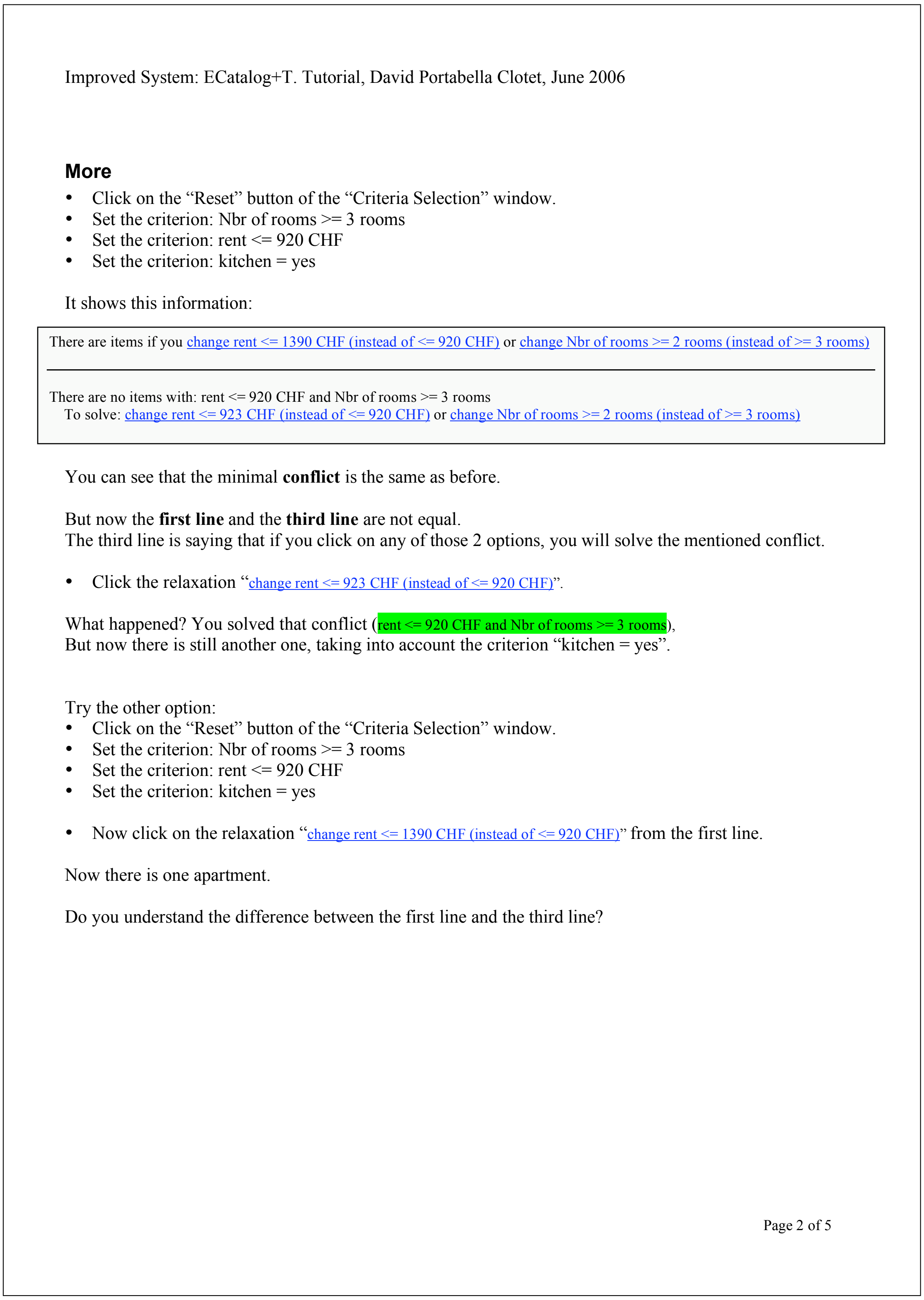
- Click on the first option

See? Now there is one apartment matching your criteria.

Look at the "Selection Criteria" window.
At the beginning you had set the criterion "rent <= 920 CHF". When you have clicked on the relaxation, this criteria has changed to <= 923 CHF. After all, it was self-explicative: change rent <= 923 CHF (instead of <= 920 CHF)

Page 1 of 5

User Study. Ecatalog+T Tutorial. Page 2 / 5.



Improved System: ECatalog+T. Tutorial, David Portabella Clotet, June 2006

More

- Click on the “Reset” button of the “Criteria Selection” window.
- Set the criterion: Nbr of rooms >= 3 rooms
- Set the criterion: rent <= 920 CHF
- Set the criterion: kitchen = yes

It shows this information:

There are items if you change rent <= 1390 CHF (instead of <= 920 CHF) or change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)

There are no items with: rent <= 920 CHF and Nbr of rooms >= 3 rooms
To solve: change rent <= 923 CHF (instead of <= 920 CHF) or change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)

You can see that the minimal **conflict** is the same as before.

But now the **first line** and the **third line** are not equal.
The third line is saying that if you click on any of those 2 options, you will solve the mentioned conflict.

- Click the relaxation “change rent <= 923 CHF (instead of <= 920 CHF)”.

What happened? You solved that conflict (rent <= 920 CHF and Nbr of rooms >= 3 rooms),
But now there is still another one, taking into account the criterion “kitchen = yes”.

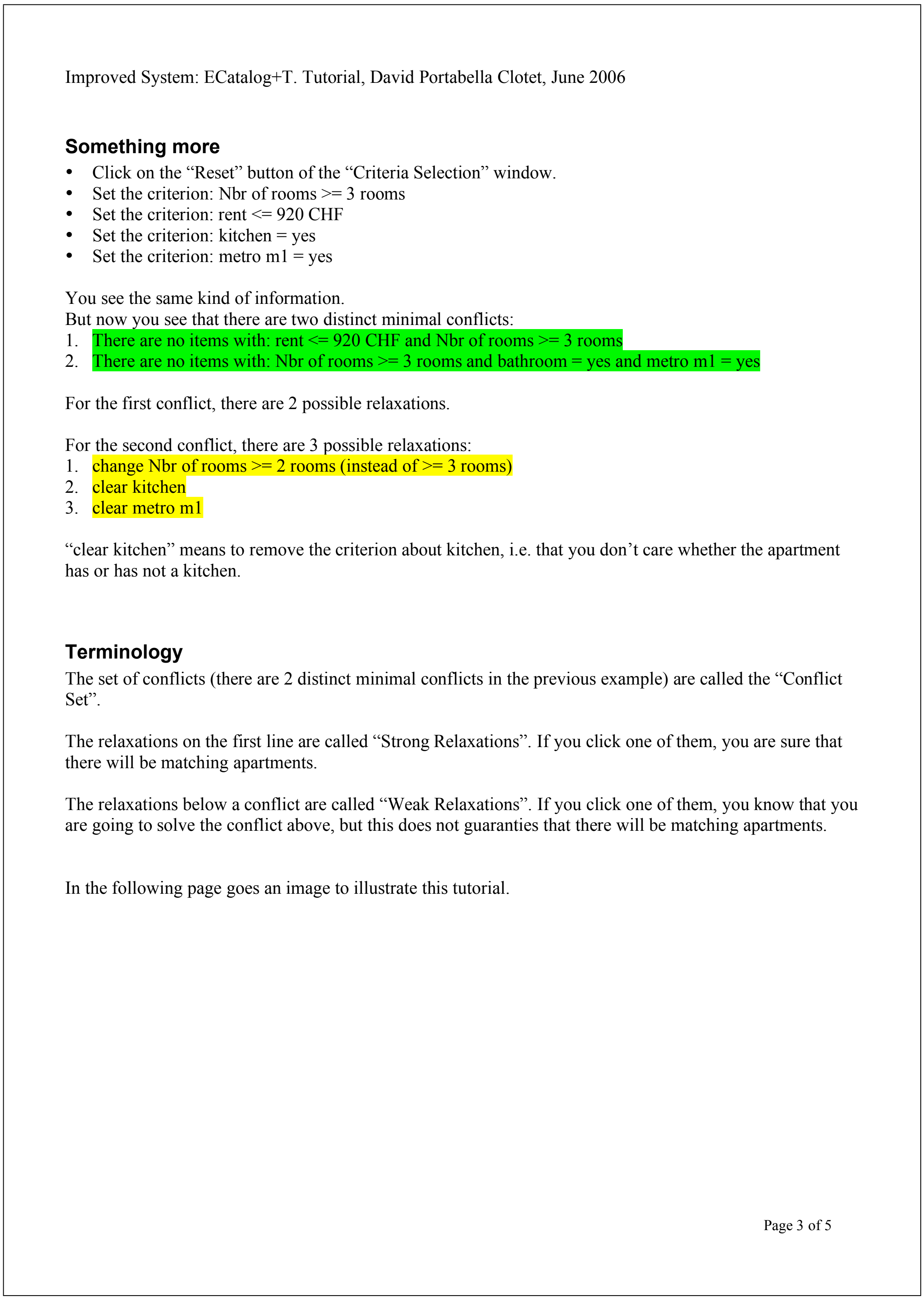
Try the other option:
- Click on the “Reset” button of the “Criteria Selection” window.
- Set the criterion: Nbr of rooms >= 3 rooms
- Set the criterion: rent <= 920 CHF
- Set the criterion: kitchen = yes

- Now click on the relaxation “change rent <= 1390 CHF (instead of <= 920 CHF)” from the first line.

Now there is one apartment.

Do you understand the difference between the first line and the third line?

User Study. Ecatalog+T Tutorial. Page 3 / 5.



Improved System: ECatalog+T. Tutorial, David Portabella Clotet, June 2006

Something more

- Click on the “Reset” button of the “Criteria Selection” window.
- Set the criterion: Nbr of rooms >= 3 rooms
- Set the criterion: rent <= 920 CHF
- Set the criterion: kitchen = yes
- Set the criterion: metro m1 = yes

You see the same kind of information.
But now you see that there are two distinct minimal conflicts:

1. There are no items with: rent <= 920 CHF and Nbr of rooms >= 3 rooms
2. There are no items with: Nbr of rooms >= 3 rooms and bathroom = yes and metro m1 = yes

For the first conflict, there are 2 possible relaxations.

For the second conflict, there are 3 possible relaxations:

1. change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)
2. clear kitchen
3. clear metro m1

“clear kitchen” means to remove the criterion about kitchen, i.e. that you don’t care whether the apartment has or has not a kitchen.

Terminology

The set of conflicts (there are 2 distinct minimal conflicts in the previous example) are called the “Conflict Set”.

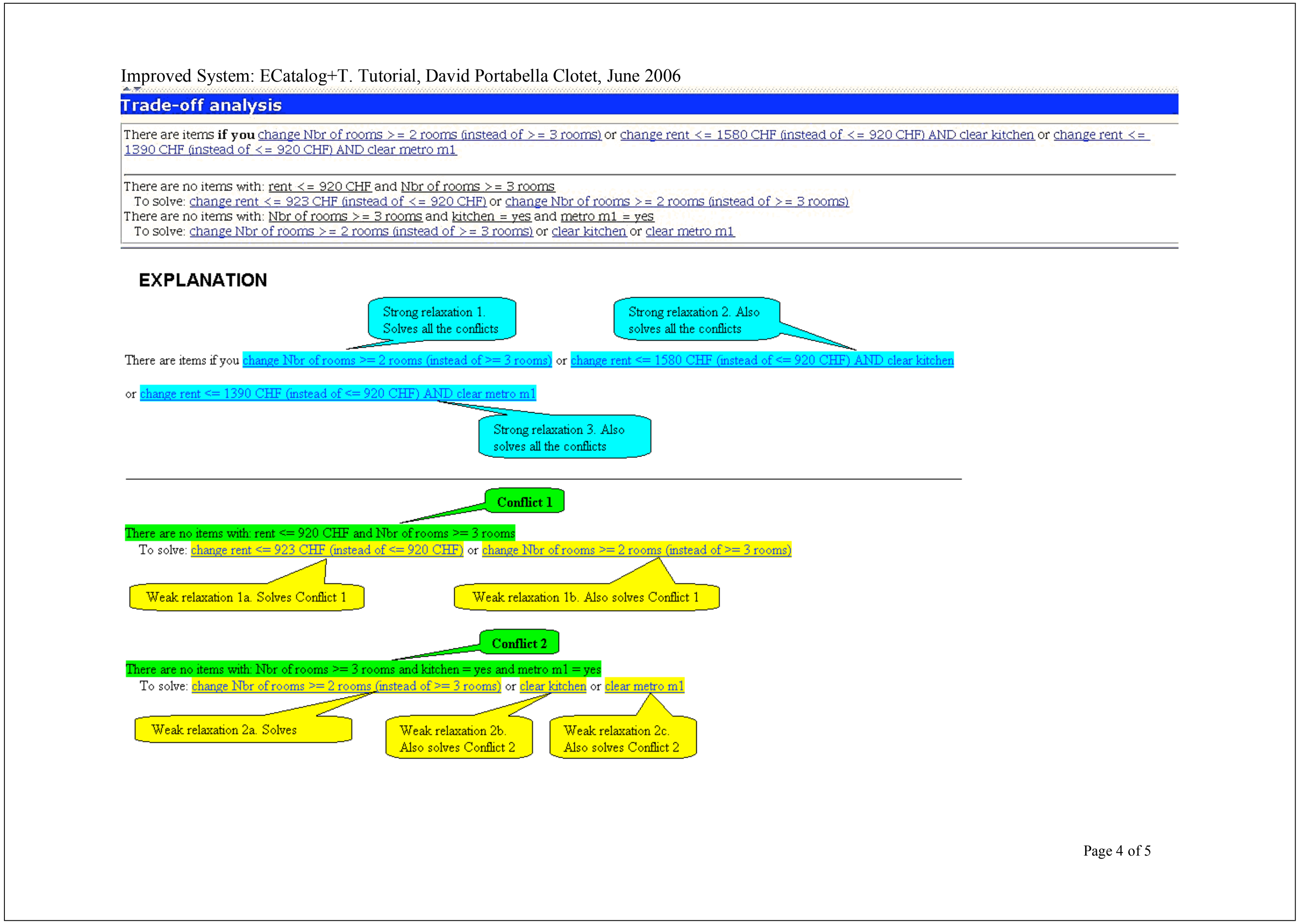
The relaxations on the first line are called “Strong Relaxations”. If you click one of them, you are sure that there will be matching apartments.

The relaxations below a conflict are called “Weak Relaxations”. If you click one of them, you know that you are going to solve the conflict above, but this does not guaranties that there will be matching apartments.

In the following page goes an image to illustrate this tutorial.

Page 3 of 5

User Study. Ecatalog+T Tutorial. Page 4 / 5.



Improved System: ECatalog+T. Tutorial, David Portabella Clotet, June 2006

Trade-off analysis

There are items **if you** change Nbr of rooms > = 2 rooms (instead of > = 3 rooms) or change rent < = 1580 CHF (instead of < = 920 CHF) AND clear kitchen or change rent < = 1390 CHF (instead of < = 920 CHF) AND clear metro m1

There are no items with: rent < = 920 CHF and Nbr of rooms > = 3 rooms
To solve: change rent < = 923 CHF (instead of < = 920 CHF) or change Nbr of rooms > = 2 rooms (instead of > = 3 rooms)
There are no items with: Nbr of rooms > = 3 rooms and kitchen = yes and metro m1 = yes
To solve: change Nbr of rooms > = 2 rooms (instead of > = 3 rooms) or clear kitchen or clear metro m1

EXPLANATION

There are items if you change Nbr of rooms >= 2 rooms (instead of >= 3 rooms) or change rent <= 1580 CHF (instead of <= 920 CHF) AND clear kitchen

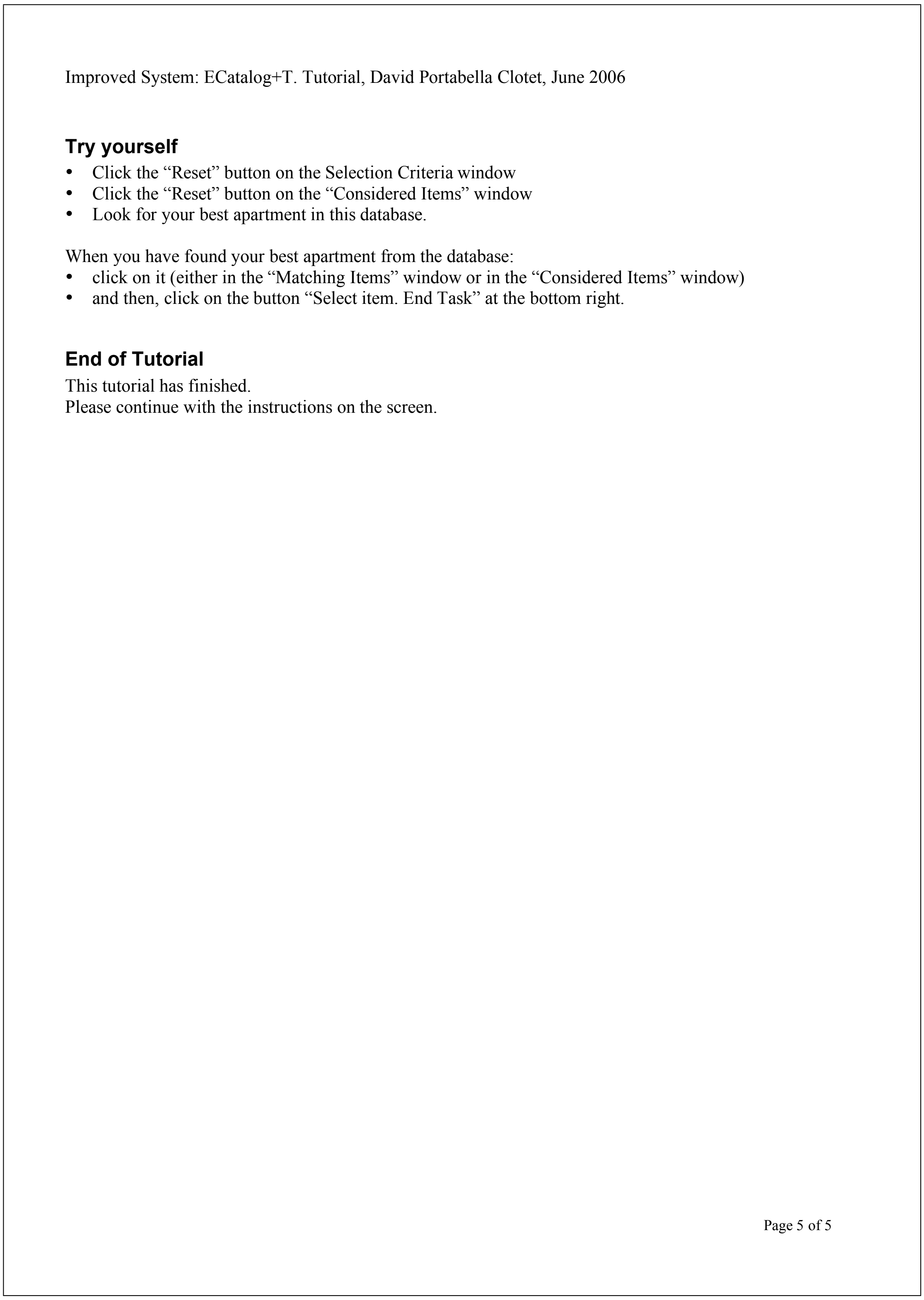
or change rent <= 1390 CHF (instead of <= 920 CHF) AND clear metro m1

There are no items with: rent <= 920 CHF and Nbr of rooms >= 3 rooms
To solve: change rent <= 923 CHF (instead of <= 920 CHF) or change Nbr of rooms >= 2 rooms (instead of >= 3 rooms)

There are no items with: Nbr of rooms >= 3 rooms and kitchen = yes and metro m1 = yes
To solve: change Nbr of rooms >= 2 rooms (instead of >= 3 rooms) or clear kitchen or clear metro m1

Strong relaxation 1. Solves all the conflicts
Strong relaxation 2. Also solves all the conflicts
Strong relaxation 3. Also solves all the conflicts
Conflict 1
Weak relaxation 1a. Solves Conflict 1
Weak relaxation 1b. Also solves Conflict 1
Conflict 2
Weak relaxation 2a. Solves
Weak relaxation 2b. Also solves Conflict 2
Weak relaxation 2c. Also solves Conflict 2
Page 4 of 5

User Study. Ecatalog+T Tutorial. Page 5 / 5.



Improved System: ECatalog+T. Tutorial, David Portabella Clotet, June 2006

Try yourself

- Click the “Reset” button on the Selection Criteria window
- Click the “Reset” button on the “Considered Items” window
- Look for your best apartment in this database.

When you have found your best apartment from the database:

- click on it (either in the “Matching Items” window or in the “Considered Items” window)
- and then, click on the button “Select item. End Task” at the bottom right.

End of Tutorial

This tutorial has finished.
Please continue with the instructions on the screen.

Page 5 of 5

Characteristic	**CBTE**	**PWADD**	**CBTE + PWADD**
Gender			
Male	12	11	10
Female	1	2	3
Age range			
18 – 24	1	4	2
25 – 35	8	8	11
36 – 45	4	1	0
Computer use			
Less than 1 hour /day	0	0	1
1 – 4 hours /day	0	2	1
5 – 7 hours / day	0	4	1
7 – 10 hours /day	11	7	7
More than 10 hours /day	2	0	3
Internet use			
Every day	13	10	12
A few times a week	0	3	1
Use Internet to compare products/services			
Yes	12	12	10
No	1	1	3
Computer Abilities			
Expert	11	10	10
Typical user	1	3	3
Rarely use computers	1	0	0
Know SQL			
Yes	9	7	9
No	4	6	4

Table A.5: Demographic characteristics of experiment participants by group.

	Participants												
	P01	**P02**	**P03**	**P04**	**P05**	**P06**	**P07**	**P08**	**P09**	**P10**	**P11**	**P12**	**P13**
Q1	3	5	5	5	6	5	3	6	7	6	6	4	5
Q2	3	7	4	3	5	6	6	7	7	7	6	5	6
Q3	3	6	5	4	5	6	7	7	7	6	6	6	6
Q4	3	7	1	5	6	1	4	7	1	7	6	5	6
Q5	6	2	3	5	1	6	2	1	1	5	1	2	3
Q6	4	2	2	6	1	3	2	2	1	1	1	2	1
Q7	5	2	4	5	5	6	2	2	1	2	1	3	3
Q8	3	2	2	3	2	6	4	7	7	7	6	5	3
Q9	3	2	2	2	1	2	2	2	1	1	1	2	1
Q10	3	6	4	6	3	5	6	6	7	6	6	6	5
Q11	3	5	1	3	4	4	4	4	1	5	1	1	1
Q12	4	5	4	3	3	4	6	6	2	5	5	2	4
Q13	3	2	4	4	2	2	4	3	1	3	1	4	2
Q14	3	6	4	6	4	2	5	5	4	6	6	3	2
Q15	3	4	5	5	7	2	6	7	7	7	7	6	2
Q16	3	5	4	4	6	2	6	4	7	6	7	2	2
Q17	3	7	5	4	6	2	4	7	5	4	6	3	6
Q18	3	6	5	6	4	3	5	7	7	7	6	5	6
Q19	3	6	4	6	4	4	3	5	7	6	6	6	5
Q20	3	6	5	5	6	6	5	7	7	7	6	5	5
Q21	3	6	5	4	5	3	6	7	7	5	6	6	4
Q22	2	7	4	6	7	2	6	4	6	7	6	4	7
Q23	2	7	3	6	6	2	6	7	2	5	7	3	6
Q24	3	6	6	5	4	2	6	7	7	7	6	3	6
Q25	3	6	4	4	4	2	3	5	3	5	6	3	5
Q26	3	6	6	4	6	6	5	7	7	7	7	5	5
Q27	3	7	5	3	7	6	6	7	6	6	7	4	6
Q28	3	7	5	4	6	4	6	7	6	7	7	5	6
Q29	6	7	4	7	7	5	6	7	6	6	7	5	6
Q30	3	7	2	6	6	3	6	7	5	5	5	4	5
Q31	2	4	3	4	7	6	4	7	2	7	7	5	4
Q32	2	4	3	4	5	2	4	7	4	6	7	4	4
Q33	5	7	5	5	7	2	4	6	7	6	7	5	6
Q34	3	7	4	6	6	5	6	7	6	7	7	5	6
Q35	6	7	7	6	7	2	7	7	7	7	4	6	4
Q36	6	7	5	6	6	2	7	7	7	6	4	6	2
Q37	6	7	5	6	7	2	7	7	6	7	4	6	5
Q38	6	7	5	6	7	2	5	7	7	7	4	5	7

Table A.6: Responses from the user study. Group 1 (CBTE). Participants P1 – P13.Questions Q1 – Q38 are given in Table 22 and Table 23.Measures on a seven-point Likert strongly disagree/strongly agree scale. See Section 4.4.

	Participants												
	P14	**P15**	**P16**	**P17**	**P18**	**P19**	**P20**	**P21**	**P22**	**P23**	**P24**	**P25**	**P26**
Q1	6	7	6	7	5	6	3	3	6	3	5	5	3
Q2	4	7	6	7	5	6	4	4	5	3	7	2	3
Q3	4	7	6	7	6	5	4	3	5	4	7	5	3
Q4	4	7	6	7	5	6	1	4	6	1	7	1	4
Q5	1	1	2	2	2	2	2	3	5	4	4	2	6
Q6	4	1	2	1	2	1	2	3	2	3	3	5	4
Q7	1	1	2	2	5	1	4	5	1	3	5	1	6
Q8	4	7	4	4	6	2	4	4	7	6	3	7	3
Q9	3	2	2	3	5	1	4	3	1	2	2	1	4
Q10	5	4	7	4	5	6	2	2	5	5	5	3	2
Q11	2	3	1	3	3	2	1	2	2	7	2	5	4
Q12	4	5	7	4	3	3	2	2	5	2	5	3	2
Q13	4	1	2	5	3	2	3	4	5	4	2	4	4
Q14	6	7	4	3	5	5	1	2	4	3	4	3	3
Q15	4	4	4	4	4	6	2	2	3	3	6	3	4
Q16	5	5	4	3	4	6	2	3	5	4	5	3	4
Q17	5	5	7	5	5	6	3	2	4	5	5	3	3
Q18	5	5	6	7	4	6	2	4	2	5	6	3	3
Q19	4	5	6	6	4	4	2	3	2	5	6	5	3
Q20	5	7	7	6	5	5	2	4	4	5	6	3	3
Q21	5	7	5	6	5	5	2	3	3	4	6	3	3
Q22	5	4	4	7	5	6	2	4	3	5	6	3	2
Q23	5	5	6	7	6	5	2	4	4	5	6	5	6
Q24	6	5	6	6	5	6	2	4	3	5	6	4	4
Q25	6	5	7	5	5	5	3	4	2	5	5	4	4
Q26	5	7	7	6	6	6	2	4	3	5	6	3	4
Q27	5	7	7	3	4	7	1	3	2	5	6	3	4
Q28	5	7	7	4	5	6	1	3	2	5	6	3	4
Q29	5	7	6	6	6	6	2	3	6	7	6	5	5
Q30	4	7	6	5	5	6	1	3	6	6	6	4	2
Q31	4	4	3	5	5	2	2	4	5	5	5	3	4
Q32	4	4	3	4	5	5	2	4	5	5	5	3	4
Q33	5	7	4	4	5	5	2	4	5	5	6	3	2
Q34	5	7	3	5	6	6	3	4	6	7	6	3	2
Q35	5	5	6	4	6	7	6	6	2	2	6	6	5
Q36	4	4	6	3	6	7	6	5	3	1	5	5	4
Q37	4	6	6	7	6	7	6	5	3	2	6	6	4
Q38	6	7	6	6	6	6	2	4	2	4	6	4	4

Table A.7: Responses from the user study. Group 2 (PWADD). Participants P14 – P26.Questions Q1 – Q38 are given in Table A.3 and Table A.4.Measures on a seven-point Likert strongly disagree/strongly agree scale. See Section 4.4.

	Participants												
	P27	**P28**	**P29**	**P30**	**P31**	**P32**	**P33**	**P34**	**P35**	**P36**	**P37**	**P38**	**P39**
Q1	6	3	5	7	6	6	3	6	6	4	5	7	4
Q2	6	2	5	7	6	4	3	7	6	7	6	6	6
Q3	6	2	6	6	5	4	2	7	6	6	6	6	6
Q4	1	3	5	5	7	4	3	7	6	6	6	6	6
Q5	4	2	1	1	1	1	5	1	3	4	2	1	1
Q6	1	2	1	1	2	2	6	1	2	2	2	1	1
Q7	1	2	1	1	2	4	6	1	6	1	2	1	4
Q8	4	2	4	5	6	4	3	4	5	7	3	7	4
Q9	1	6	2	1	2	2	4	1	3	3	2	1	4
Q10	7	5	5	7	6	5	2	5	6	7	5	7	4
Q11	5	1	1	1	4	2	7	1	2	1	6	1	1
Q12	3	2	6	6	6	5	2	5	4	6	2	6	5
Q13	1	1	1	1	3	3	6	2	4	4	6	1	2
Q14	4	6	5	7	4	6	3	6	2	3	2	6	4
Q15	6	6	5	7	6	5	3	7	6	6	7	7	2
Q16	6	5	5	6	6	4	2	6	5	2	6	7	6
Q17	5	4	6	7	7	3	2	7	6	4	6	7	6
Q18	4	6	6	6	7	3	4	7	6	4	4	6	6
Q19	6	6	5	6	5	3	3	7	6	7	4	7	6
Q20	6	6	5	7	7	3	2	7	6	7	6	7	6
Q21	6	6	6	7	5	3	2	7	6	5	6	6	6
Q22	6	5	5	7	6	5	3	7	5	7	6	7	6
Q23	4	5	5	6	6	3	2	6	6	7	4	7	7
Q24	4	6	5	6	5	4	3	7	6	6	5	7	6
Q25	6	6	6	6	5	3	3	7	6	6	4	7	6
Q26	6	6	5	7	7	3	3	7	6	6	5	7	6
Q27	5	6	5	7	6	2	3	7	6	7	6	7	6
Q28	5	6	5	7	6	2	3	7	4	6	5	7	6
Q29	5	6	6	6	6	4	3	6	6	6	6	7	6
Q30	4	5	6	6	5	4	3	7	4	6	4	7	4
Q31	5	6	4	6	4	4	4	6	5	7	6	6	4
Q32	4	6	4	7	4	4	4	6	5	7	6	6	4
Q33	5	7	6	7	6	4	4	7	5	4	6	7	4
Q34	6	7	6	5	6	5	5	7	5	7	7	6	6
Q35	6	5	4	7	7	7	1	7	6	7	7	6	2
Q36	6	5	4	7	7	7	2	6	4	7	6	6	2
Q37	6	5	4	6	6	7	2	7	6	6	5	7	5
Q38	6	5	5	7	7	3	2	7	6	7	6	6	7

Table A.8: Responses from the user study. Group 2 (CBTE+PWADD). Participants P27-P39.Questions Q1 – Q38 are given in Table A.3 and Table A.4.Measures on a seven-point Likert strongly disagree/strongly agree scale. See Section 4.4.

Appendix B

Software and data

Software and data available
Researchers are expected to provide a complete documentation for their empirical studies, including data and methodology, in order that other researchers have the opportunity to reproduce the studies and verify the results. This also allows making meta-analysis, i.e. to combine the results of several studies that address a set of related research hypotheses, especially useful when studies have contradictory results. Moreover, user studies are expensive to carry out, and sharing interaction logs can typically allow other researchers to study hypotheses not anticipated by in the original studies.

For this purpose, we released, in addition to the information provided in this thesis, the complete data and software necessary to easily reproduce our results, as well as the obtained interaction logs.
All the data and software is available at http://ecatalog.sourceforge.net/.

Comparis8000 database
In order to get a more real scenario, we have contacted a company[1] that aggregates the inventory of used cars from several concessionaries in Switzerland. They have provided us with a catalog of 61443 cars, from which 7924 are described with the following 35 attributes: brand, model, model type, current price, original price, date of the 1^{st} matriculation, mileage, category, fuel type, transmission, number of seas, number of doors, performance in hp, cubic capacity, number of cylinders, color and the Boolean features metallic color, abs, airbag, air conditioning, alarm, cd player, central locking, electrical seats, electrical windows, fog lamps, heating seats, leather interior, navigation system, power steering, speed control, sport chassis, sun roof, theft prevention, traction control and xenon lights.

With the permission of Comparis.ch, we have compiled and released a subset of this

[1]We would like to acknowledge Comparis.ch for sharing this database to the research community

database under the name of “Comparis8000”. “Comparis8000” is a database of 7924 used-cars modeled by 35 attributes.
It is available to download at http://ecatalog.sourceforge.net/ for the sake of research only[2].

Comparis200 database
A subset of the “Comparis8000” database more suitable for user studies is released under the name of “Comparis200”. It contains 200 used cars modeled with the same 35 attributes.
It is available to download at http://ecatalog.sourceforge.net/ for the sake of research only.

[2]If you use this database, Comparis.ch would appreciate if you send a short mail to them, Johann Burkhard <johann.burkhard@comparis.ch>, stating your email and a short description of its use.

Bibliography

[1] J. Von Neumann & O. Morgenstern. Theory of Games and Economic Behavior. Princeton University Press, Princeton, NJ, 1947.

[2] Fishburn. Utility Theory for Decision Making. Wiley, Ney York, 1970.

[3] A. Tversky. *Elimination by aspects: A theory of choice.* Psychological Review, vol. 79, no. 4, pages 281–299, 1972.

[4] L. Ross, MR Lepper & M. Hubbard. *Perseverance in self-perception and social perception: biased attributional processes in the debriefing paradigm.* J Pers Soc Psychol, vol. 32, no. 5, pages 880–92, 1975.

[5] Ralph L. Keeney & Howard Raiffa. Decisions with multiple objectives : preferences and value tradeoffs. Wiley, Ney York, 1976.

[6] W. Thorngate. *Efficient decision heuristics.* Behavioral Science, vol. 25, no. 3, 1980.

[7] Simonson, Itamar & Tversky, Amos. *Context: Tradeoff Contrast and Extremeness Aversion.* Journal of Marketing Research, vol. 29, no. 3, pages 281–295, aug 1992.

[8] John W. Payne, James R. Bettman & Eric J. Johnson. The adaptive decision maker. Cambridge University Press, New York etc., 1993. John W. Payne, James R. Bettman, Eric J. Johnson Ill.

[9] R.T. Clemen. Making hard decisions: an introduction to decision analysis. Duxbury Press, 1996.

[10] Jiyong Zhang & Pearl Pu. *Survey of Solving Multi-Attribute Decision Problems.* Technical Report No. IC/2004/54 200454, Swiss Federal Institute of Technology (EPFL), Lausanne (Switzerland), 2004.

[11] Darius Braziunas. *Computational Approaches to Preference Elicitation.* Rapport technique, Department of Computer Science, University of Toronto, 2006.

[12] Boi Faltings, Marc Torrens & Pearl Pu. *Solution Generation with Qualitative Models of Preferences.* Computational Intelligence, vol. 20, no. 2, pages 246–263, May 2004.

[13] J.A. Rodríguez-Aguilar, A. Reyes-Moro, M. López-Sánchez & Jesús Cerquides. *Enabling assisted strategic negotiations in actual-world procurement scenarios.* Electronic Commerce Research, 2005.

[14] R. Bayer & E.M. McCreight. *Organization and Maintenance of Large Ordered Indices.* Acta Informatica, vol. 1, no. 3, pages 173–189, 1972.

[15] Christos Papadimitriou. The theory of database concurrency control. Computer Science Press, Inc., New York, NY, USA, 1986.

[16] W.J. Frawley, G. Piatetsky-Shapiro & C.J. Matheus. *Knowledge Discovery in Databases: An Overview.* AI Magazine, vol. 13, no. 3, pages 57–70, 1992.

[17] C. J. Date. An introduction to database systems, 6th edition. Addison-Wesley, 1995.

[18] Parke Godfrey. *Minimization in Cooperative Response to Failing Database Queries.* International Journal of Cooperative Information Systems, vol. 6, no. 2, pages 95–149, 1997.

[19] J.R. Bitner & E.M. Reingold. *Backtrack programming techniques.* Communications of the ACM, vol. 18, no. 11, pages 651–656, 1975.

[20] E. Tsang. Foundations of constraint satisfaction. Academic Press San Diego, 1993.

[21] R.H. Guttman, A.G. Moukas & P. Maes. *Agent-mediated electronic commerce: a survey.* The Knowledge Engineering Review, vol. 13, no. 02, pages 147–159, 2001.

[22] Ronald Fagin. *Fuzzy queries in multimedia database systems.* In PODS '98: Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, pages 1–10, New York, NY, USA, 1998. ACM Press.

[23] Benjamin Han & Shie-Jue Lee. *Deriving minimal conflict sets by CS-trees with mark set in diagnosis from first principles.* IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 29, no. 2, pages 281–286, 1999.

[24] Marc Torrens & Boi Faltings. *SmartClients: Constraint satisfaction as a paradigm for scaleable intelligent information systems.* In AAAI Workshop on Artificial Intelligence for Electronic Commerce, volume Technical Report WS-99-01, pages 10–15. AAAI, AAAI Press, July 1999.

[25] U. Junker. *QUICKXPLAIN: Conflict detection for arbitrary constraint propagation algorithms.* IJCAI'01 Workshop on Modelling and Solving problems with constraints, 2001.

[26] Jérome Amilhastre, Hélène Fargier & Pierre Marquis. *Consistency Restoration and Explanations in Dynamic CSPs—Application to Configuration.* Artificial Intelligence, vol. 135, no. 1–2, pages 199–234, 2002.

[27] P.G.W. Keen & M.S. Scott-Morton. *Decision support systems: an organizational perspective.* Addison Wesley, Reading, 1978.

[28] S. J. Kaplan. *Cooperative Responses from a Portable Natural Language Query System.* Artificial Intelligence, vol. 2, no. 19, 1982.

[29] Michael David Williams. *What Makes RABBIT Run?* International Journal of Man-Machine Studies, vol. 21, no. 4, pages 333–352, 1984.

[30] Amihai Motro. *FLEX: A Tolerant and Cooperative User Interface to Databases.* IEEE Transactions on Knowledge and Data Engineering, vol. 2, no. 2, pages 231–246, June 1990.

[31] Todd, Peter & Benbasat, Izak. *The Use of Information in Decision Making: An Experimental Investigation of the Impact of Computer-Based Decision Aids.* MIS Quarterly, vol. 16, no. 3, pages 373–393, sep 1992.

[32] Robin D. Burke, Kristian J. Hammond & Benjamin C. Young. *Knowledge-Based Navigation of Complex Information Spaces.* In AAAI/IAAI, Vol. 1, pages 462–468, 1996.

[33] G. Linden, S. Hanks & N. Lesh. *Interactive assessment of user preference models: The automated travel assistant.* Proceedings of the Sixth International Conference on User Modeling, vol. 6778, 1997.

[34] P. Pu & B. Faltings. *Enriching buyers' experiences: the SmartClient approach.* Proceedings of the SIGCHI conference on Human factors in computing systems, pages 289–296, 2000.

[35] G. Haeubl & V. Trifts. *Consumer Decision Making in Online Shopping Environments: The Effects of Interactive Decision Aids.* Marketing Science, vol. 19, no. 1, pages 4–21, 2000.

[36] Callahan, E. and Koenemann, J. A comparative usability evaluation of user interfaces for online product catalogs. In Proceedings of ACM Conference on Electronic Commerce (EC'00), 2000, 197-206.

[37] H. Shimazu. *ExpertClerk: Navigating Shoppers' Buying Process with the Combination of Asking and Proposing.* In Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI'01), Seattle, Washington, USA, 2001.

[38] Ulrich Junker. *Preference Programming for Configuration.* In Proceedings of 4th Workshop on Configuration (IJCAI'01), pages 50–56, Seattle, May 12 2001.

[39] Barry Smyth & Paul McClave. *Similarity vs. Diversity.* In ICCBR '01: Proceedings of the 4th International Conference on Case-Based Reasoning, pages 347–361, London, UK, 2001. Springer-Verlag.

[40] Daniel Felix, Christoph Niederberger, Patrick Steiger & Markus Stolze. *Feature-Oriented vs. Needs-Oriented Product Access for Non-Expert-Online Shoppers.* In I3E '01: Proceedings of the IFIP Conference on Towards The E-Society, pages 399–406, Deventer, The Netherlands, The Netherlands, 2001. Kluwer, B.V.

[41] J. Jedetski, L. Adelman & C. Yeo. *How Web site decision technology affects consumers.* Internet Computing, IEEE, vol. 6, no. 2, pages 72–79, 2002.

[42] Marc Torrens, Boi Faltings & Pearl Pu. *Smart Clients: Constraint Satisfaction as a Paradigm for Scaleable Intelligent Information Systems.* Special issue on Constraints and Agents. CONSTRAINTS: an Internation Journal. Kluwer Academic Publishers, no. 7, pages 49–69, 2002.

[43] P. Pu & B. Faltings. *Personalized Navigation of Heterogeneous Product Spaces using SmartClient.* In International Conference on Intelligent User Interfaces. ACM Press, 2002.

[44] David McSherry. *Diversity-Conscious Retrieval.* In ECCBR '02: Proceedings of the 6th European Conference on Advances in Case-Based Reasoning, pages 219–233, London, UK, 2002. Springer-Verlag.

[45] Robin D. Burke. *Interactive Critiquing for Catalog Navigation in E-Commerce.* Artif. Intell. Rev, vol. 18, no. 3-4, pages 245–267, 2002.

[46] Pearl Pu, Boi Faltings & Marc Torrens. *User-Involved Preference Elicitation.* In Working Notes of the Workshop on Configuration. Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-2003), Acapulco, Mexico, August 2003.

[47] Lorraine McGinty & Barry Smyth. *On the Role of Diversity in Conversational Recommender Systems.* In Kevin D. Ashley & Derek G. Bridge, editeurs, Case-Based Reasoning Research and Development, 5th International Conference on Case-Based Reasoning, ICCBR 2003, Trondheim, Norway, June 23-26, 2003, Proceedings, volume 2689 of *Lecture Notes in Computer Science*, pages 276–290. Springer, 2003.

[48] Kevin McCarthy, James Reilly, Lorraine McGinty & Barry Smyth. *On the Dynamic Generation of Compound Critiques in Conversational Recommender Systems.* In Paul De Bra & Wolfgang Nejdl, editeurs, Adaptive Hypermedia and Adaptive Web-Based Systems, Third International Conference, AH 2004, Eindhoven, The Netherlands, August 23-26, 2004, Proceedings, volume 3137 of *Lecture Notes in Computer Science*, pages 176–184. Springer, 2004.

[49] James Reilly, Kevin McCarthy, Lorraine McGinty & Barry Smyth. *Incremental critiquing.* In Proceedings of The 24th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence. (AI-04), Cambridge, UK., 2004.

[50] David McSherry. *Similarity and Compromise.* In Kevin D. Ashley & Derek G. Bridge, editeurs, Case-Based Reasoning Research and Development, 5th International Conference on Case-Based Reasoning, ICCBR 2003, Trondheim, Norway, June 23-26, 2003, Proceedings, volume 2689 of *Lecture Notes in Computer Science*, pages 291–305. Springer, 2003.

[51] Boi Faltings, Pearl Pu, Marc Torrens & Paolo Viappiani. *Designing Example-critiquing Interaction.* In International Conference on Intelligent User Interfaces, pages 22–29, Island of Madeira (Portugal), January 2004. ACM.

[52] James Reilly, Kevin McCarthy, Lorraine McGinty & Barry Smyth. *Explaining Compound Critiquing.* In Proceedings of the 9th UK Workshop on Case-Based Reasoning (UKCBR-04), pages 12–20, Cambridge, UK, 2004.

[53] Pearl Huan Z. Pu & Pratyush Kumar. *Evaluating example-based search tools.* In EC '04: Proceedings of the 5th ACM conference on Electronic commerce, pages 208–217, New York, NY, USA, 2004. ACM Press.

[54] Pearl Pu, Boi Faltings & Marc Torrens. *Effective Interaction Principles for Online Product Search Environments.* In WI '04: Proceedings of the Web Intelligence, IEEE/WIC/ACM International Conference on (WI'04), pages 724–727, Washington, DC, USA, 2004. IEEE Computer Society.

[55] Ion Muslea & Thomas J. Lee. *Online Query Relaxation via Bayesian Causal Structures Discovery.* In Manuela M. Veloso & Subbarao Kambhampati, editeurs, Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA, pages 831–836. AAAI Press / The MIT Press, 2005.

[56] Pearl Pu & Li Chen. *Integrating tradeoff support in product search tools for e-commerce sites.* In EC '05: Proceedings of the 6th ACM conference on Electronic commerce, pages 269–278, New York, NY, USA, 2005. ACM Press.

[57] Kevin McCarthy, James Reilly, Lorraine McGinty & Barry Smyth. *An Analysis of Critique Diversity in Case-Based Recommendation.* In Ingrid Russell & Zdravko Markov, editeurs, Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS), Clearwater Beach, Florida, USA, pages 123–128. AAAI Press, 2005.

[58] Paolo Viappiani, Boi Faltings, Vincent Schickel-Zuber & Pearl Pu. *Stimulating Preference Expression Using Suggestions.* In IJCAI-05 Multidisciplinary Workshop on Advances in Preference, pages 186–191, Edinburgh, UK, August 2005.

[59] Gediminas Adomavicius & Alexander Tuzhilin. *Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions.* IEEE Trans. Knowl. Data Eng, vol. 17, no. 6, pages 734–749, 2005.

[60] Li Chen and Pearl Pu. Trust Building in Recommender Agents. Workshop of 2nd International Conference on E-Business and Telecommunication Networks (ICETE'05), pages 135-145, Reading, UK, October 2005.

[61] Paolo Viappiani & Boi Faltings. *Design and Implementation of Preference-based Search.* In Springer, editeur, The 7th International Conference on Web Information Systems Engineering, LNCS4255, Wuhan, China, October 2006.

[62] Paolo Viappiani, Boi Faltings & Pearl Pu. *Evaluating Preference-based Search Tools: a Tale of Two Approaches.* In Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06), pages 205–211, Boston USA, July 2006. AAAI press.

[63] Li Chen & Pearl Pu. *Evaluating Critiquing-based Recommender Agents.* In Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06), pages 157–162, Boston USA, July 2006. AAAI press.

[64] Pearl Pu & Li Chen. *Trust building with explanation interfaces.* In IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces, pages 93–100, New York, NY, USA, 2006. ACM Press.

[65] B. Shneiderman. *Tree visualization with tree-maps: 2-d space-filling approach.* ACM Transactions on Graphics (TOG), vol. 11, no. 1, pages 92–99, 1992.

[66] B. Shneiderman. *Dynamic queries for visual information seeking.* IEEE Software, vol. 11, no. 6, pages 70–77, 1994.

[67] Christopher Ahlberg & Ben Shneiderman. *Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays.* In Human Factors in Computing Systems. Conference Proceedings CHI'94, pages 313–317, 1994.

[68] E.H. Chi, P. Barry, J. Riedl & J. Konstan. *A spreadsheet approach to information visualization.* Proceedings of Information Visualization Symposium, vol. 97, pages 17–24, 1997.

[69] S.K. Card, J. Mackinlay & B. Shneiderman. Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann, 1999.

[70] P.C. Wason. *On the failure to eliminate hypotheses in a conceptual task.* Quarterly Journal of Experimental Psychology, vol. 12, pages 129–140, 1960.

[71] A. Koriat, S. Lichtenstein & B. Fischhoff. *Reasons for confidence.* Journal of Experimental Psychology: Human Learning and Memory, vol. 6, pages 107–118, 1980.

[72] Fred D. Davis. *Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology.* MIS Quarterly, vol. 13, no. 3, pages 319–340, sep 1989.

[73] SG Hart & C. Wickens. *Workload assessment and prediction.* MANPRINT, an approach to systems integration, pages 257–296, 1990.

[74] J. Klayman. *Varieties of confirmation bias.* Psychology of Learning and Motivation, vol. 32, pages 385–418, 1995.

[75] Geyskens I., Steenkamp J.-B.E.M., Scheer L.K. & Kumar N. *The effects of trust and interdependence on relationship commitment: A trans-Atlantic study.* International Journal of Research in Marketing, vol. 13, pages 303–317(15), October 1996.

[76] Mark Bovens & Paul Hart. Understanding policy fiascoes. Transaction, New Brunswick, 1996.

[77] G. Dosi & D. Lovallo. *Rational entrepreneurs or optimistic martyrs? Some considerations on technological regimes, corporate entries, and the evolutionary role of decision biases.* In In Technological Innovation: Oversights and Foresights, eds. R. Garud, P. Nayyar, and Z. Shapira, 41, pages 41–68. Cambridge University Press, 1997.

[78] Doney, Patricia M. & Cannon, Joseph P. *An Examination of the Nature of Trust in Buyer-Seller Relationships.* Journal of Marketing, vol. 61, no. 2, pages 35–51, apr 1997.

[79] Policy Considerations for Electronic Commerce. International Telecommunication Union, discussion paper, World Telecommunication Day, 1999.

[80] Dan Ariely. *Controlling the Information Flow: Effects on Consumers' Decision Making and Preferences.* The Journal of Consumer Research, vol. 27, no. 2, pages 233–248, sep 2000.

[81] J.L. Herlocker, J.A. Konstan & J. Riedl. *Explaining collaborative filtering recommendations.* Proceedings of the 2000 ACM conference on Computer supported cooperative work, pages 241–250, 2000.

[82] Baldwin L. P. & Currie W. L. *Key Issues in Electronic Commerce in Today's Global Information Infrastructure.* Cognition, Technology & Work, vol. 2, no. 1, pages 27–34, 2000.

[83] D. Goersch. *Internet Limitations, Product Types, and the Future of Electronic Retailing.* In Proceeding of the 1st Nordic Workshop on Electronic Commerce, 2001.

[84] K. Head M.and Hassanein. *Trust in e-Commerce: Evaluating the Impact of Third-Party Seals.* Quarterly Journal of Electronic Commerce (QJEC), vol. 3, no. 3, pages 307–325, 2002.

[85] D.H. McKnight. *What Trust Means in E-Commerce Customer Relationships: An Interdisciplinary Conceptual Typology.* International Journal of Electronic Commerce, vol. 6, no. 2, pages 35–59, 2001.

[86] D. Gefen, E. Karahanna & D.W. Straub. *Trust and TAM in Online Shopping: An Integrated Model.* MIS Quarterly, vol. 27, no. 1, pages 51–90, 2003.

[87] Viswanath Venkatesh, Michael G. Morris, Gordon B. Davis & Fred D. Davis. *User Acceptance of Information Technology: Toward a Unified View.* MIS Quarterly, vol. 27, no. 3, 2003.

[88] Milena M. Head & Khaled Hassanein. *Building Online Trust through Socially Rich Web Interfaces.* In In Proceedings of the Second Annual Conference on Privacy, Security, and Trust (PST'2004), pages 15–22, Fredericton, Canada, October 2004.

[89] Arnold A. Kamis & Edward A. Stohr. *Parametric search engines: what makes them effective when shopping online for differentiated products?* Inf. Manage., vol. 43, no. 7, pages 904–918, 2006.

[90] M. Parkin. Economics. Addison-Wesley, Boston, Massachusetts, 2006.

[91] J. F. Hair, R. E. Anderson, R. L. Tatham & W. C. Black. Multivariate data analysis with readings. Prentice-Hall International, 1995.

[92] J.A. Jacko & A. Sears. The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications. Lawrence Erlbaum Assoc Inc, 2003.

[93] C. Silverstein, M. Henzinger, H. Marais & M. Moricz. *Analysis of a very large AltaVista query log.* SRC Technical Note, vol. 14, page 1998, 1998.

[94] J. C. Westland & T. H. K. Clark. Global Electronic Commerce: Theory and Case Studies. MIT Press,Cambridge, MA, 2000.

[95] Ben Shneiderman & Catherine Plaisant. Designing the user interface: Strategies for effective human-computer interaction (4th edition). Pearson Addison Wesley, 2004.

[96] Akamai. Retail web site performance: Consumer reaction to a poor online shopping experience. Jupiter Research, 2006.

Curriculum Vitae

David Portabella Clotet
Av. du Grey 14
1004 Lausanne
Switzerland

URL: http://lia.epfl.ch/˜portabel/
email: david.portabella@epfl.ch
Date of Birth: October 7, 1976
Nationality: Spanish

Education

- *PhD Candidate* (since November 2003)
  Artificial Intelligence Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.
- *Venture Challenge course* (Nov 2005 for 6 months)
  VentureLab, Switzerland
- *Graduate School in Computer Science* (2001-2002)
  School of Computer and Communication Sciences, EPFL
- *Master thesis* (Oct 2000 for 6 months)
  Logic Systems Laboratory, EPFL
- *B.S. and M.S. in Telecommunications Engineering* (1994 – 2000)
  Universititat Politècnica de Catalunya (UPC), Barcelona, Spain
  Honours in Speech Recognition, Programming and Computer Architecture & Operating Systems.

Professional Experience

- *Research and Teaching Assistant* (Since 2003)
  Artificial Intelligence Laboratory, EPFL
- *Internship* (Summer 2000)
  Anglatecnic Professional Broadcasting and IT, Barcelona, Spain
- *Internership* (1999 – 2000)
  TALP Research Center, Barcelona, Spain

- *Summer Internership* (Summer 1998)
  Pontificia Universidad Javeriana, Bogotá, Colombia

- *Network administrator* (1994–1999)
  Tot Micro, Manresa, Spain

Honors and Achievements

- *Fellowship* (2002)
  Department of Computer Science, EPFL Graduate School

- *Fellowship* (1999)
  Department of Signal Theory and Communications, UPC

- *Fellowship* (1998)
  Spanish Agency of International Cooperation, Ministerio de Asuntos Exteriores and Universidad Pontificia Javeriana, Bogotá, Colombia

- *CIRIT award* (1994)
  Generalitat de Catalunya, Consell Interdepartamental de Recerca i Innovació Tecnologica

Publications

1. D. Portabella & M. Rajman. *Explicit Trade-off and Prospective Analysis in Electronic Catalogs.* Workshop on Recommender Systems. European Conference on Artificial Intelligence. Riva del Garda, Italy, August 2006.

2. D. Portabella & M. Rajman. *Explicit Passive Analysis in Electronic Catalogs.* Proceedings of the National Conference on Artificial Intelligence, AAAI-06. Poster. Boston, July 2006.

3. D. Portabella & V. Pallotta & M. Rajman. *Systematic definition and assent to eContracts for Web Services.* CoAla 2005 Workshop on Contract Architectures and Languages, Enschede, The Netherlands, November 2005.

4. D. Portabella & M. Rajman. *A Dialogue-based Grounding mechanism and new Service Description Features for adapting Semantic (Web) Services to Personal Assistants.* Technical Report No. ID: IC/2004/85, Swiss Federal Institute of Technology (EPFL), Lausanne (Switzerland), November, 2004.

5. D. Portabella & M. Rajman & S. Cevey. *The IM2.MDM demonstrator: a dialogue based interface for the meeting database.* Poster. IM2 Summer Institute 2003, Centre de conférences du Régent, Crans-Montana, October 6-7-8.

6. D. Portabella & A. Febrer & A. Moreno. NaniTrans*: A Speech Labeling tool.* Proc. LREC'2000, Athens, 2000.