Digital Business Ecosystem

Contract n° 507953

WP 7: User Profiling

Del 7.2: Initial Description of Profiling mechanism design and rationale with respect to one or two use cases

Information Society Technologies

Project funded by the European Community under the "Information Society Technology" Programme

Contract Number: 507953
Project Acronym: DBE
Title: Digital Business Ecosystem

Deliverable N°: 7.2
Due date: 31/08/2005
Delivery Date: 12/10/2005

Short Description:
This paper introduces the User Profiling metamodel and describes the use of the User Profile having regard to two use cases (including a Graphical User Interface). Several mechanisms for implicit profiling activities are introduced and their potential contribution to User Profiling in the project is critically discussed.

Partners owning: FZI (Christian Bartsch)
Partners contributed: TUC, INTEL, ISUFI, T6
Made available to: DBE Partners and European Commission

VERSIONING		
VERSION	DATE	AUTHOR, ORGANISATION
0.1	**01/06/2005**	**CHRISTIAN BARTSCH (FZI)**
0.2	**05/08/2005**	**RIM HELAOIU (FZI) –PROFILING MECHANISM**
0.3	**07/10/2005**	**GEORGE KOTOPOLOUS (TUC) – 1ST REVIEW**
0.4	**07/10/2005**	**JOHN KENNEDY (INTEL) – 2ND REVIEW**
1.0	**12/10/2005**	**CHRISTIAN BARTSCH (FZI) – FINAL VERSION**

Quality check
1st Internal Reviewer: George Kotopolous (TUC)
2nd Internal Reviewer: John Kennedy (INTEL)

„The challenge in an information rich world is not only to make information available to people at any time, at any place, and in any form, but specifically to say the right thing at the right time in the right way…”

G. Fischer

TABLE OF CONTENTS

List Of Figures

LIST OF TABLES

1 INTRODUCTION

This Deliverable provides the basic Meta Object Facility (MOF) model for connecting the User Profile (UP) to the Knowledge Base (KB). The User Profile follows the design approach recommended by the DBE partner TUC[1] extending the OCL[2]-based Query Modelling Language (QML) with parameters and structures which are necessary for representing any kind of information relevant to User Profiling. Additionally a basic Graphical User Interface (GUI) design for a User Profile is introduced to illustrate the functionality provided by the User Profile Metamodel (UPM).

Although the extended BML model developed in Deliverable D7.1 can be used to handle BML information in a User Profile, this approach requires a complete model of each desired language to be represented in a profile every time. Thus a unique model would be necessary for both SDL and SSL and each model would need an adapted API to connect to the knowledge base. To avoid this scenario the abstract QML metamodel was extended with a User Profile model assuring flexibility in addressing any kind of modelling language in the future. Additionally, general discussion will put light on what our understanding of a profiling mechanism is, in this context. A short excursion will talk about aspects and potential risks regarding the quality of the recommendation process, and also the quality of the preference compositions using the current non-organised structure for available service models. Finally an overview about profiling mechanisms will show the potential, the risks and the feasibility of implicit profiling techniques in the project's context.

---

[1] Technical University of Crete (TUC)
[2] Object Constraint Language (OCL)

2 USER PROFILING MECHANISM

Before introducing the User Profile Metamodel (UPM) and the Graphical User Interface, which support the user in organising their preferences and representing them in a concise way, a short interpretation will clarify what is generally meant by a profiling 'mechanism'[3]. Two types of mechanisms can be considered:

- a simple algorithm or method for accomplishing a defined task (an algorithm-based mechanism) and
- a complete model consisting of many parts and processes (an assemblage-based mechanism).

To avoid possible misunderstandings this differentiation is needed to clarify on what the further focus of a User Profile will be on.

2.1 ALGORITHM-BASED MECHANISM

A possible interpretation of the term User Profiling Mechanism could be the concrete algorithm-based mechanism for fulfilling a predefined user behaviour-based task such as Log-File Analysis, or implicit ranking. A mechanism in this context is just a single part of the whole User Profile, and wouldn't have any functionality without an existing User Profile infrastructure. The interaction shown in Figure 1 represents the sources for retrieving information such as would be used by the Recommender or the Evolutionary Environment.

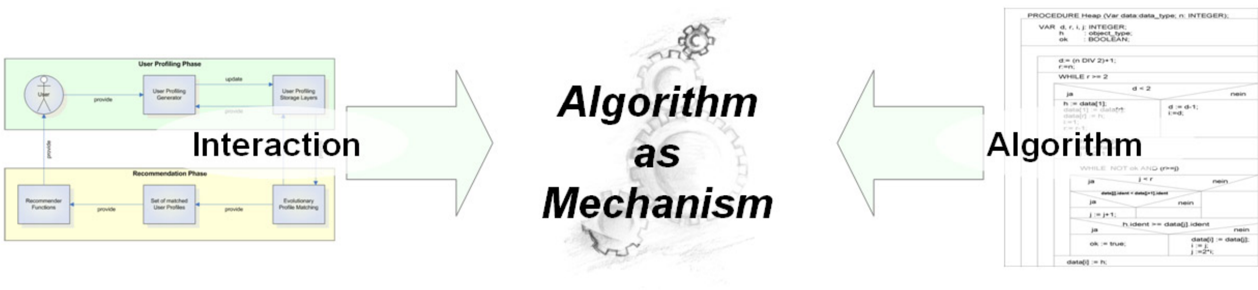


Figure 1: Profiling Mechanism: Algorithm Interpretation

The algorithm in this case is the mechanism that processes information and provides activities for the user profile. The interaction is also part of the mechanism as it shows the connection between a user profile and other components like the recommender. See also [Bart05b] for a more detailed description of the interaction cycle.

[3] The term was introduced in the work-plan (Addendum n°1) but didn't give a concrete definition of what is exactly meant by this.

2.2 ASSEMBLAGE-BASED MECHANISM

A User Profiling Mechanism can on the other hand be interpreted as an assemblage of components or parts contributing to the same objective: organising, retrieving and representing information as user preferences in a User Profile[4]. This can include everything from the User Profile information model which extends the MOF-based QML metamodel to the usability of a graphical user interface. The way user preferences are arranged, ranked, labelled and stored also belongs to a profiling mechanism, as does the complete interaction with other DBE components such as the Recommender System. This view of a profiling mechanism can also contain basic algorithms and basic functionality for retrieving and manipulating information and preferences relevant to the user.

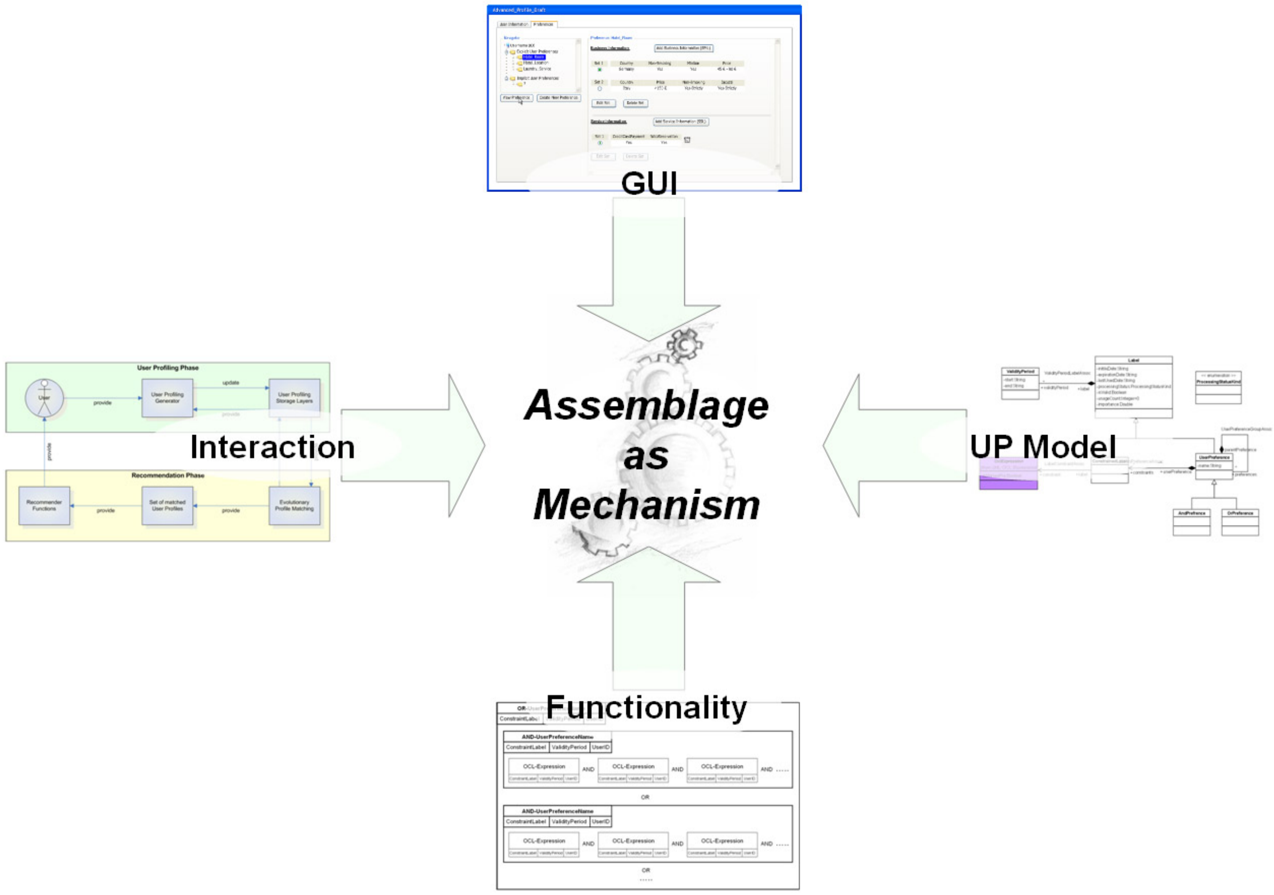


Figure 2: Profiling Mechanism: Assemblage Interpretation

Figure 2 illustrates and clarifies a possible interpretation of the term profiling mechanism. A mechanism is *not* a pure technical aspect using algorithms and code, but more assemblage between important parts which are necessary to create and maintain a user

[4] A User Profile can consist of arbitrary many different kinds of information like demographical data, usage history etc. In this context we put the focus on the term mechanism to user preferences as they build the core information source on what a user's character is represented.

profile. Therefore it is also possible to consider an intuitive graphical user interface to be a part of a good mechanism or an adequate and generally accepted User Profile Metamodel. Functionality as a connecting link between the UPM and the GUI also belongs to a mechanism if the term ‘mechanism’ is interpreted and used in an assembled manner.

For the further work it was decided to follow a holistic approach and to handle the term ‘mechanism’ as an assemblage of usable parts and components which build the User Profile as a whole. As a result of this it can be defined that the User Profile as a whole is considered to be the mechanism. It is assumed that from this perspective the tasks described in Work Package 7 (User Profiling) can be dealt with more efficiently and effectively. This decision is based on the reasons and arguments described later on in Chapter 6.

3 USER PROFILE MODEL

Within the project a DBE user[5] might be interested to receive special services such as personalised recommendations or optimised search result. This obviously requires information about the user, his preferences and maybe also about his behaviour. As the KB infrastructure design "follows the OMG's MOF metadata approach" [PAGK04, p.7] a Metamodel is necessary to integrate User Profiles in the KB Peer Architecture. For this reason a Metamodel for User Profiles has been developed. Currently there is no focus on a special DBE user group such as DBE Developer or DBE Consumer. The user profile in this phase of the project was designed for general usage by a DBE user. After the first real world case studies and experiences adjustments (if really needed) of the UPM might be possible.
The two User Profile Model packages introduced in this chapter have been designed with respect to the basic requirements 'general information about a user' and 'user preferences': The *UserInformation* model package and the *UserPreferences* model package. Both are needed to provide basic information within the user profile. With this approach other components like the Recommender can utilise user information and their preferences to deliver functionality for further services like individual recommendations or other activities like matching profiles. Of course much more information about a user could be stored within a profile like the usage history of searching and selecting services or more detailed information about his business. These are possibilities for later stages in the project to refine the user profiling process and the recommendation functionality.

3.1 MODEL PACKAGE: *USERINFORMATION*

The *UserInformation* model package is used for describing a user by using a basic data set. It also applies a unique user ID to each user preference set. This is needed because a preference set is stored in the Knowledge Base as a single data set which has to be identifiable by a user ID. The source and the administration of User IDs within the DBE hasn't been confirmed as of the time or writing, but a first proposal introduces a DBE Decentralised Identity approach [SEDS05, p. 29 ff].
Additionally, the model package offers the possibility[6] to the user to fill in the User Profile with basic information about them such as 'contact', 'address', 'phone' etc. This kind of

[5] The term DBE User refers to the definition given in [Ferr05].
[6] Although most of the information is strictly required by the profile there are still some attributes in the User Information class which are optional like 'Fax', 'Email' or 'WebURL'.

information is purposely not connected to the generic BML package which contains many classes for business information topics. The reason for this is to avoid possible consistency and configuration problems with the BML business information stored in a Service Manifest (SM). Each Service Manifest is based on a certain model which is typically created by a business analyst to represent a certain type of business. These models contain specific BML and SDL information which can be extended or reused by other SMEs. So the Service Manifest BML is only used in this specific model. For example the contact person in Service Manifest A is different from the contact person in Service Manifest B as both have different responsibilities for the described service in the respective Service Manifest.

To describe information about a user in a User Profile, an additional model is necessary because the generic package of BML contains much more information classes than are really needed for describing general user information in a profile. As a result of several discussions with the corresponding DBE partners it was decided to simplify the usability and the handling of consistency in this development phase. Thus a basic information set was integrated in the *UserInformation* model independently from the generic BML package. It is assumed that this kind of information is always required for doing business using the DBE and every person could be able to provide this. In the further course of the User Profile design and development cooperation with other DBE partners, options are thinkable for extending the number and granularity of user information (e.g. using a new Metamodel or referencing the already existing BML generic package), to provide an optimised recommendation quality.

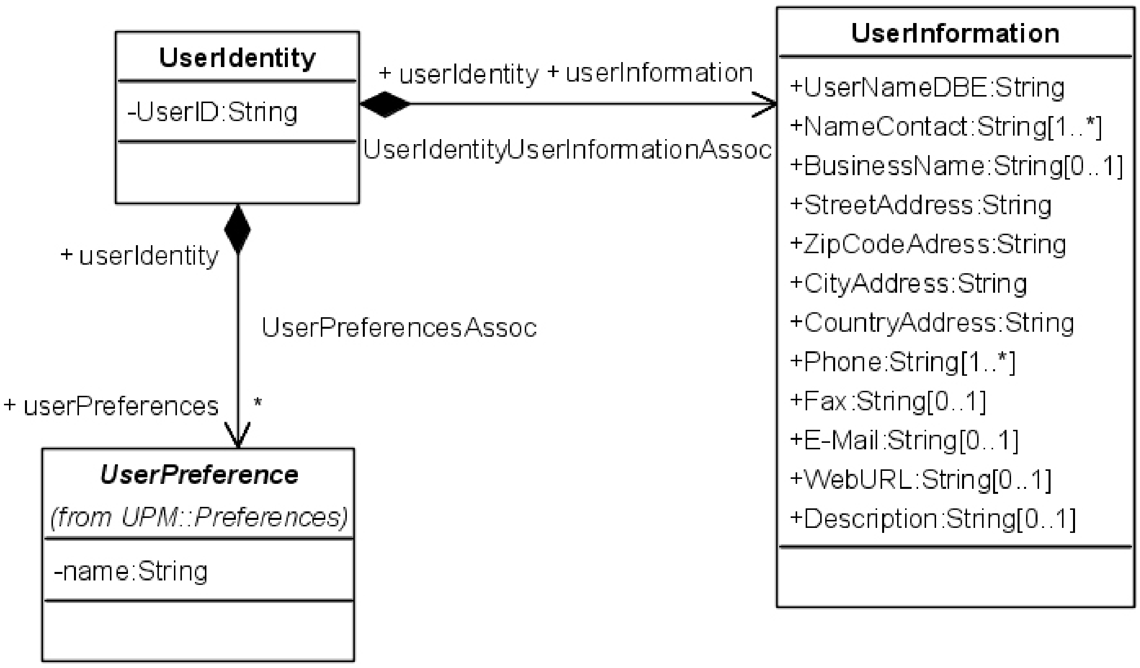


Figure 3: UserInformation Core Package

The connection between the classes in Figure 3 is as follows: Each *UserPreference* is aggregated with a unique *UserIdentity*. The UserID of type String is created once and associated to every newly created *UserPreference*. Also the *UserInformation* is associated with this UserID. Any set of information can easily be identified in the KB and assembled by the recommender using the UserID.

Class '*UserIdentity*'	
UserID	The unique User Identifier in the DBE
Class '*UserPreference*' – Reference	
name	Each User Preference has a name (details can be found in section 4.2)
Class '*UserInformation*' – Reference	
UserName	A natural language user name assigned to the UserID
NameContact	The name of a real world person which is responsible contact for the User Profile
BusinessName	The Name of the Business, Company, Branch etc.
StreetAddress	Address of the Business
ZipCodeAddress	ZipCode belonging to the City
CityAddress	Place of Business (Name of the City)
CountryAddress	Place of Business (Name of the Country the City is in)
Phone	Common Phone Contact
Fax	Common Fax Contact
Email	Common Email Contact
WebURL	Business Internet Address
Description	General Description about the business or any other additional information

Table 1: Description for ‚UserInformation Core Package'

Additional information which is proposed to be in a User Profile is a standardised categorisation code for an SME. For this purpose the SME categorisation code developed by Neil Rathbone ('Rathbone Code') lends itself to be useful [Rath05]. One suggested approach is that an easy to use SME-Categorisation-Wizard be developed to guide a user through a set of relevant questions. The User's answers could then be used to determine the most applicable SME Category Code. The development of a wizard or a tool with similar functionality is not within the scope of the User Profiling work package.

3.2 MODEL PACKAGE: *USERPREFERENCES*

The class diagram *UserPreference* documents how sets of preferences can be created. These sets can be enriched with a variety of internal labels which allow further handling and processing. Each *UserPreference* is a *Label* with specific information attributes like 'importance', 'initialDate' and so on. A *Label* can have an aggregated v*alidityPeriod* if, for example, the user wants to assign a certain time period for which the preference is valid.

Class 'Label'	
initialDate	The date when the data set was created
expirationDate	The date when the data set will expire
lastUseDate	The date when the data set was used the last time
processingStatus	Shows a status information if a data set was created manually by the user or if it was created automatically by a user profiling mechanism
isValid	A flag which shows if a data set is valid and available for profiling purposes
usageCount	A data set specific counter which increases anytime the data set was used by a component for processing activities
importance	Gives a value between the interval 0 and 1 for defining the importance of a preference attribute (0=not important, 1=very important)
Class 'ValidityPeriod'	
start	starting date of the period
end	end date of the period
Class 'InvariantContextDecl'	
pathname	describes the path of the preference
Class 'ProcessingStatusKind' (<<enumeration>>)	
Contains status attributes to signalise if a preference set was added or changed automatically or manually	
Class 'UserPreference'	
name	each User Preference has a name
Classes 'AndPreference' / 'OrPreference'	
Subclasses of UserPreference used for defining either concatenated And-Preferences or alternative preferences (see Figure 5)	

Table 2: Description for ‚UserPreferences Core Package'

A *UserPreference* aggregates *ConstrainedLabels* which build preference sets using *OclExpressions*. AND- and OR-operators for preference conjunctions and disjunctions are needed for realising nested preference expressions and preference sets.

The UserPreference diagram in Figure 4 visualises the model for representing user preferences using a *InvariantContextDec* class, which is defines a specific criterion in the preference. In this context a superclass *Label* was introduced to provide attributes for labelling preferences. Each Label can own a *ValidityPeriod* which will be useful for periodical service requests. This means that a service may only be needed between certain periods of time. If the period is valid the preference will be taken into account by the recommender, if not the recommender will ignore the preference. A *UserPreference* class is a subclass of *Label* just like the class *ConstraintedLabel* is. The complete QML uses this class as the abstract super-class of all other expressions in the metamodel. Deliverable [KaKK05, pages 22-42] describes the QML in detail. A preference can be nested using other preference. The nesting is realised by creating *AndPreferences* or *OrPreferences* which are both subclasses of *UserPreference*. The enumeration class *ProcessingStatusKind* includes information about the usage status of a preference. Either it is created automatically or it is manually set by the user. If the user, for example, creates a preference or a preference set by hand, than the status of this preference set will be labelled with ‘manual’. If they are created in other ways then there will be other labels such as ‘automatically’. The concrete usage and implementation of the aforementioned classes is shown in Figure 5 and in the User Profiling Scenarios in chapter 4.

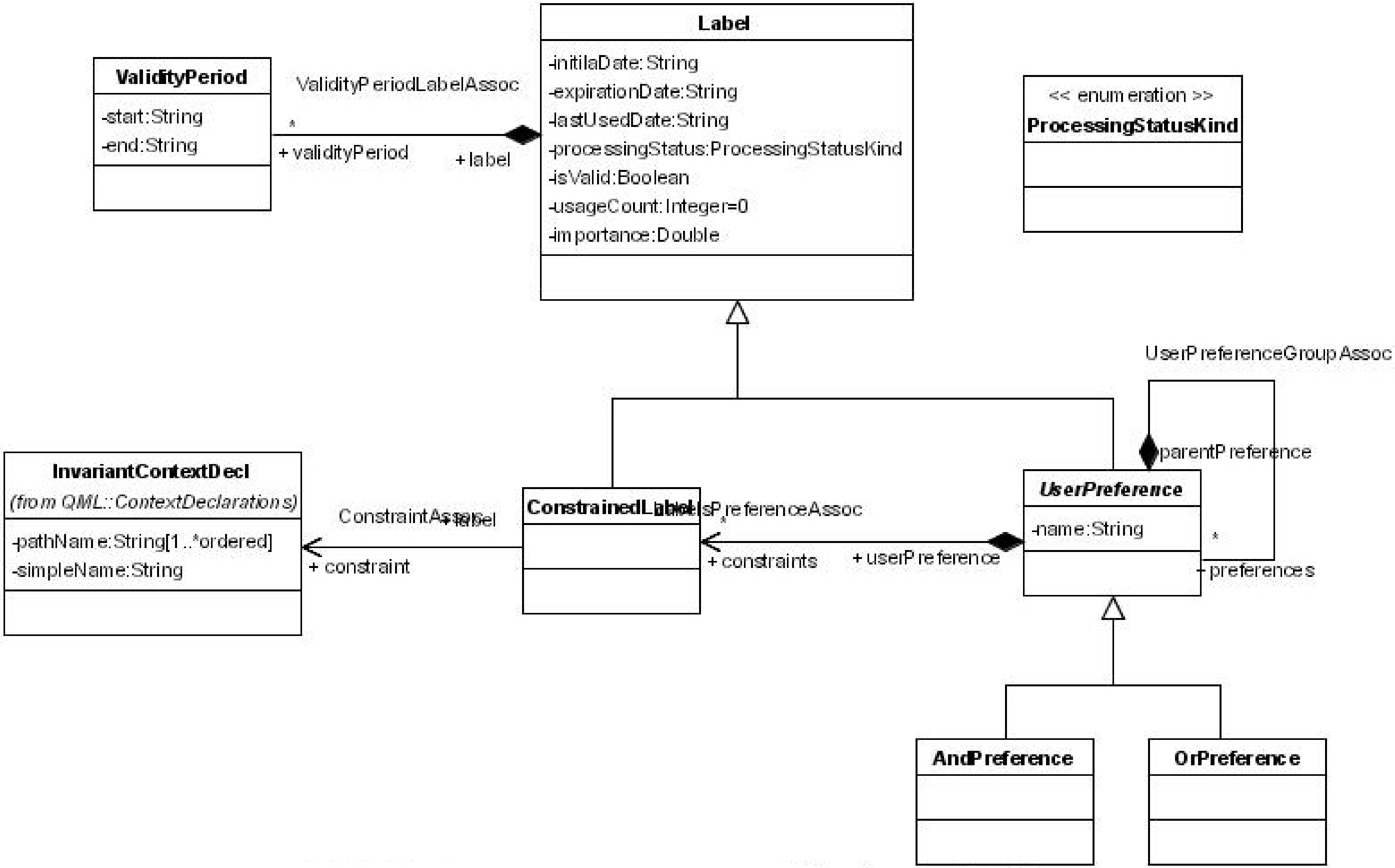


Figure 4: UserPreference Core Package

3.3 USING THE USER PREFERENCE MODEL

The following illustration demonstrates how the aforementioned *UserPreference* model can be used when creating a preference. The structure in Figure 5 will become more evident when the concrete GUI design approach is covered in Chapter 4.

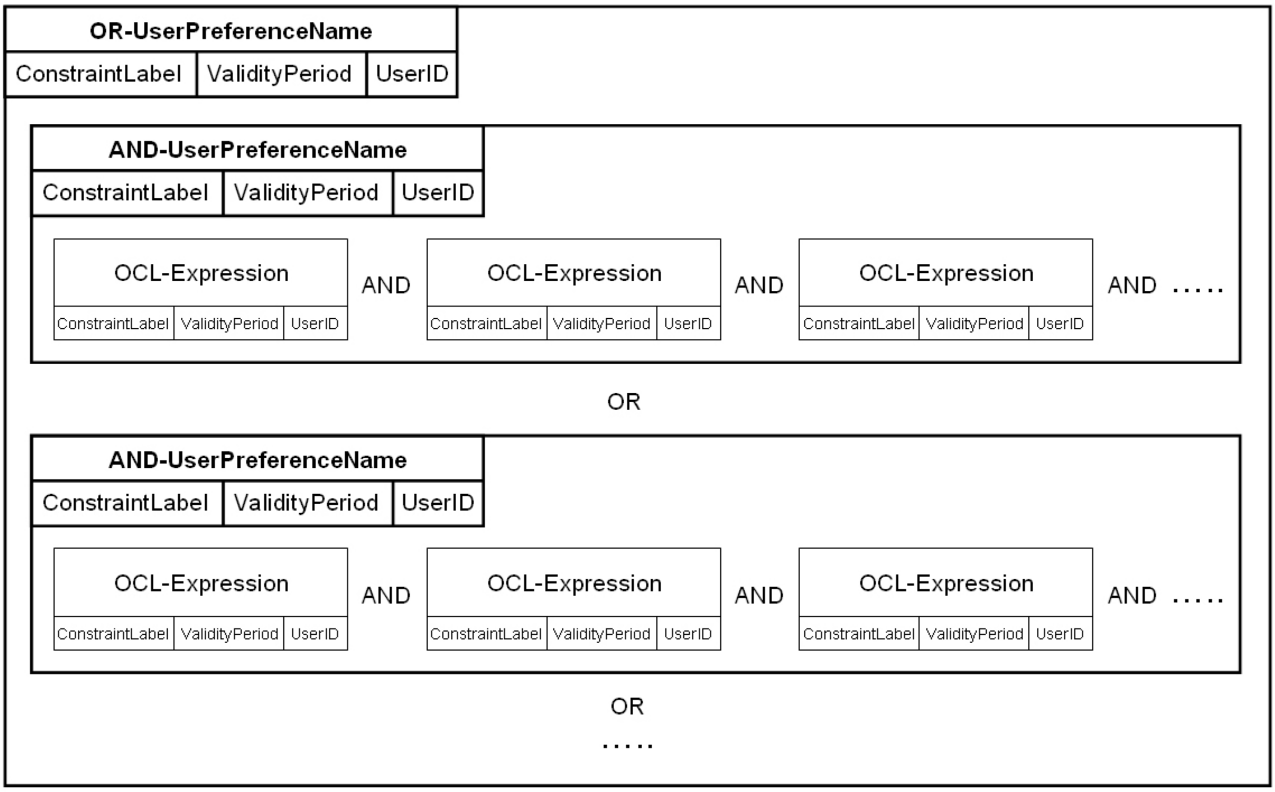


Figure 5: Visualisation of a Preference Set

In the diagram a preference is defined as an *OR-UserPreference* consisting of so called *AND-UserPreferences*. The number of *AND-UserPreferences* is not limited. Each AND-*UserPreference* contains OCL-Expressions which are concatenated with AND operators to express preferences which belong together. Theoretically an unlimited number of arbitrary OCL-Expressions can exist within an *AND-UserPreference* but is has to be kept in mind that the more AND concatenated OCL-Expressions that exist within a preference, the lower the probability will be for a successful validation process. This is because any given attribute is part of an OCL-Expression. So, the more attributes a preference contains the more variables that have to be valid to make an *AND-UserPreference* true. For further profiling activities each OCL-Expression and preference has a certain label with time stamps and other information as well as a UserID which allows the Recommender and

other components to assign each preference to a specific user. A AND-Preference in the GUI example in chapter 4 is interpreted as a Preference Set in the User Profile. OR-Preferences can be interpreted as a number of different Preference Sets within a User Preference.

4 USER PROFILING SCENARIOS

The following use cases are described exclusively from the User Profiling point of view and show the basic associated activities. In this phase an eye is kept on the possibilities of searching, selecting and arranging preferences in a User Profile. These procedures are based on explicit user interaction as implicit algorithm-based mechanisms have not been implemented at this stage of the project. Chapter 6 will introduce and discuss possible options for applying implicit user behaviour mechanisms to a user profile.

4.1 USING "EVENT-DRIVEN PROCESS CHAINS" (EPC)

Event-driven process chains (EPC) will be used to illustrate the processes and work-flows in the following use cases. The extended EPCs method has been developed within the framework of ARIS (Architecture for Integrated Information Systems) [Sche94] and is used by many companies to model, describe and analyse business processes. It defines an architecture for a complete, enterprise-wide information system. ARIS utilises three views, namely, the functional view, the information view, and the organisation view. These views are defined in all life cycle phases of the information system, namely, requirements definition, design specification, and implementation description. All three views are treated in isolation, and the relationships between the three views are represented by a control view. Business processes are described by process chain diagrams.
An EPC is an ordered graph of events and functions. It provides various connectors that allow alternative and parallel execution of processes. Furthermore it is specified by the usages of logical operators, such as OR, AND, and XOR. A major strength of EPC is its simplicity and easy-to-understand notation. This makes EPC a widely acceptable technique to denote business processes [Wiki05a]. In addition to describing the control flow, an EPC diagram has constructs to show the information/data necessary towards the development of an information system. Another distinguishing feature of the EPC diagram is the explicit modelling of events. The basic elements of the event-driven process chain are shown in Table 3.

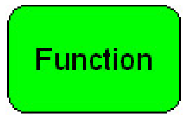
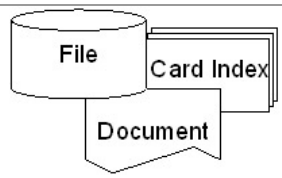
Element	Description	EPC Symbol
Event	A status that has business relevance, that can trigger one or more functions at the point where it occurs, and that can result from a function	
Function	Functions are active elements in EPC. They model the tasks or activities within the company. Functions describe transformations from an initial state to a resulting state. In case different resulting states can occur, the selection of the respective resulting state can be modelled explicitly as a decision function using logical connectors. Functions can be refined into another EPC. In this case it is called hierarchical function.	
Information Object	The representation of a real world object. It can be input data, serving as arguments to a function, or output data, produced by a function.	
Business Organisation Unit	The representation of an enterprise's organisational units, detailing which person within the structure of an enterprise is responsible for a specific function	

Table 3: EPC Elements and Symbols

4.2 SCENARIO 1 - "EVENT ORGANISATION" PROFILE

The ficticious "Royal Business System Hotel" (RBS) is a nice located family business hotel which is well-known for its exclusive ambience and as venue for holding receptions, parties and private views. A unique selling proposition (USP) offered by the Hotel is that events of any size can be organised professionally within a very short time period for very attractive prices. Depending on the requirements of a customer, different event coordination concepts may be needed. There may be, for example, service suppliers which can deliver services up to a certain number of guest or others which need a certain amount of guests before they can submit an attractive quote. When taking catering services, laundry services etc. into account the number of available service suppliers and their business conditions is extremely high and complex.

The hotel always wants to offer the best services with the best prices for these special events to its customers but it doesn't have the time and resources to check and rank all available services, every week, for special offers and prices. At this point the hotel's

service and business preferences come into play with the objective being to evoke an automated personalised service-recommendation process. One of the incentives for the hotel to join DBE is to reach the aforementioned objective by getting recommendations and personalised search results based on individual preference sets defined by the hotel. The hotel registers to the DBE after installing DBE studio and opens the individual User Profile. The hotels experiences of former events are now used to create preferences for certain services.

Currently the Hotel has contracts with one company for each aforementioned service. Although the quality and service of these suppliers is good, the Royal Business System Hotel wants to be more flexible in offering services to its customers. In addition to this it wants to expand its supplier portfolio for each service up to three suppliers per category. This is to make the Hotel independent from single sourcing contracts with current service suppliers.

Conceivably, a company party could have the following requirements:

→ Basic Information

- 10 year anniversary of the company
- 100 expected guests
- party special required

→ Needed services (estimated by the Hotel)

- catering service
- laundry service
- cleaning service
- entertainment service

The following process will illustrate how the User Profile can be used by the RBS Hotel to efficiently fulfil customer requests.

4.2.1 Basic User Profiling Process

This top-level process results in refined sub-processes such as the Profile Updating Process explained later on in this document. In general, preference information which is manually entered into a profile is composed of either former business experiences of the company, or is information explicitly gathered from customers, business partners etc. Thus the basic user profiling process starts with an incoming customer request event.

Customer requirements are gathered and analysed to identify possible services needed to satisfy the user's request for a quotation (labelled ID.1001 in Figure 6). Many different channels like email, phone, fax etc. are possible for getting in contact with a customer. After gathering customer requirements, either an immediate search (push) for potential services using the recommender is possible (ID.1006), or a recommendation process (pull) is activated (ID.1002). The difference between both possibilities is that using the pull-mechanism uses information already available in the profile of the hotel to discover services, whereas the push-mechanism is a manual search for services which could match the customer's request for quotation.

After finishing the recommendation process, the hotel can either directly update its profile with the newly recommended services (ID.1005), or a result can directly be selected from a ranked list to view the service details (ID.1003). If the selected result matches the customer's requirements an adequate proposition can be made (ID.1004).

Using the push-mechanism for searching a specific service either returns a result or returns nothing. If a result has been found, the hotel, which is the owner of the profile, can select a service and finally make a proposition. If no result is returned by the recommender then the hotel can either update its profile with additional preference information (ID.1005) or start searching for an adequate solution using other instruments than the DBE (ID.1007). These alternative tools could include a simple web search using the yellow pages or other search engine, and consulting personal business networks such as partners, friends and so on.

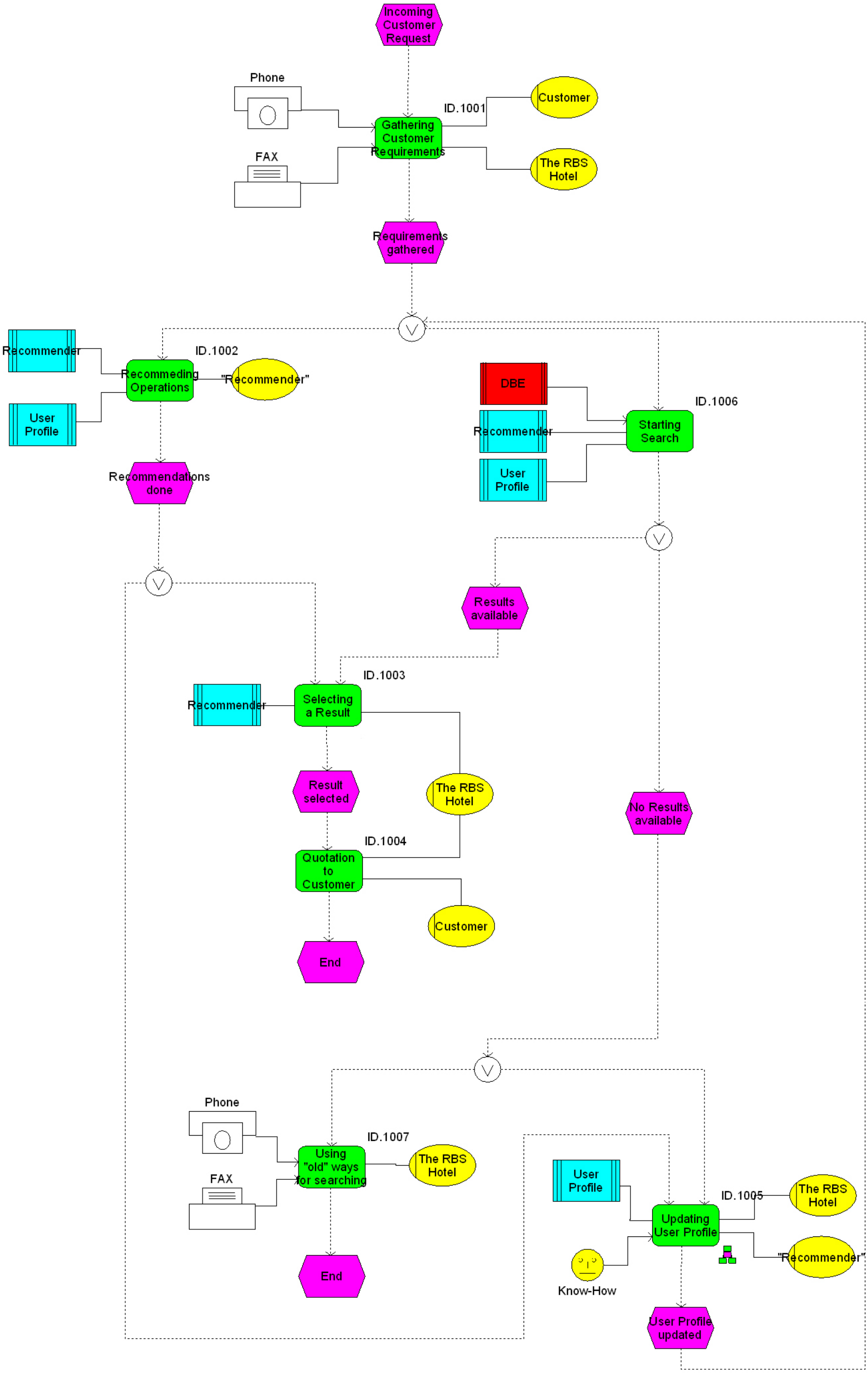


Figure 6: Basic User Profiling Process

4.2.2 Basic User Profile Updating / Editing Process

A refinement process of the “Updating User Profile” Function (ID.1005) mentioned in the process in section 4.2.1 is explained in detail in the following paragraphs.

After the User Profile is completely loaded, it appears visible to the user (ID.2001). In the current User Profile the user can choose between a Preference Tab (ID.2002) or a User Information Tab (ID.2022). Selecting the Preference Tab enables the User either to create a complete new preference (ID.2004) or to select a preference already existing in the profile (ID.2003). When creating a new preference it is required that it be given a name (ID.2005). Now the preference can be loaded and made visible to the User (ID.2006). Three different activities are now possible with a selected and loaded preference. The user has the option either to add business information (ID.2007) or service information (ID.2021) to the preference. Additionally it is possible to select a preference set which already exists within a preference to edit the set (ID.2018).

If choosing the add business information option the user uses the adapted functionality of the Query Formulator Tool (QF) to browse and select appropriate M1 level models and enter M0 level data[7] (ID.2008 and ID.2009). If an adequate model has been found with the tool the user can select relevant business information attributes (ID. 2010), edit them to their needs (ID.2011) and finally add this attribute data to the current preference set (ID.2012). Of course this can be repeated for other attributes if desired. To complete the activated preference set it has to be saved to the User Profile (ID.2013). The procedure for adding service information to a preference set is very similar to the aforementioned approach apart from the fact that the focus is on business services and their attributes and settings (ID.2014 to ID.2017).

As mentioned previously, the user can also select an already existing preference set for editing or deleting purposes (ID.2018). If they decide to delete a preference set (ID.2020) the complete set will be deleted but the preference containing it will be kept as maybe there are other preference sets in the preference. The edit function enables the user to edit business attributes (ID.2011) or business services (ID.2015) in a preference set. If all changes within a preference set are completed, the user will see the basic User Profile screen again (ID.2006) to perform other profile activities.

The user may also wish to add some basic information to their profile, or to edit their existing user information. In this case they will choose the User Information Tab to evoke the user information profile view (ID.2022). Within this view the user can add or edit basic data like name, company, contact etc. (ID.2023) and save it to their profile (ID.2024).

[7] M1 and M0 level data refers to the hierarchical MDA approach which is used in the project. Information on M1 level describes an attribute like ‚name’, ‚size’ etc. Information on M0 level ist he concrete value of the attribute like ‚Hilton’, ‚large’ etc.

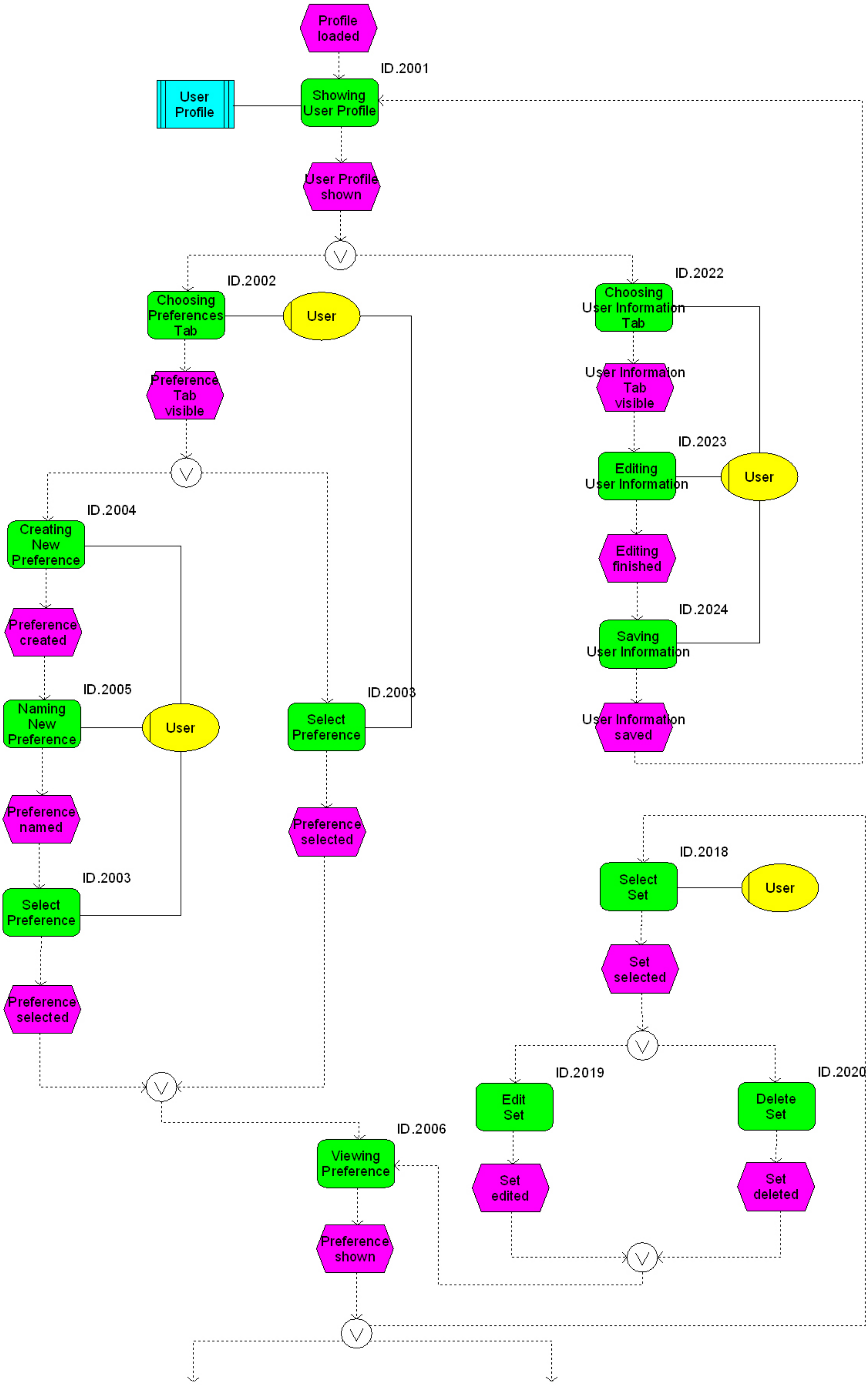


Figure 7: Basic User Profile Updating / Editing Process (1 of 2)

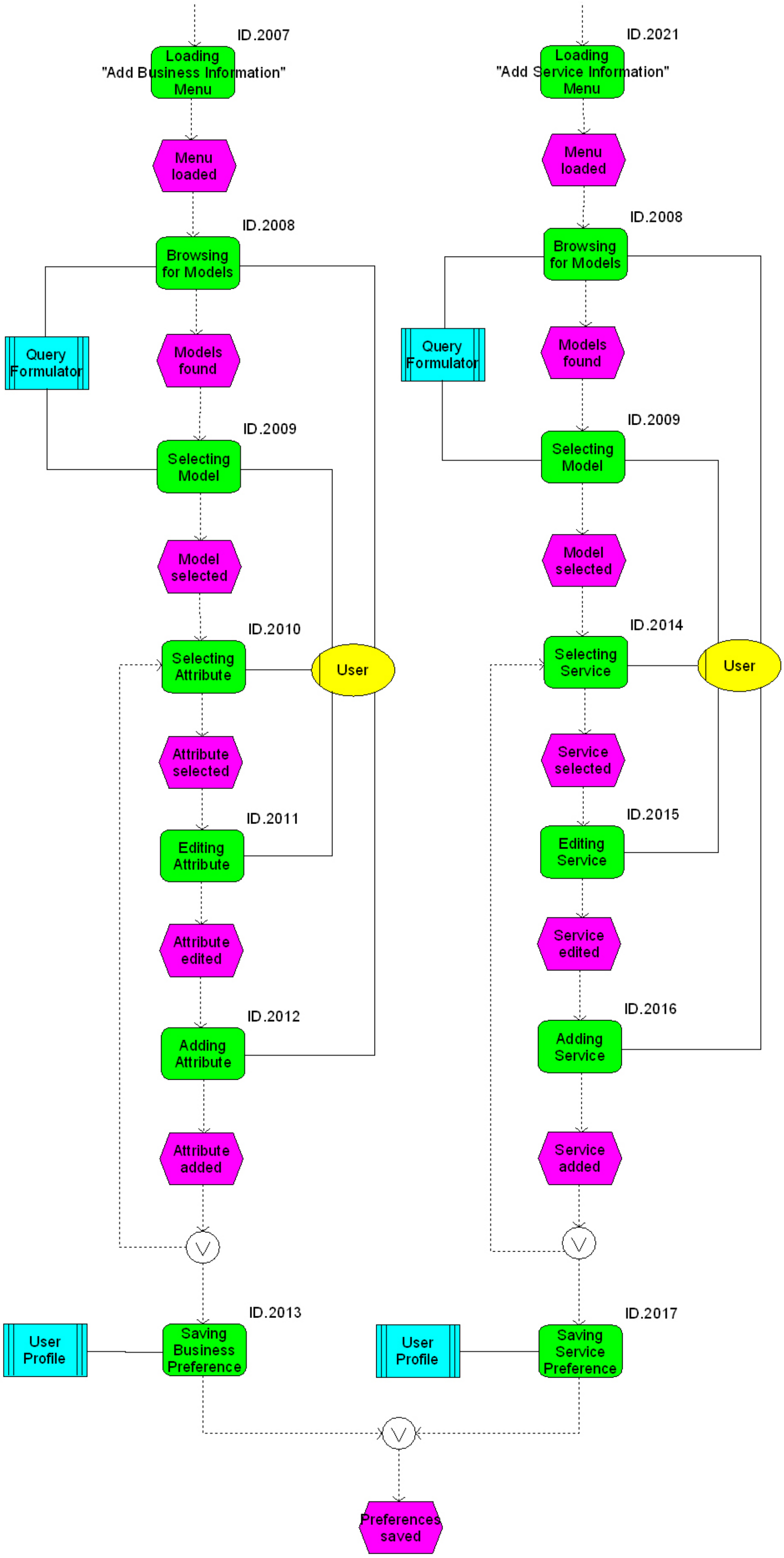


Figure 8: Basic User Profile Updating / Editing Process (2 of 2)

4.2.3 Complete GUI Example for Scenario 1

The following figures illustrate the processes described in sections 4.2.1 and 4.2.2 using a draft design for a future User Profile editor interface. The graphics were designed with Microsoft Visio 2003 and are not exhaustive. The current Java Eclipse Plug-In development approach uses the Java SWT for graphical representation. A final corporate DBE design User Interface based on the work of WP 20 will be added to the User Profile in a later stage of the project.

In Figure 9 a complete new preference is explicitly created by the user. Therefore a new name is required for this preference. This Preference will act as a 'container' which comprehends different preference sets belonging to the same main preference.

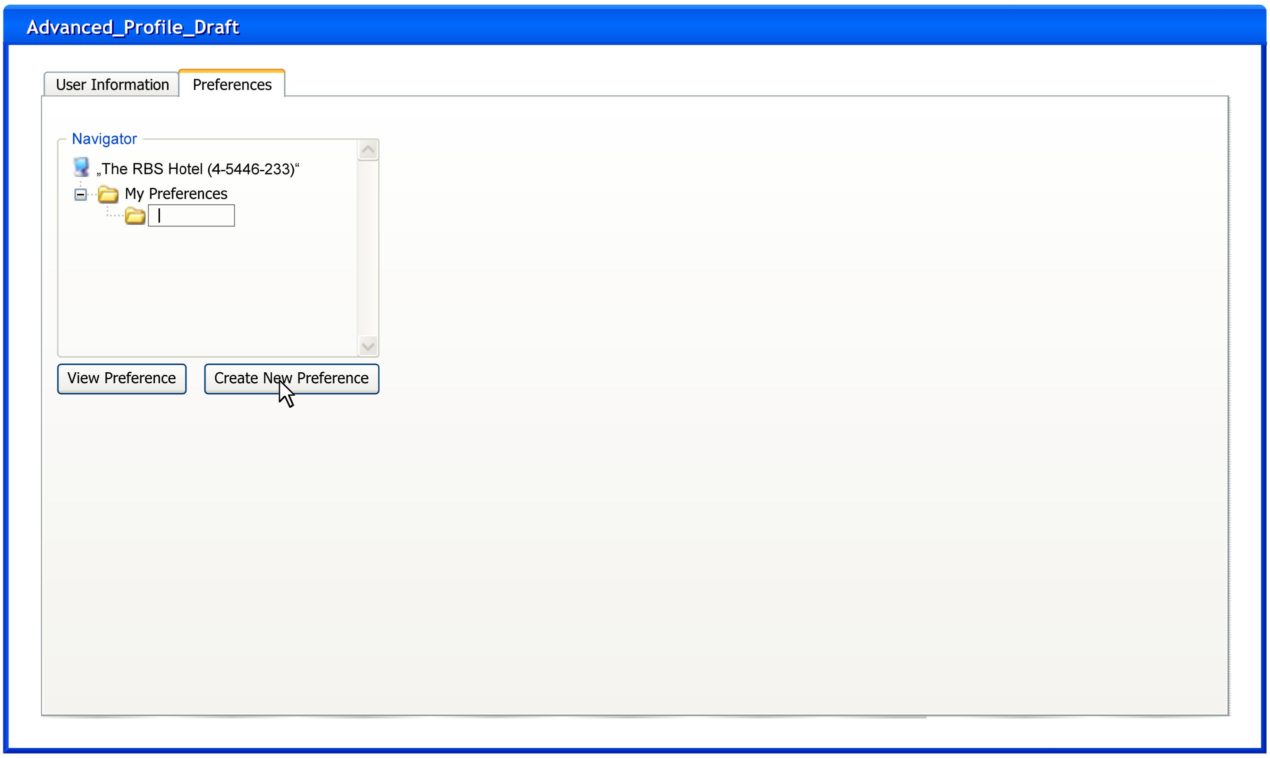


Figure 9: Scenario 1 – Creating a new Preference

In this case the new preference is named "Catering Service" which means that all further preference sets belonging to this subject area will be combined as a part of this main preference. Business Information and Service Information are separated for clarity purposes. Each preference set which is now created, is a part of the Catering Service preference. It is possible to create as many preference sets as the user wants to. With this approach it is possible for the user to express any business or service preference combinations they like. The Recommender will take complete preference sets into account when working in the pull or push mode, so the user has the freedom to define arbitrarily many preference sets containing different conjunctions of attributes or services. The separation of Service Information and Business Information is useful for usability reasons

as the clearness of the preferences is better. It’s theoretically possible to mix up service and business information but this could be confusing for a user like e.g. an SME.

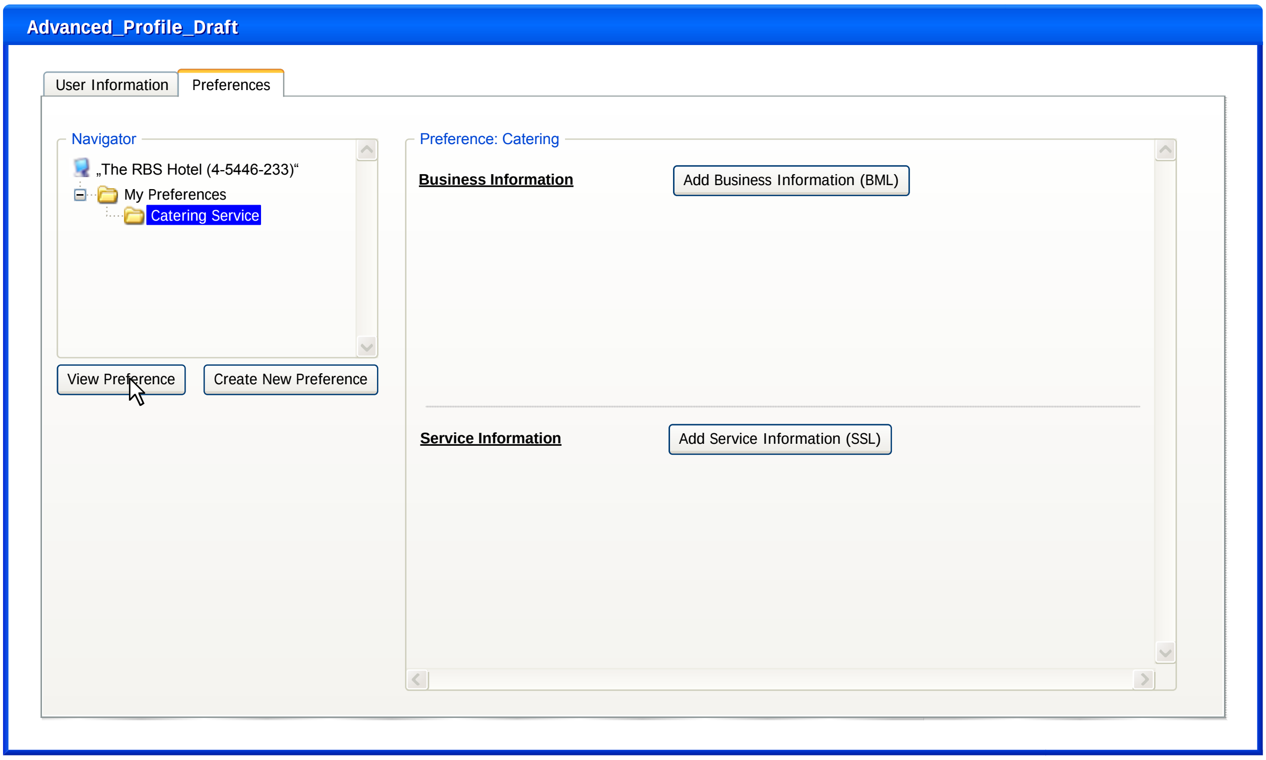


Figure 10: Scenario 1 – Invoking Preference Details

The point where the user has already chosen a model and has selected a certain attribute from this model is visualised in Figure 11. The model search process is supported by the Query Formulator Tool which is adapted for User Profiling purposes. A more detailed discussion on searching and cross-composing attributes from available services within the current non-hierarchical structure is provided in chapter 5. If an attribute (in this case ‘City’) is selected from the model found with the adapted Query Formulator Tool, several different settings can be supplied. The attribute ‘City’ can own an individual value which is assigned to the attribute using operators ‘smaller than’, ‘greater than’ or ‘equals’. As only the equals-operator makes sense when using a string attribute, this operator is automatically selected. Additionally the user can assign an ‘Importance’ status to the attribute which mirrors the user’s appraisal of how relevant the attribute, and its value, really is to them. This fuzzy-mechanism-based evaluation is very important for further recommendation processes and has a range from zero to one hundred percent. This range can be interpreted as ‘not very important but nice to have’ up to ‘extremely important and a must’. The attribute ‘City’ is very important and the preference set will only be valid if there will be

an exact match with the value ‘London’. After all settings are done the attribute is added to the preference set which is visible in the lower part of the “Add Preference Window”.

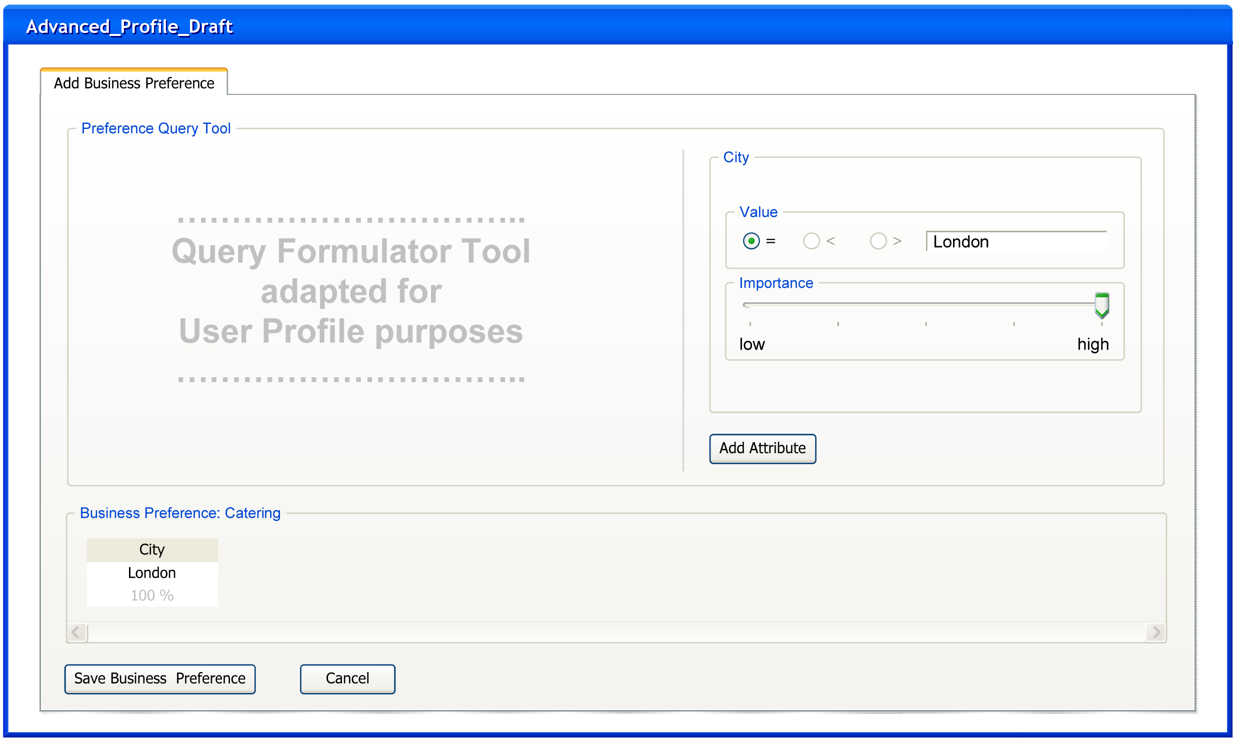


Figure 11: Scenario 1 – Adding Business Preference 1

In Figure 12 the user has added an attribute ‘Capacity’ to the preference set which already includes the attribute ‘City’. Here they decided that the capacity should be larger than 100.

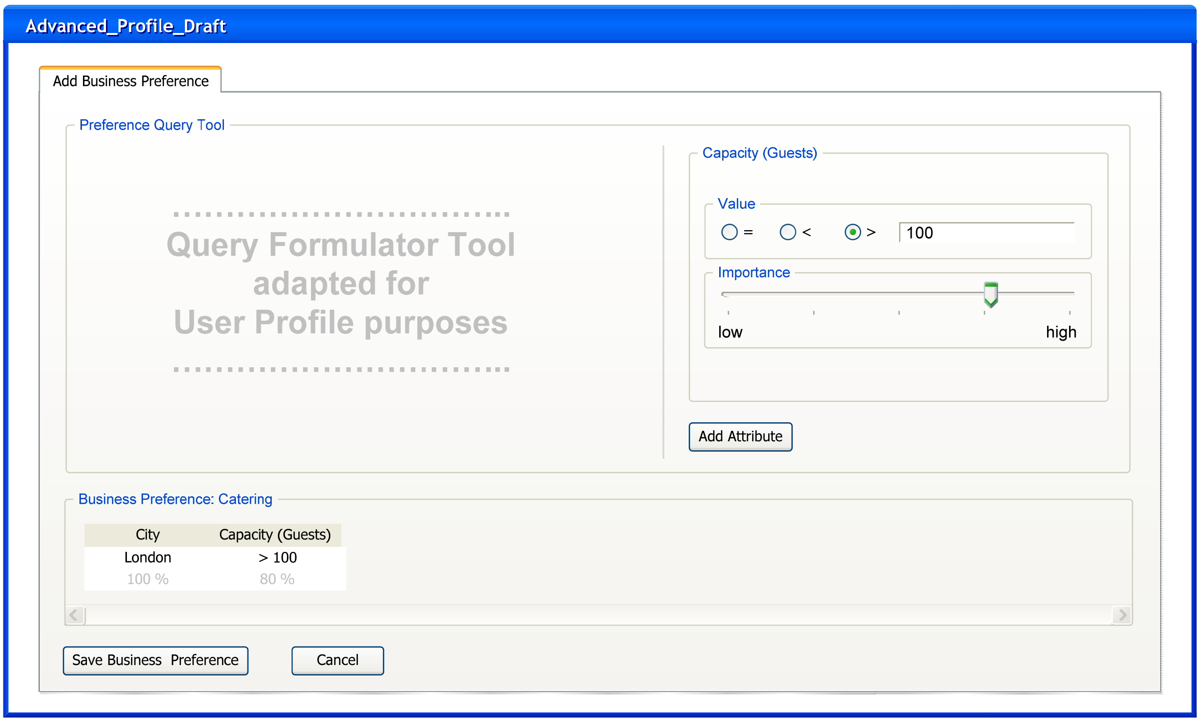


Figure 12: Scenario 1 – Adding Business Preference 2

Additionally this attribute was assessed to be very important but not critical. The interpretation of the preference set so far could be as follows: The user prefers a catering service which has to be located in the city of London and should be able to serve foods and drinks for at least a hundred guests. As this is not a 'must have' but only a 'should be' (estimated 80 percent importance) also catering services with the capacity value of e.g. 90 persons will be taken into account to validate the preference set to be true.

Figure 13 demonstrates the case where a Boolean attribute has been added. Here a Mobile-Cocktail Bar should be available. The importance slider shows that this attribute is very important.

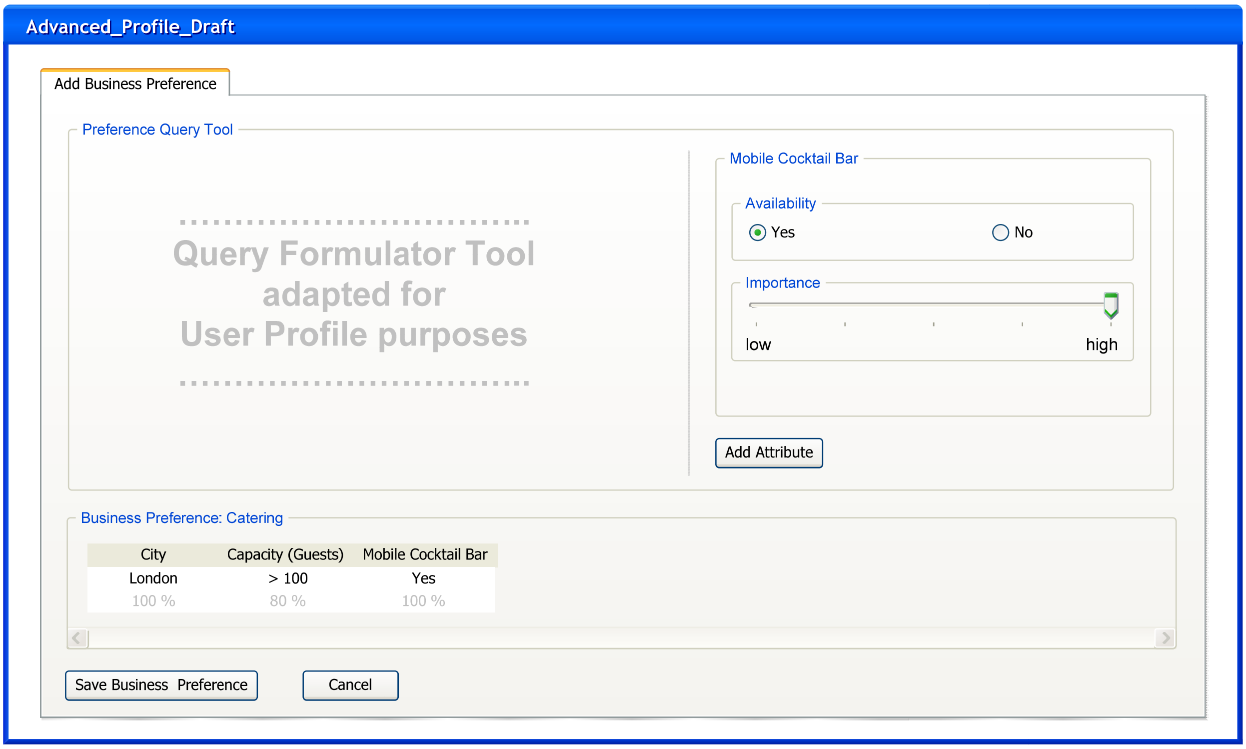


Figure 13: Scenario 1 – Adding Business Preference 3

The following Figure 14 shows the summary window for the "Catering Service" Preference. At this stage it includes one Business Preference Set consisting of three concatenated attributes. A Service Preference Set is not created yet. The Recommender will make suggestions regarding catering services taking the complete Business Preference Set 1 into account. It is possible for the user to edit or add preference sets at any time.

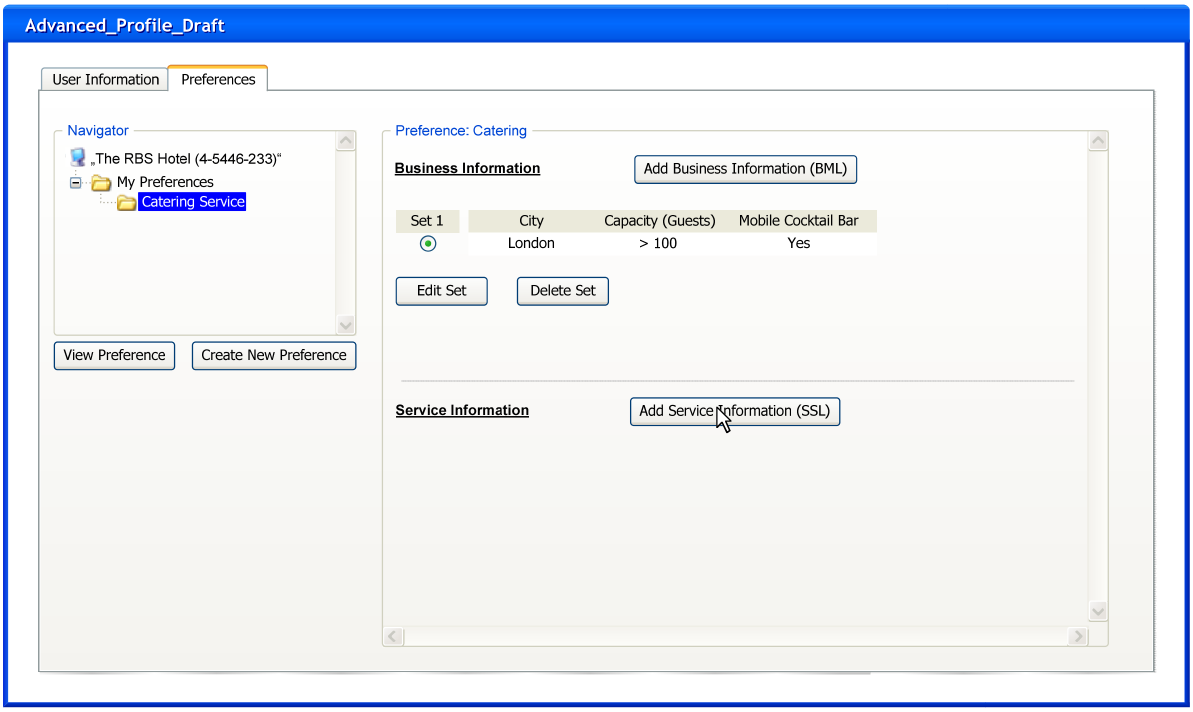


Figure 14: Scenario 1 – Preference Overview 1

We now create the Service Preference Set which is part of the "Catering Service" preference. Again the Query Formulator Tool is used to browse and select adequate services from models found by searching and browsing all available models using the tool. In this case a service of interest would be the "CreditCardPayment" Service. Also here the services should be available in the profile using the boolean attribute. The importance slider defines the service to be very important.

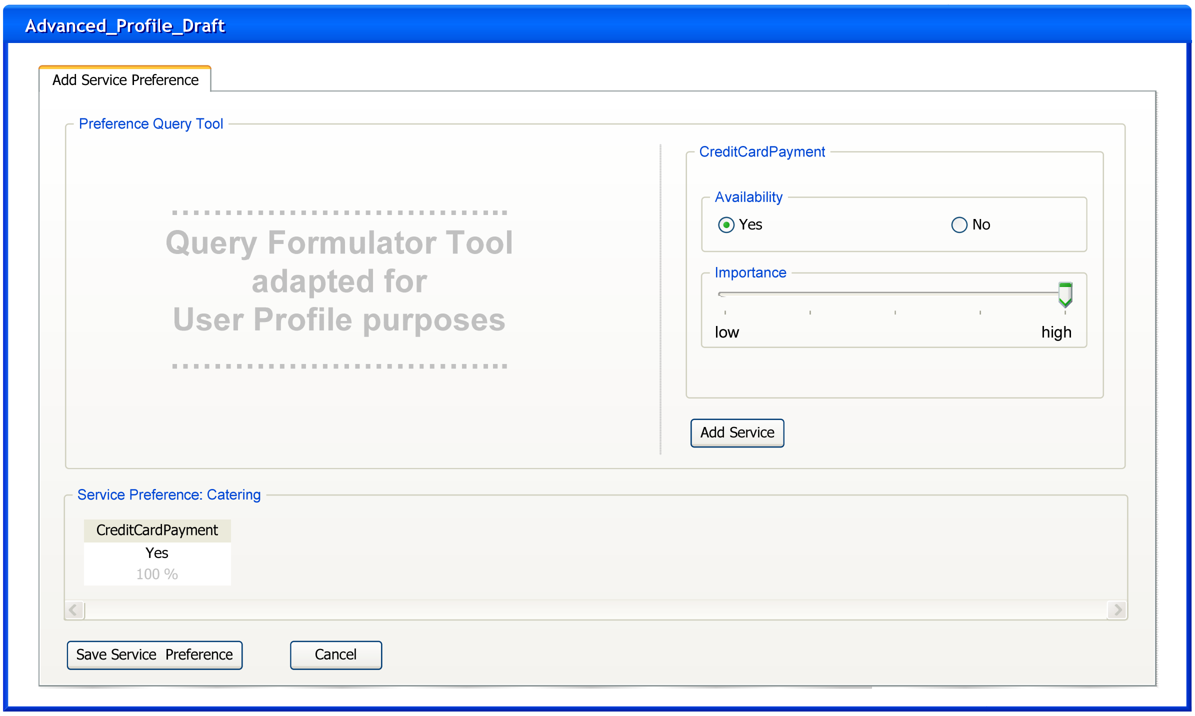


Figure 15: Scenario 1 – Adding Service Preference 1

Creating service preferences can also be realised by assigning specific values to attributes using operators (see Figure 16). In this case the user likes to find a service which offers a discount of at least 2 percent if the customer does cash transactions. The importance of this feature is only at 50 percent, so it is nice to have, but not really important.

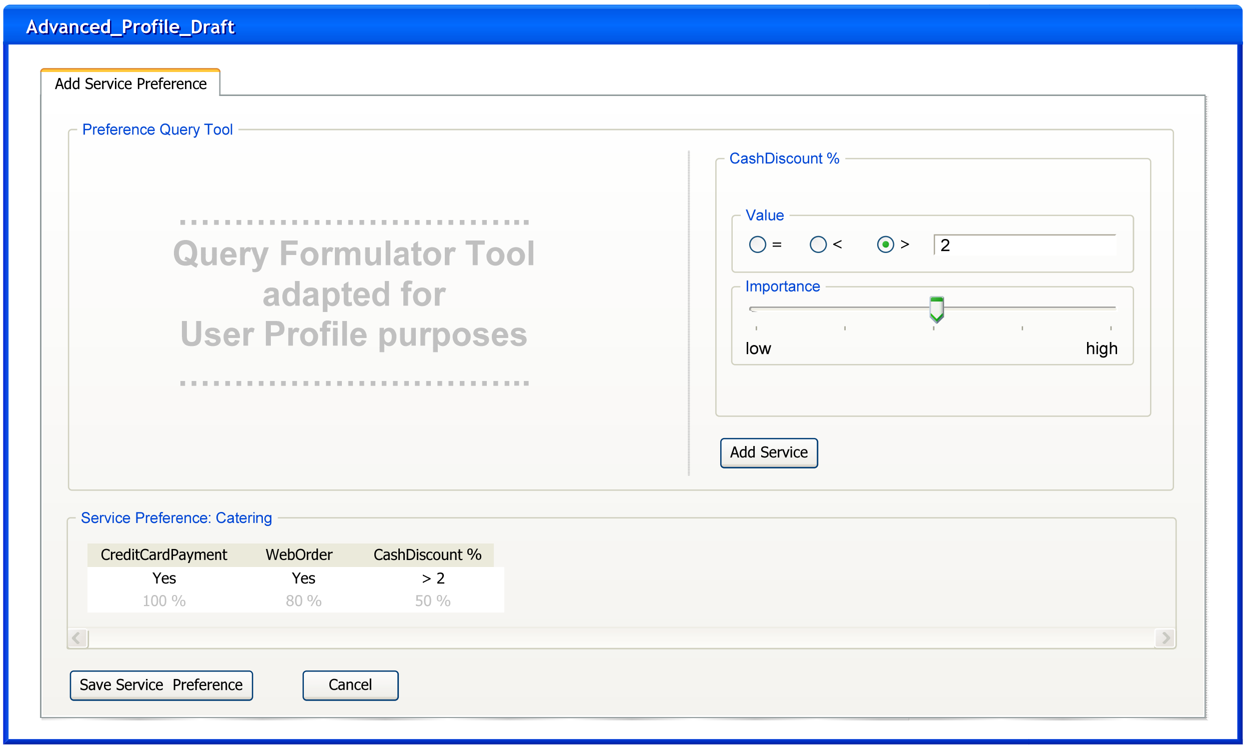


Figure 16: Scenario 1 – Adding Service Preference 3

If the user selects the “Catering Service” preference they find an overview of all preference sets available in this preference. A high degree of flexibility exists for creating and composing preferences and attributes. During a recommendation process all sets within a preference are taken into account, are rated and used for matching the User Profile’s data with any other available and useful information to identify the most personalised services.

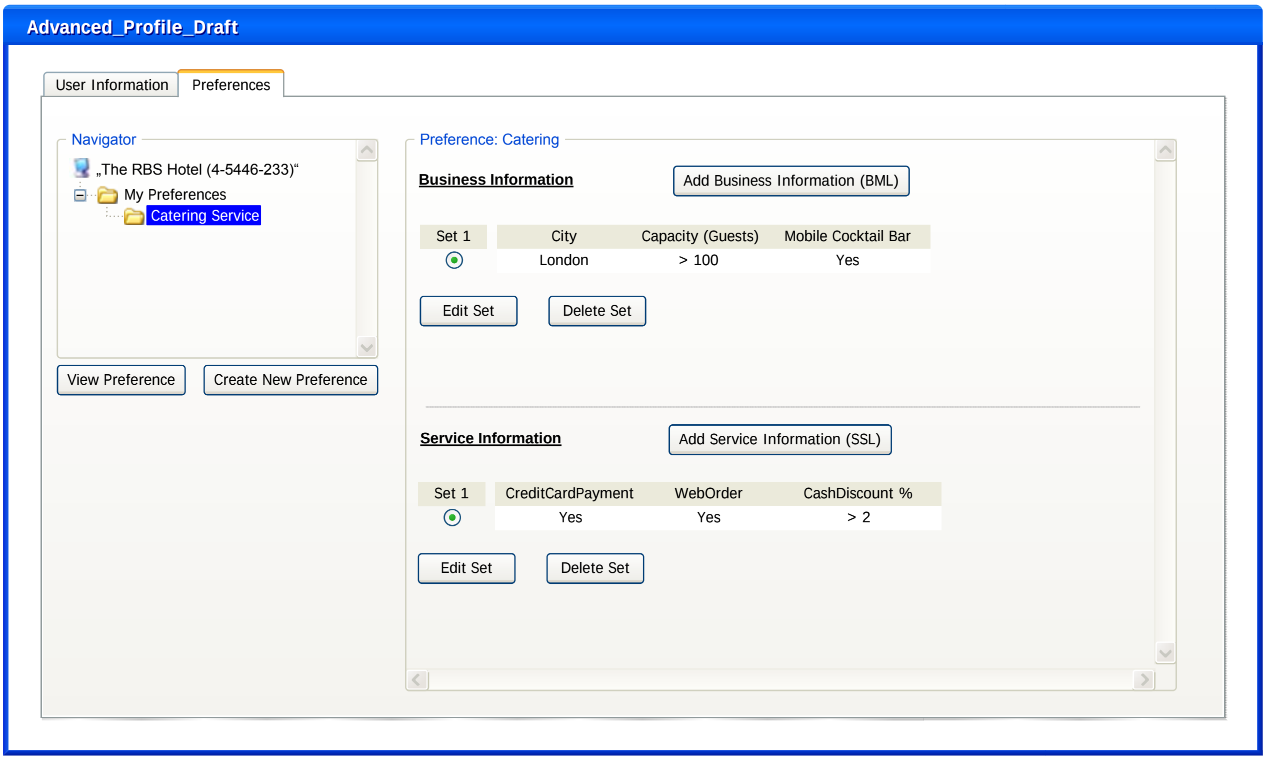


Figure 17: Scenario 1 – Preference Overview 2

Finally the hotel, which is the user in this example, has defined three more preferences “Laundry Service”, “Cleaning Service” and “Entertainment Service” which will probably cover the original customer requirements described in the beginning of this section (see Figure 18). Based on this kind of preference definition the hotel is able to search for adequate services within the DBE system to identify adequate service providers. There could be, for example, a number of laundry services which offer special deals for hotels, the hotel has never heard before. Although the final decision of what service will be chosen remains with the user, the benefit of this preference approach comes from personalised recommendations taking specifically concatenated user preferences into account. Even if the user doesn’t look actively for a new service, the Recommender works in the “back-office” and makes suggestions and recommendations which, the user can retrieve the next time they log into their account. Of course several other notification services including the identification of new services are possible. For example, the user could receive a short message via their mobile phone when the Recommender has found new personalised services. In this case the user could open their User Profile and check the recommendations, and they may choose a service or even a model on which new preference sets can be created.

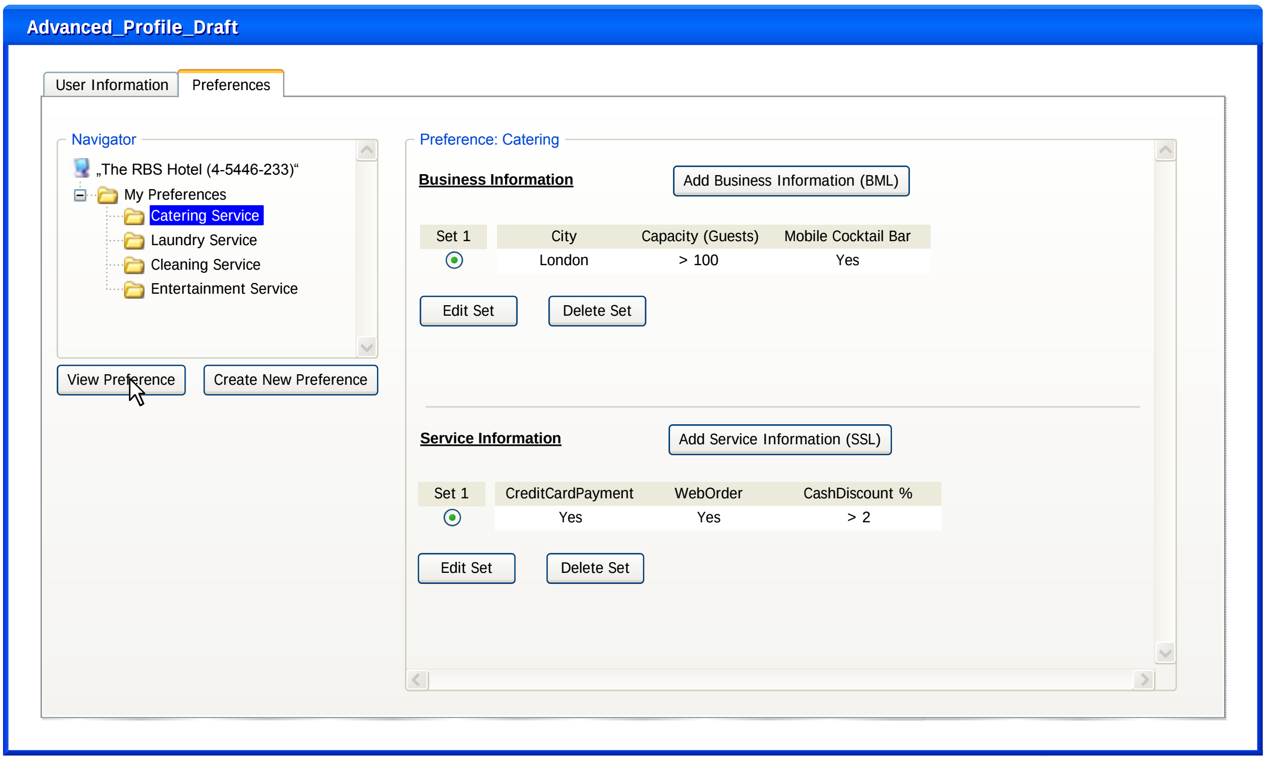


Figure 18: Scenario 1 – All Use Case Preferences

In addition to the User Preferences it is also possible to enter some general information about the user by activating the User Information Tab (see Figure 19). This information could ideally be the basis for more sophisticated recommendation processes like matching User Profiles, or making suggestions and recommendations by matching based on some personal information attributes like ‘City’ or ‘Website’. Comprehensive business information will not be represented in the first User Profile release as only the basics will be of interest in the general User Information Tab.

A general approach, where the user has the possibility to add arbitrary information about them is thinkable to be realised in a later phase of the project (e.g. referencing or extending the existing BML model). Then, the current *UserInformation* model package (see Section 3.1) could more general if several challenges regarding the BML integration are solved. For a start it is assumed that the data mentioned in Table 4 can be provided by any potential DBE participant without problems.

Following attributes are suggested for the User Information Tab:

User Name	Zip / Postal Code	Fax
Contact Name	City	Email
Business Name	Country	Website
Street	Phone	Description

Table 4: User Information Attributes

Apart from the username, all other attributes can be edited arbitrarily. As a unique UserID will be provided by the DBE system in the User Profile initialisation phase, there is no need for editing this attribute in the User Profile's lifetime.

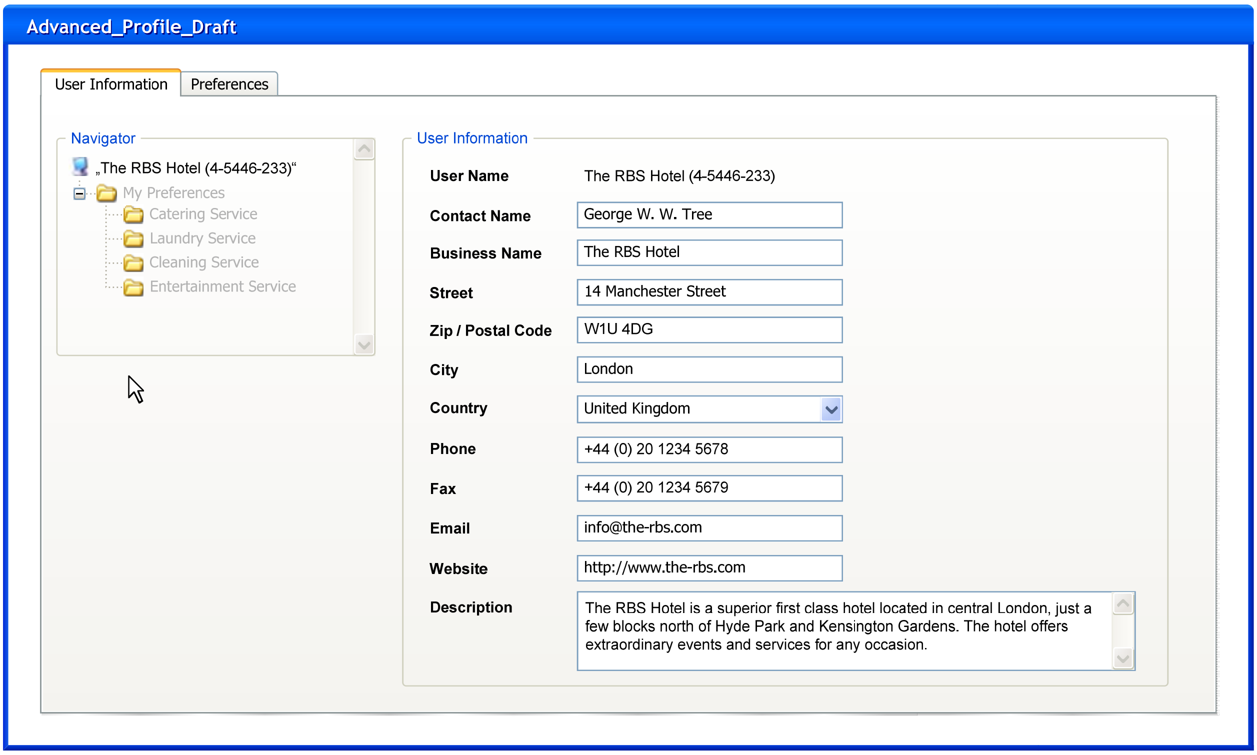


Figure 19: Scenario 1 – User Information

The more information and preferences a User assigns to their profile, the better and more precisely personalised results the Recommender will be able to provide. It is important that the logical structure and consistency of concatenated preferences and attributes should conform to rational thinking. This means, for example, that a user shouldn't define a preference for non-smoking rooms on the one hand but on the other hand also add a preference for smoking rooms. It can thus be seen that the quality of a User Profile and the

recommendation results based on this information depend on the quality and the rationality of the user and their personal preferences. They have to create preferences which fit as close as possible to their real world preferences. This “manual” part of creating preferences is the initial and the most important one as several difficulties are associated with the implicit creation of preferences which is introduced and discussed in chapter 6. Another important factor affecting preferences is the information contained in the services found with the Query Formulator. Several indicators are needed to decide what kind of attribute values will be made available for editing to the user. Some examples may clarify this:

QF-Action[8]: The user selects the attribute “price” form a service

UP-Action: The profile provides a “From Value1(Int) to Value2(Int)” Box

QF-Action: The user selects the attribute “city” from a service

UP-Action: The profile provides a simple “textfield(String)” Box

QF-Action: The user selects the attribute “creditCardPayment” from a service

UP-Action: The profile provides a “Yes(Boolean) / No(Boolean)” Checkbox

Further work on the information content of services will be done as the long-term objective should be an abandonment of translation tables for conversions.

4.3 Scenario 2 – „Web Development“ Profile

The following scenario will explain how a small software development company could use the User Profile to defining preferences.

The fictitious SME web design company “Digital Business Design Web - DBD Web” is specialised in creating websites using the open source Content Management System (CMS) Typo3. The layout and design of professional websites using a Typo3 CMS is defined by specific Typo3 design templates. Although the company has very good skills in developing Typo3 templates it was decided by the management to enhance these layouts using sophisticated Macromedia Flash! animations and menus. Even Macromedia Shockwave movies and intros are taken into account to increase the company’s design

[8] QF-Action = Query Formulator Action, UP-Action = User Profile Action

portfolio. Their objective is to reach a state-of-the-art level for web design. Unfortunately the company doesn't have enough orders to employ a Macromedia design specialist, so they decide to locate external companies which offer professional and affordable just-in-time services for the development of Macromedia Flash! and Shockwave applications.

Additionally, the company "DBD Web" is interested in services offering individual online learning courses for Macromedia Flash! application development. Figure 20 visualises a possible User Profile which takes most of the aforementioned requirements into account. Regarding the "Macromedia Service" Preferences the Profile owns three Preference Sets.

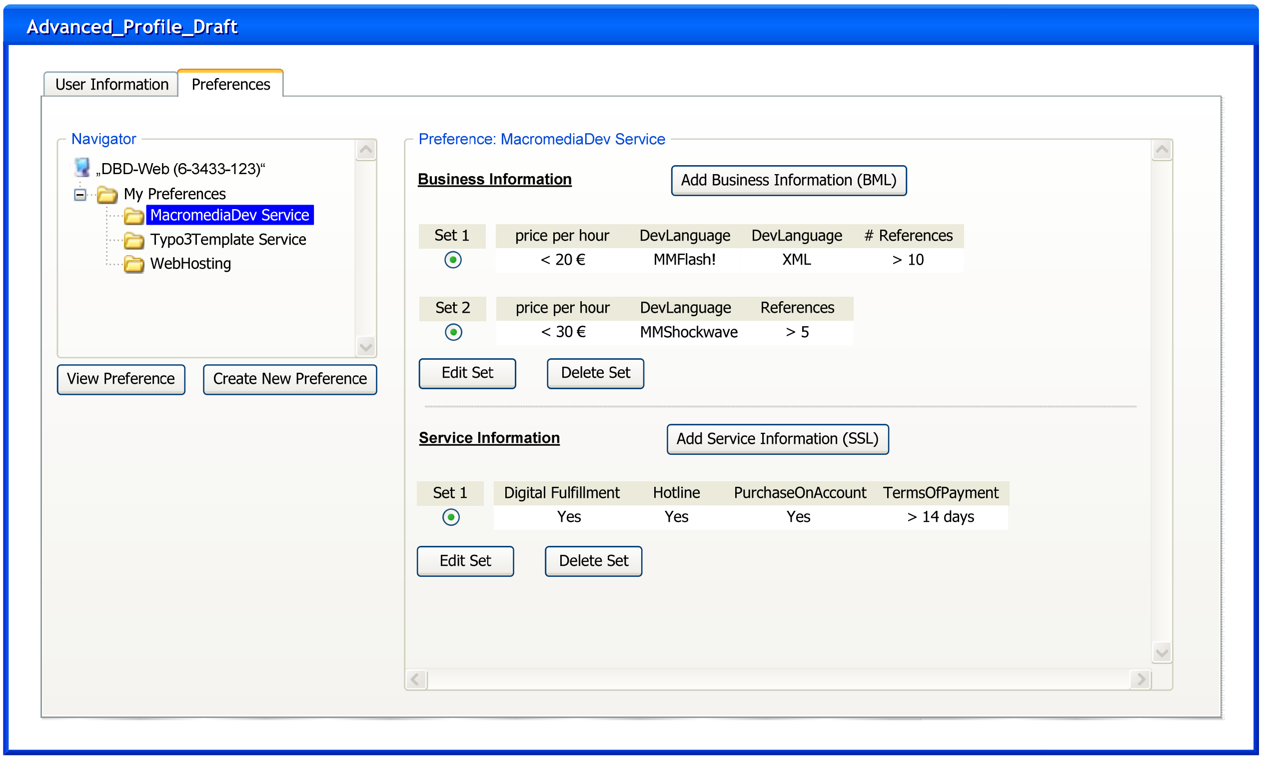


Figure 20: Scenario 2 – Final Preference Profile

Taking Preference Set 1 in Figure 20, this describes the following preference: "We prefer a company which offers a service using Macromedia Flash! and XML technology but the development per hour fee should be less than 20 Euro. Additionally the company should produce at least 10 references to show its competence."

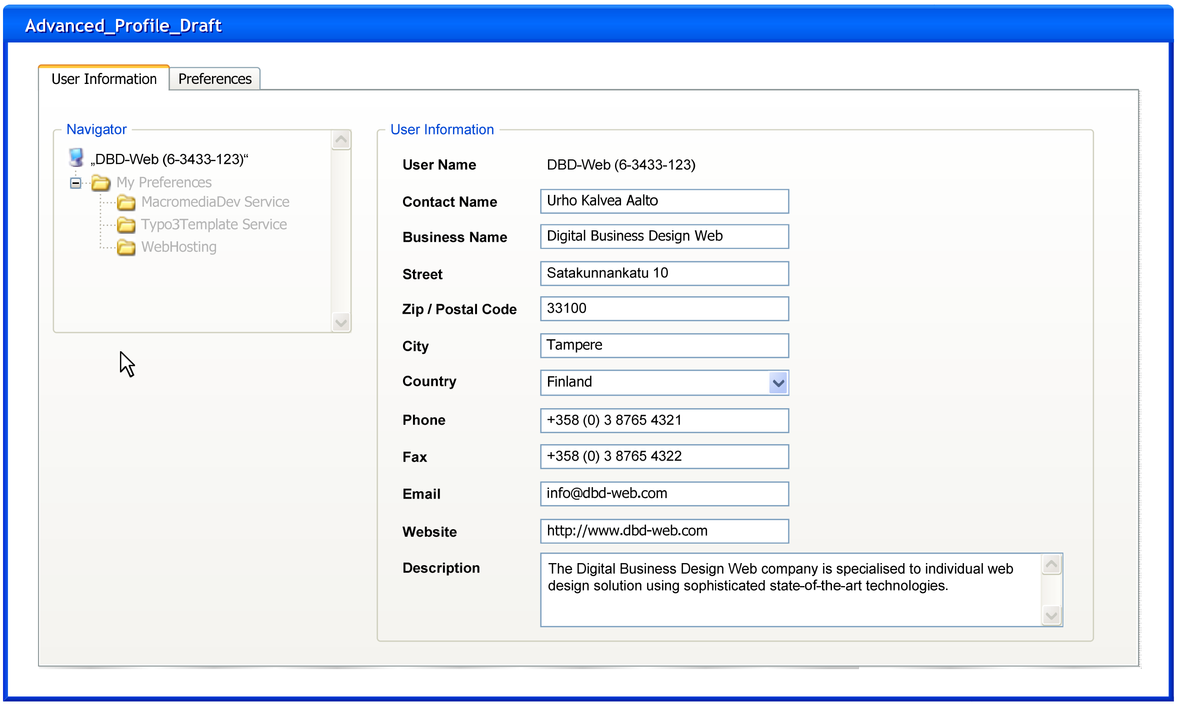


Figure 21: Scenario 2 – User Information

Based on these preferences and attributes the Recommender should be able to discover personalised services, if available. Like scenario 1, user information is provided by the company in the User Information Tab. Figure 21 gives an impression of what this information could look like.

5 Discussing Preference Structures

A general problem for the usability and the quality of the current User Profile preference information is the fact that up to now only preferences which are based on exactly the same concrete model can be found by the recommendation process. Although a user may desire to combine attributes into a preference by selecting items from different kinds of models, this is not possible at the moment. For example a user cannot create a preference 'Hotel Room' which includes a combination of a room attribute 'minibar' from Hotel Model A with an attribute 'sea view' from Hotel Model B.

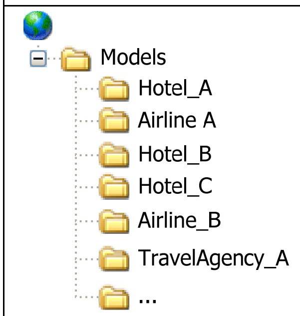
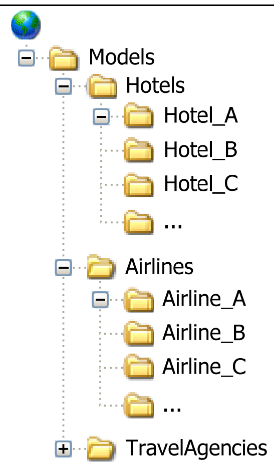
actual structure	proposed structure
	

Table 5: Current and proposed domain structure for models

Although this combination is *theoretically* possible but a preference with attributes from different models will never be matched in a recommendation process and so it is superfluous. This results from the circumstance that no lexical structure for models currently exists (see 'actual structure' in Table 5). The Recommender has to follow the exact path (pointer) to the attribute's source model to locate the model. If two attributes from different models are concatenated with, for example an AND operator, the Recommender won't be able to return a model which matches this preference. The reason for this is the lack of a tree-based structure for models as suggested in the 'proposed

structure' in Table 5 (and in this context also for the profile attributes that are a part of these models).

It still is not clear how a flexible recommendation with acceptable performance and a user friendly preference design can be provided. In the following section we discuss simple scenarios illustrating the consequences on matching quality of reusing and extending models from the model repository. This demonstrates the impact on the selected and used models when using a non-hierarchical structure. Figure 22 explains how the selection of a certain model currently influences the searching and matching process of the recommender and its results. The basic assumption is that a customer and a provider chose and use the same model from the model repository to achieve a successful matching.

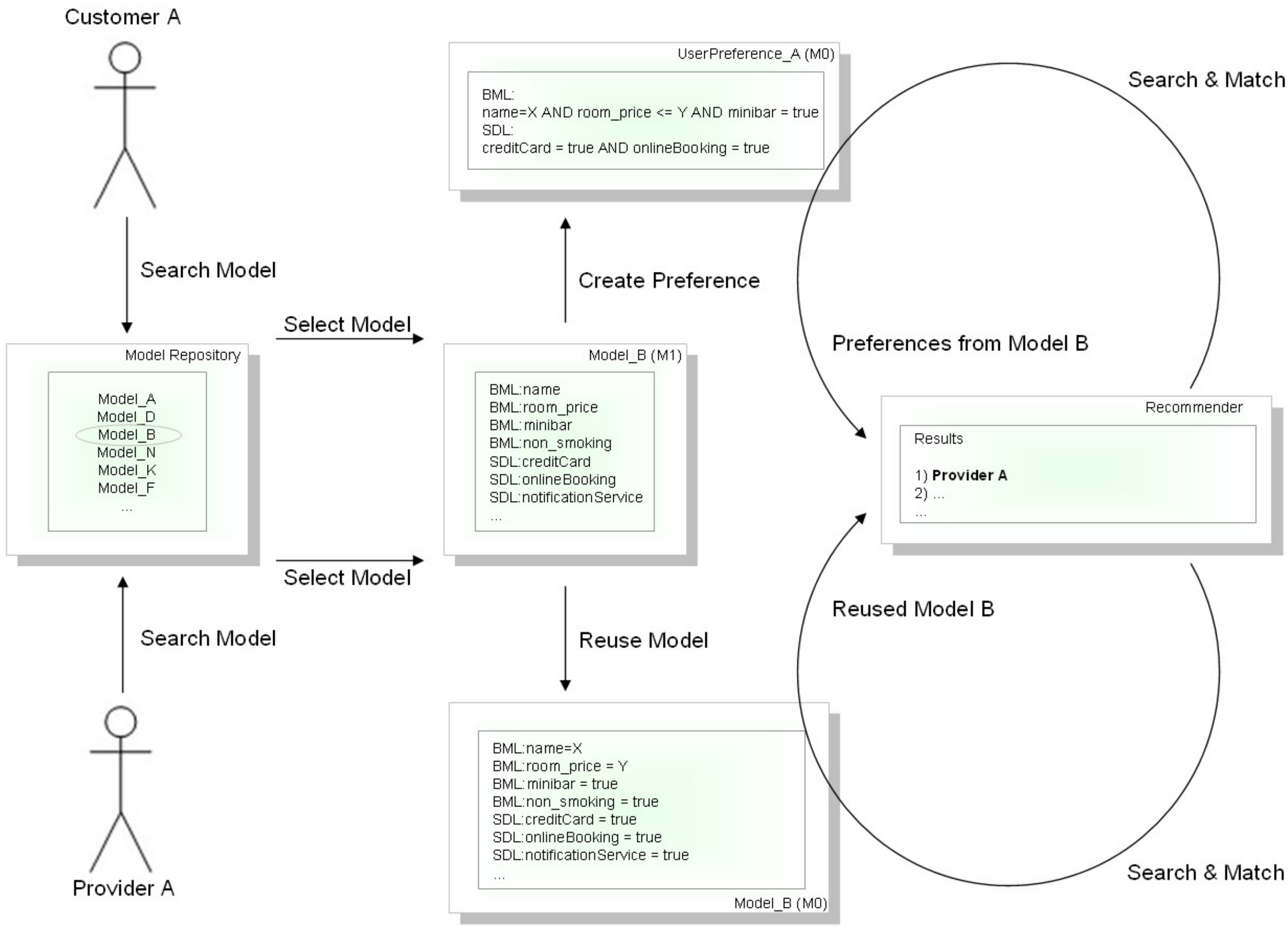


Figure 22: Matching Process using the same Source Model

If Customer A selects attributes and values from Model B and Provider A also *reuses* the existing Model B for describing his business, than the Recommender will return a match. This is because both parties use exactly the same source model for describing their purposes. All attributes used by the customer and the provider have their origin in the

same model so all attributes belong to the same semantic space. Realising a match this way is obviously quite straight forward but very restrictive.

The aforementioned case presented the results of a matching process after *reusing* an existing model and its attributes. The next case looks at the recommendation results after extending an existing model with additional attributes. Although Figure 23 looks quite similar to Figure 22 there is a small (but important) difference in that Provider B has extended the existing Model B by one more attribute (named ‘pool’) so that their service may be described completely. Personalising this Model B to their need automatically creates a new Model C. Theoretically the Model B is now a subset of Model C as all attributes a Customer A has chosen from Model B can also be found in Model C.

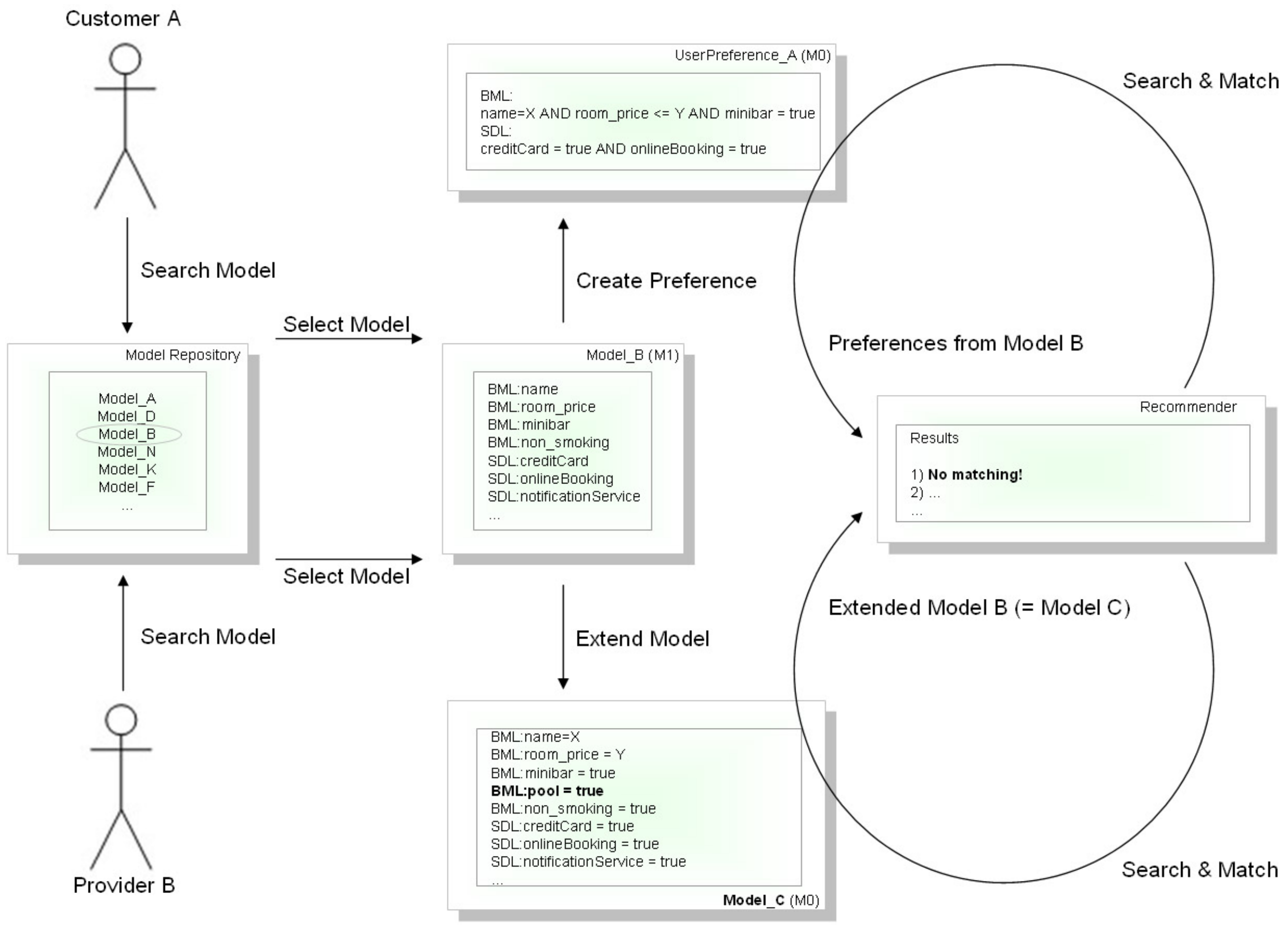


Figure 23: Matching Process using different Source Models (Example 1)

In this example, all attributes Customer A added to their profile belong to Model B and all attributes Provider B used belong to the newly created Model C. Although a room in Model B logically has the same meaning as in Model C, the Recommender won’t find any matching results as from the technical point of view these attributes are not congruent. This ontological problem comes from the equation “Model_B.room ≠ Model_C.room”.

The next Figure 24 throws light on the case where multiple providers reuse different models. Extending the aforementioned scenario, a matching of a customer's preference taken from Model B will be done against the Models B and C, both reused by diverse Providers. As expected the recommender will only find an exact match for providers who used the same model (Model B) as Customer A did.

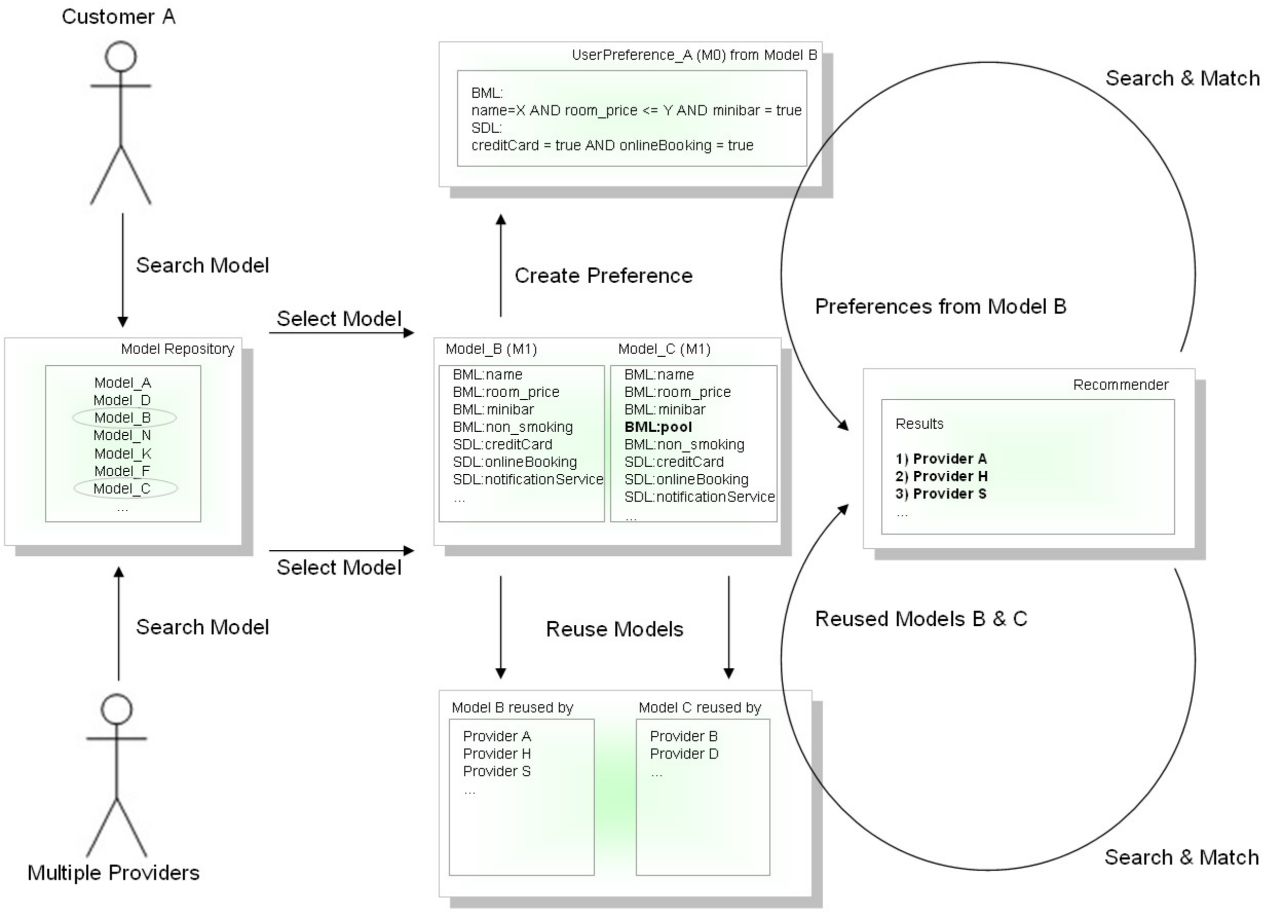


Figure 24: Matching Process using different Source Models (Example 2)

A possible solution for this lack of complete recommendations is given in Figure 25. It proposes using a Domain structure for Models. As well as the pure model a superordinate is defined which allows the models to be classified in an appropriate Opportunity Space. We are aware of the fact that the DBE architecture should be decentralised using a Peer-to-Peer network. A domain structure like the one proposed in Table 5 is not inconsistent with the decentralised approach as it is not a matter of centralising but rather maintaining a domain structure. This could easily solve the problem of the current matching process taking only identical models into account. With this proposed technique, all Models within a certain domain would be considered for matching. The example clarifies. Model B and Model C both belong to Domain A. The Recommender is now able to match all Models

within the same Domain against the customer's model as it "knows" that all Models belong to the same Opportunity Space.

Although this proposal is relatively straight forward to comprehend, the question arises who will be responsible for the creation and maintenance of such a domain structure in the future of the DBE. Conceivably, the DBE community could maintain update the list of domains. This idea has to be discussed in the near future.

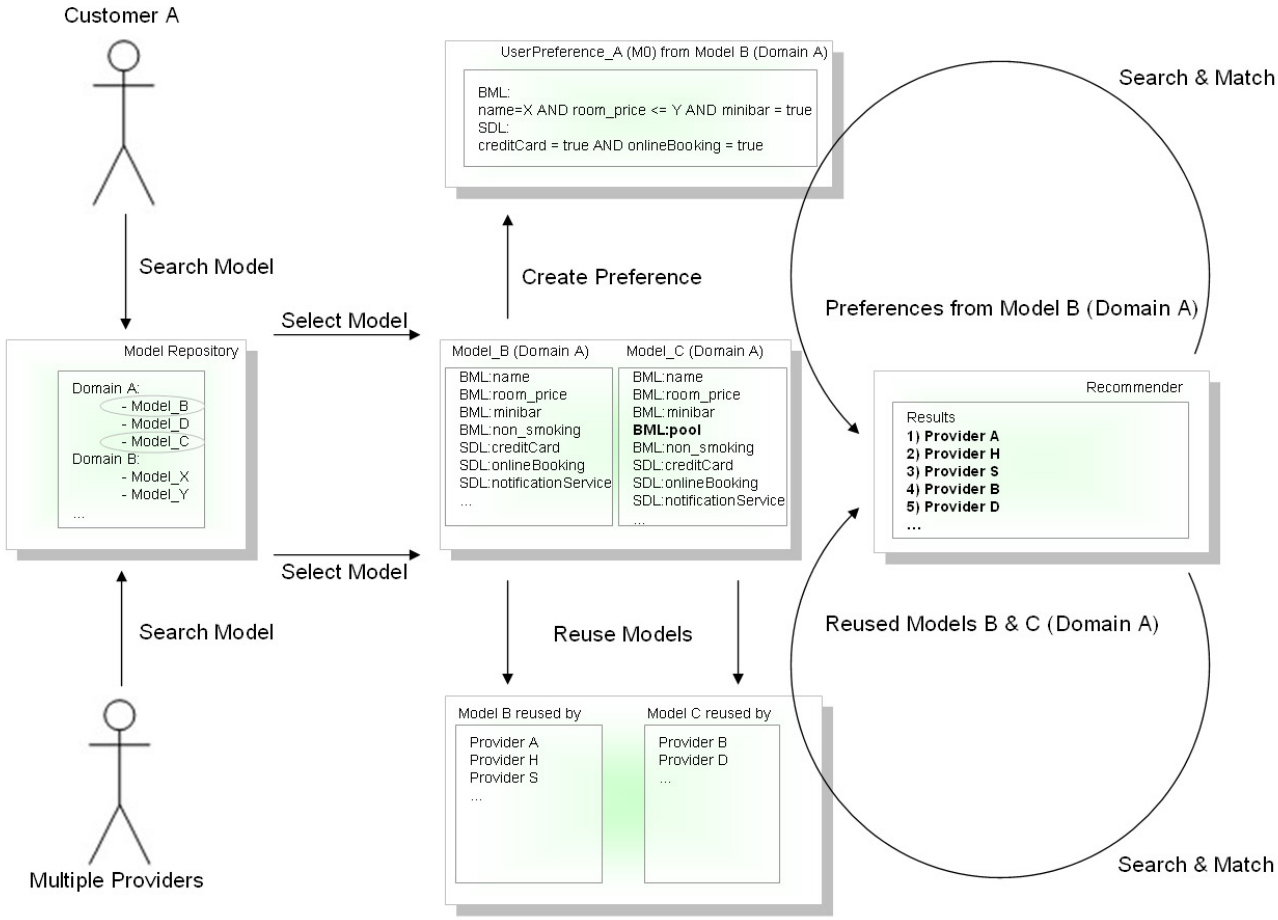


Figure 25: Matching Process using a Domain Structure

Although the current implementation (without domains) possesses advantages like high flexibility, minimal maintenance of model descriptions and possibilities for self-organisation, the further realisation (especially if the number of services increases) will require a large effort in developing Semantic Web and Ontology based approaches, something which is not in the scope of Work Package 7. It is not yet certain that this alternative approach would provide a more efficient and scaleable solution to the current structure but the DBE partner TUC are performing initial research on this topic:

6 PROFILING MECHANISMS FOR RETRIEVING USER INFORMATION

As described in the beginning of this document, we consider the term “profiling mechanism” as a composition of different relevant components working together. All components as the user friendly GUI, the generally accepted Meta Object Facility model for User Profiling purposes and selected algorithms and techniques for the User Profile functionality are important for an integrated User profiling approach. In the following we introduce two discussable possibilities for retrieving user information based on the user’s behaviour and decisions. This approach is not exhaustive but it shows basic ideas and possibilities of useful mechanisms for gathering user information in the DBE environment.

Electronic profiling is a popular topic recently, both for Internet start-ups and research efforts in the area of electronic commerce. In the rush to create profiles and make use of them, companies pay little attention to whether profiles are convenient for the user. Besides, most profiles require considerable user effort, usually in filling out online forms or questionnaires. The burden of searching out desired products, services, or information, now carried by the user, should shift to the e-service provider thanks to implicit user profiling. This section surveys and roughly discusses the possibilities to introduce behaviour-based approaches to the User Profile within the Digital Business Ecosystem. An obvious approach would be to use Log Files created by the User’s interaction with the DBE.

6.1 LOG FILES

Mosing describes a log file as “a ‘file that records the activity on a Web server’”. [Mosi03, p.8] A Log Files, which is generated nearly on every web server records information such as, which files from a web server are requested, when they are requested, who requested them, and where they were referred from. Using this data helps identifying trends but also incompatibility and other bugs in a website or a web server. The usage of specific analysis software provides information, for example, to quickly and easily view high-level trends. For further details, the report “Cookies and Log Files – The ‘Transparent Internet User’ or Data Protection on the Internet in the EU” by Max W. Mosing is recommended [Mosi03].

Internet log files can be grouped mainly into two types: web-server-side log file and client-side log file. Retrieving user specific information regarding their behaviour seems to be

difficult as most log file analysis tools are designed for processing and retrieving general information about a group of users but not about individuals (further explanation below in the text). Although client-side log files could be more useful for this purpose, the study “Characterizing Browsing Strategies in the World-Wide Web“ [CaPi95] has proved, that it is also difficult to interpret client-side log files for further purposes. In server log files, users' tasks may not be discernible. A client-side log file can record all browser events on a client and capture more detailed usage data than a server log file. Generally spoken web servers maintain log files listing every request made to the server. Further details can be found in [Andr98a].

Log File analysis tools (also called web log analysers) were developed to permit the retrieval of this data as well as some other information used for some general statistics. The software parses a log file from a web server (like Apache), and calculates many of indicators from values contained in this log file. Common indicators reported by most of web log analysers include [Wiki05b]:

- Number of visits, and number of unique visitors
- Visits duration and last visits
- Authenticated users, and last authenticated visits
- Days of week and rush hours
- Domains/countries of hosts visitors
- Hosts list
- Most viewed, entry and exit pages
- Files type
- Operating System used
- Browsers used
- Spiders or Robots used,
- Search engines, key phrases and keywords used to find the analyzed web site
- HTTP errors
- …

This huge collection of information provides a basis for further profiling activities. The more server-based interaction data is provided, the better conclusions can be drawn on behalf of the user. Combined with individual information provided by a client-side log file interesting interpretations of user behaviour could be possible. Probably all activities and techniques analysing recorded user data, need a certain amount of data to create reliable statements. Therefore a critical mass is needed to provide usable results. It is not clear how many interactions users will have with their user profiles to generate data which can be logged. This has to be evaluated in tests with SMEs using profiles in real world scenarios.

6.2 IMPLICIT PROFILING APPROACHES

As seen in section 6.1, Log files provide general statistics and information about the user's behaviour with no "individualised" information. Individual in this context means that retrieved data can be assigned to a user and his usage behaviour. The use of analyser tools and scripts (e.g. written in Perl) could help to extract specific data from each log File. However, in the best case, this could merely reveal information about a service chosen by the user as without supplying further details about the user preferences no more individualised information can be found by a simple analyser tool. An alternative would be that the recommender learns from the user's reactions such as user feedbacks and ratings. Three implicit User Profiling approaches are introduced in the following sections.

6.2.1 Feedback and Rating

In information retrieval and filtering, users are usually not able to express their information needs in exact terms. But, they can easily evaluate a document on relevancy to their information needs. [ZhSe00] calls the evaluation by a user a user relevance feedback which is used for improving retrieval and filtering accuracy. The user's evaluation could be provided in ratings (explicit ratings) which would express their opinions of recommendations they have obtained. In order to support user needs, user profiles should support mechanisms for making them adaptable according to feedback from user reaction to information provided to them, as user interests tend to change over time. Learning mechanisms for user profiling issues should be integrated to incorporate the user's feedback and rating activities into the profile's structure. Further approaches on this are to be identified.

6.2.2 Methods for handling User Profiles

This section gives an overview of the main User-Profiling Methods for the creation and update of user profiles. The methods are described concisely in [KuSh00] which will give an impression of the currently available techniques for user information retrieval:

- *User-Created Profile*
  This is the most simple and natural approach. The user specifies his/her area(s) of interest by a list of (possibly weighted) terms. The specified terms are used to guide the filtering process.

- *System-Created Profile by Automatic Indexing*
  A set of data items which have already been judged by the user as relevant, are analysed by software, in order to identify the most frequent and meaningful

concepts in the text. Those concepts, weighted according to the frequency of their appearance, constitute the user profile (see also section 6.2.3).

- *System- plus User-Created Profile*
  This is a combination of the above two approaches. First, an initial profile is created automatically (by automatic indexing). Then, the user reviews the proposed profile and updates it (by adding or deleting terms, and changing their weights).

- *System-Created Profile based on Learning by Artificial Neural-Network (ANN)*[9]
  Based on a sample set of data items that have already been judged relevant by the user, an ANN may be trained. The inputs of the ANN are the meaningful terms, and the outputs are the relevance judgments of the users. An algorithm can calculate a Causal Index that gives the relative magnitude (and sign) of the influence of each input on each output. After training, the ANN may serve as the user profile for future filtering.

- *User-Profile Inherited from a User-Stereotype*
  This method assumes that the “system” has pre-defined user-stereotypes. A user-stereotype is represented as a content-based profile, i.e. a weighted-vector of terms or even a weighted tree that represents a set of (virtual) users who have common information usage and filtering behaviour. A user-stereotype is also represented by a set of demographic and social attributes that are common to those users. (It is beyond the scope of this paper to describe how user-stereotypes and their profiles are created.) A new user is attached to a predefined stereotype to which he/she is most close with respect to the demographic and social attributes.

- *Rule-based Filtering*
  All previous methods deal with the creation of a content-based profile. Contrarily, a rule-based profile consists of a set of filtering rules. Questioning the user on his/her information usage and filtering behaviour can generate such rules. An alternative method for creating a rule-based profile for a user is to inherit filtering rules from user-stereotypes, similar to the inheritance of a content-based profile. As before, this method assumes that the “system” has pre-defined user-stereotypes, but in this case a user-stereotype is represented by a set of filtering-rules that are common to the users who belong to the stereotype. User-stereotype is also represented by a set of demographic and social attributes that are common to those users. A new user is attached to a predefined stereotype

[9] Further information is available under: http://en.wikipedia.org/wiki/Artificial_neural_network

6.2.3 Learning User Preferences

To learn about user preferences many approaches have resorted to intelligent algorithms which improve the user model during his/her interaction with the system. An example for this method is the LMS (Least Mean Square) approach initially introduced by Widrow and Hoff in 1959 [Shet05]. The modified algorithm searches the weighted vectors representing the user information, receives the user's feedback and uses a learning rate to control the learning speed. Terms whose weight is lower than the threshold are removed from the user's profile. An interesting approach is an information filtering system called WAIR (Web Agents for information Retrieval) [ByYo01] which introduces methods for learning preferences by analysing browsing behaviours implemented in a Web-based personalised information filtering system. This is a profile based web crawling platform for personalised information services on the Web. There are mainly three agents in the platform: a user interface agent keeps track of user behaviour, a retrieval agent to construct query depends on the user profile to retrieve web pages; and a filtering agent to evaluate the relevance of web pages and present a particular number of pages to the user.

Other concepts like Similarity functions, hierarchical clustering methods and Threshold-finding methods have been used to create a more robust context for personalisation and a more efficient user model. The latter for example could enable the system to automatically draw assumptions about the user beliefs, goals and plans. Researches in this field lead to some intelligent recommender systems like "GRUNDY" [Rich79] which plays the role of a librarian and recommends books on the basis of its assumptions about the user's personal characteristics.

Such characteristics include, for example, the user's educational and intellectual level, preference for thrill, fast moving plots or romance, tolerance for descriptions of sexuality, violence and suffering, etc. All characteristics are represented by simple linear scale values with associated certainty ratings.

Other instances are: The "VIE-DPM" system [Kobs85a, Kobs85b] is part of the German-language dialog system "VIE-LANG" [TBST83] and is based on a complex representation system for storing a wide range of beliefs and goals. The "HAM-ANS" system [HoMM84], when run in the hotel reservation mode, has standard assumptions about what every user knows about hotel rooms, and keeps track of what it has told the user about the specific room it is offering him/her. The prototype system, "LIBRA" (Learning Intelligent Book Recommending Agent) which uses a database of book information has content information about titles consists of textual meta-data rather than the actual text of the items themselves. Users provide 1–10 ratings for a selected set of training books; the system then learns a profile of the user using a Bayesian learning algorithm and produces a ranked list of the most recommended additional titles from the system's catalogue.

User modelling in dialog systems is doubtless still in its infancy. Research in this field is just now spreading from university labs to industry. No commercial system embodying a user model has been developed to date, and it is expected that in the next few years commercial systems will perform only a very restricted and simple kind of user modelling. As well as this, such an approach can only generate limited success, when applied to a specific area like the case of the systems presented above. The Digital DBE, on the other hand, has a huge range of services embedded within a theoretically infinitely huge Opportunity Space consisting of many domains. The feasibility for developing individual methods for each domain (e.g. one for Tourism, another one for Web Developers etc.) is very low for flexibility and extension reasons. That's why the DBE would require one universal method for user profiling. It's not clear if this problem can be solved with one super-algorithm, and that's why this approach seems not convenient and manageable for the DBE.

6.3 Collaborative Filtering and Rule Discovery

6.3.1 Collaborative Filtering

The collaborative filtering approach "is based on the assumption that people belonging to a particular group tend to behave similarly, given a set of circumstances. This approach is thus used when a user's behaviour is predicted from the behaviour of like-minded people" [Joha02].

It maintains a database of user preferences, gets inspired from other users with similar preferences and provides recommendations to a user approved by their matched patron. This approach requires a sufficient number of user ratings.

User ratings and feedback seem to be the only information source within the DBE so far. Therefore the collaborative filtering approach appears to be a reasonable approach to generate other user preferences in addition to the ones explicitly given by the user. Based on these preferences, for example, profile matching techniques could be provided by the recommender as this one is the central interface to all user profiles available in the DBE environment. One difficulty commonly faced thereby, is the cold-start problem where recommendations are required for new items or users for whom little or no information has yet been required. Central in this case doesn't mean a centralised architecture but more the central component for networking user profiles in the decentralised DBE architecture.

6.3.2 Rule Discovery

Another approach could be the usa of rule discovery techniques. They are used to „train“ user profiles on behalf of transactional user histories. They are applied individually to each user's transactional data using data mining algorithms such as, for example, the "*Apriori*" Algorithm [AMS+96] to capture the personal behaviour of users. An example for the usage of rule discovery could be:

Whenever customer X goes on a business trip to Los Angeles, he stays in expensive hotels or "whenever customer X buys tomatoes, X usually also buys soap… It is not clear if this rule captures a truly causal link between tomatoes and soap inherent to X's purchasing behaviour, or it exists because X tends to do really large shopping periodically and tends to buy everything together, including tomatoes and soap. Therefore, after data mining methods discover rules describing individual user behaviour in personalization applications, it is important to validate these rules. […]. One of the main problems is how to perform post-analysis of the discovered rules, i.e., how to validate customer profiles by separating 'good' rules from the 'bad'" [AdTu99].

[AdTu99] introduced some rule validation operators using an iterative profile building process. Moreover, a structure is proposed for classifying user profile information into two components: the factual profile and the behavioural profile as mentioned in the current User Profiling approach in DBE. They also considered that the "costs" of such an approach seem to be very high and have to be taken into account when using rule discovery techniques.

7 CONCLUSIONS

The first User Profile Model provides basic functionality in storing and editing the profile in a comfortable way. The design draft of the Graphical User Interface gives an impression of how a user would use the profile as seen in the two scenarios described.

This paper also introduced and discussed mechanism for constructing implicit behavioural profiles of the Digital Business Ecosystem users and the appropriate challenges. More research has to be done on adapting existing approaches for DBE purposes. Finding a universal mechanism or approach seems to be extraordinarily costly if the premise of "high functionality" should not be harmed. Attempts updating a user profile according to its interaction with the system, needs some kind of 'intelligence' which can only be provided when focusing on restricted domains or kinds of services. Recorded user feedback has been shown to be an interesting source of additional information to the explicit user profile. These stored opinions bring up the possibility (which has to be evaluated) of using a collaborative filtering approach by the recommender. Further work should be done on that issue.

Additionally, the overall restriction in any automated information retrieval process is the fact that a certain number of user interactions and even at least a certain number of users, the so-called critical mass, is necessary to gain enough and meaningful information to be interpreted. It is not clear, if the required user interaction with their profile within the system will be provided in the project. In a later stage the possibilities and chances of information retrieval using a user history will be checked and evaluated in cooperation with relevant DBE partners.

The proposed domain-structure in chapter 5 could provide clarity to which domain a models belongs to. On the on hand a user browsing services within the User Profile (using the adapted Query Formulator Tool) can expect a high usability and on the other hand recommendations could be done more efficient. The maintenance und updating of such a structure has to be discussed as this could be a problem within a distributed DBE architecture.

8 REFERENCES

[AdTu99] Adomavicius, G.; Tuzhilin, A.: User Profiling in Personalization Applications through Rule Discovery and Validation, Proceedings of the 5th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 377-381, San Diego, USA, 1999.

[AMS+96] Agrawal, R.; Mannila, H.; Srikant, R.; Toivonen, H.; Verkamo, A.I.: Fast discovery of association rules. In Advances in Knowledge Discovery and Data Mining, ch. 12. AAAI Press, 1996.

[Andr98a] Andrews, J. H.: Theory and practice of log file analysis. Technical Report 524, Department of Computer Science, University of Western Ontario, 1998.

[Bart05a] Bartsch, C.: Milestone 7.2 – User Profiling 1st Release Spec. DBE Project, 2005.

[Bart05b] Bartsch, C.: Deliverable 7.1 - Description of necessary information about DBE customer, to support a long term evolutionary business relationship. DBE Project, 2005.

[ByYo01] Byoung-Tak, Z.; Young-Woo, S.: Personalized Web-Document Filtering Using Reinforcement Learning. In: Applied Artificial Intelligence,Vol.15(6), pp. 665-685, 2001.

[CaPi95] Catledge, L.D., Pitkow, J.E.: Characterizing Browsing Strategies in the World-Wide Web. http://www.igd.fhg.de/archive/1995_www95/proceedings/papers/80/userpatterns/UserPatterns.Paper4.formatted.html. Called on 17/09/2005.

[Ferr05] Ferronato, P.: Deliverable 21.1 – DBE Architecture Requirements. DBE Project, 2005.

[HoMM84] Hoeppner, W.; Morik, K.; Marburger, H.: Talking it over: the Natural-Language Dialog System HAM-ANS. Report ANS-26, Research Unit for Information Science and Artificial Intelligence, University of Hamburg, 1984.

[Joha02] Johansspn, P.: User Modelling in Dialog Systems. http://www.ida.liu.se/~ponjo/downloads/papers/johansson_sar2002.pdf, Called on 01/10/2005.

[KaKK05] Kazasis, F.; Kotopoulos, G.; Kotopoulos, Y.: Deliverable 17.1 – Recommender. DBE Project, 2005.

[Kobs85a] Kobsa, A.: Benutzermodellierung in Dialogsystemen. Springer, Berlin, 1985

[Kobs85b] Kobsa, A.: VIE-DPM: a User Model in a Natural-Language Dialogue System. In: J. Laubsch, ed.: GWAI-84, 8th German Workshop on Artificial Intelligence. Springer, Berlin, 1984.

[Kobs99] Kobsa, A.: User modeling in Dialog systems: potentials and hazards. In: AI & Society, Volume 4, Issue 3, 1990.

[Kowa97] Kowalski, G.: Information Retrieval Systems - Theory and Implementation. Cluwer Academic Publishers, Boston, 1997.

[KuSh00] Kuflik, T.; Shoval, P.: Generation of user profiles for information filtering - research agenda (poster session), Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 313-315, Athens, Greece, 2000.

[Liu03] Liu, L.: Log File Analysis. http://wwwcs.uni-paderborn.de/cs/ag-szwillus/lehre/ws03_04/WebUE/liu/log_file/index.html, Called on 23/07/2005.

[Moon00] Mooney, R.J.: Content-Based Book Recommending using Learning for Text Categorization. In: Proceedings of the fifth ACM conference on Digital libraries, pp. 195-204, San Antonio, Texas, United States, 2000.

[Mosi03] Mosing, M.W.: Cookies and Log Files: The “Transparent Internet User” or Data Protection on the Internet in the EU?! http://www.it-law.at/papers/mosing_log-files_cookies_english.pdf, Called on 25/08/2005.

[Niel93] Nielson, J.: Usability Engineering. Academic Press, London, 1993.

[Niel00] Nielson, J.: Designing Web Usability. New Riders Publishing, Worcester, 2000.

[PAGK04] Pappas, N.; Anetis, G.; Gioldasis, N.; Kazasis, F.: WP14: DBE Knowledge Base – Implementation Status. DBE Project, 2004.

[Quei05] Alves des Queiroz, I.: Milestone 20.2. DBE Project, 2005.

[Rath05] Rathbone, N.: Understanding the DBE Codes. https://bml.digital-ecosystem.net/files/documents/13/186/file_186.dat?filename= Training%20presentation%2eppt. Called on 07/07/2005.

[Rich79] Rich, E.: User Modelling via Stereotypes. In: Cognitive Science 3, pp. 329-345, 1979.

[Sche94] Scheer, A-W.: Architecture for Integrated Information Systems – Foundations of Enterprise Modelling. Springer, Berlin, 1994.

[SEDS05] Seigneur, J-M.; Edmonds, A.; Dowling, J.; Sacha, J.: Peer-to-peer Architecture for DBE, Version 0.5. DBE Project, 2005.

[ShLi01] Shearin, S.; Liebermann, H.; Intelligent Profilling by Example. In: Proceedings of the 6th international conference on intelligent user interfaces, pp. 145-151, Santa Fe, New Mexico, United States, 2001.

[Shet05] Shetty, K.K.: A novel algorithm for uplink interference suppression using smart antennas in mobile communications. http://etd.lib.fsu.edu/theses/available/etd-04092004-143712/unrestricted/Ch_6lms.pdf. Called on 10/09/2005.

[TBST83] Trost, H.; Buchberger, E.; Steinacker, I.; Trappl, R.: VIE-LANG: A German Language Dialogue System. Cybernetics and Systems 14, pp.343-357, 1983.

[WaKl04] Warmer, J.; Kleppe, A.:Object Constraint Langauge 2.0. mitp-Verlag, Bonn, 2004.

[Wiki05a] Wikipedia: EPC. http://en.wikipedia.org/wiki/Event-driven_process_chain. Called on 21/08/2005.

[Wiki05b] Wikipedia: Web log analysis software. http://en.wikipedia.org/wiki/Web_log_analysis_software. Called on 10/10/2005.

[ZhLY03] Zhong, N.; Liu, J.; Yao, Y.: Web Intelligence. Springer, Berlin, 2003.

[ZhSe00] Seo, Y-W.; Zhang, B-T.: A reinforcement learning agent for personalized information filtering. In: Proceedings of the 5th international conference on Intelligent user interfaces, pp. 248 - 251, New Orleans, USA, 2000.