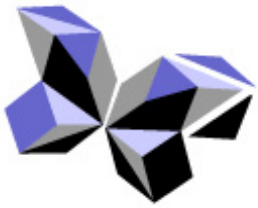


Digital Business Ecosystem

Contract n° 507953

# WP 7: User Profiling

## Del 7.4: Report About Profiling Mechanism Implemented and Lessons Learnt

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°**: 7.4

**Due date**: 31/01/2007

**Delivery Date**: 13/02/2007

**Short Description**:

The deliverable provides the final implementation features for the User Profile software provided to the DBE Studio and DBE Portal. It contains a tutorial how to generate User Preferences and how to use the User Profile in general. In order to round off the work it concludes with the "Lessons Learnt".

**Partners owning**: FZI

**Partners contributed**: FZI

**Made available to**: DBE Partners and European Commission

**VERSIONING**

| VERSION | DATE | AUTHOR, ORGANISATION |
|---|---|---|
| **0.1** | **15/09/2006** | **CHRISTIAN BARTSCH (FZI)** |
| **0.2** | **10/10/2006** | **CHRISTIAN BARTSCH (FZI)** |
| **0.3** | **27/12/2006** | **CHRISTIAN BARTSCH (FZI), MANUEL ODENDAHL (FZI)** |
| **0.4** | **29/01/2007**<br>**29/01/2007** | **ANDY EDMONDS (INTEL) – 1ST INTERNAL REVIEWER**<br>**DOMINIK DAHLEM (TCD) – 2ND INTERNAL REVIEWER** |
| **1.0** | **13/02/2007** | **CHRISTIAN BARTSCH (FZI) – FINAL VERSION** |

**Quality check**

**1st Internal Reviewer**: Andy Edmonds (INTEL)

**2nd Internal Reviewer**: Dominik Dahlem (TCD)

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# 1 INTRODUCTION

The aimed benefit of a User Profile is to personalise the user's needs more effectively and efficiently, to make interactions faster and easier and, consequently, to create user satisfaction and the likelihood of repeat visits [Scrm03]. The overall objective of the User Profile in the DBE is to represent the real world user's preferences in a software-based User Profile. This means that information about a user should be stored within a User Profile as suitable and as fine grained as possible.

Depending on the technical understanding and the industrial sector a person comes from, the term User Profile has different meanings to the people. Most people know User Profiles from common and very simple internet applications or portals. With these they can only add some general information to describe themselves and provide information like name, address, telephone number or similar. Others understand the term very technically as a detached sophisticated algorithm which performs specific methods and functions to collect, agglomerate and create information which might be the basis for further activities. Another interpretation of the term User Profile has been created for this project as an assembled User Profile approach is not available in this form so far. Naturally a User Profile in the DBE contains basic information about the users themselves which is well known from other applications. In addition to this users are able to represent their preferences by creating Preference Sets. This means that users create individual expressions using a set of attributes which mirror best their real intentions and preferences related to certain issue.

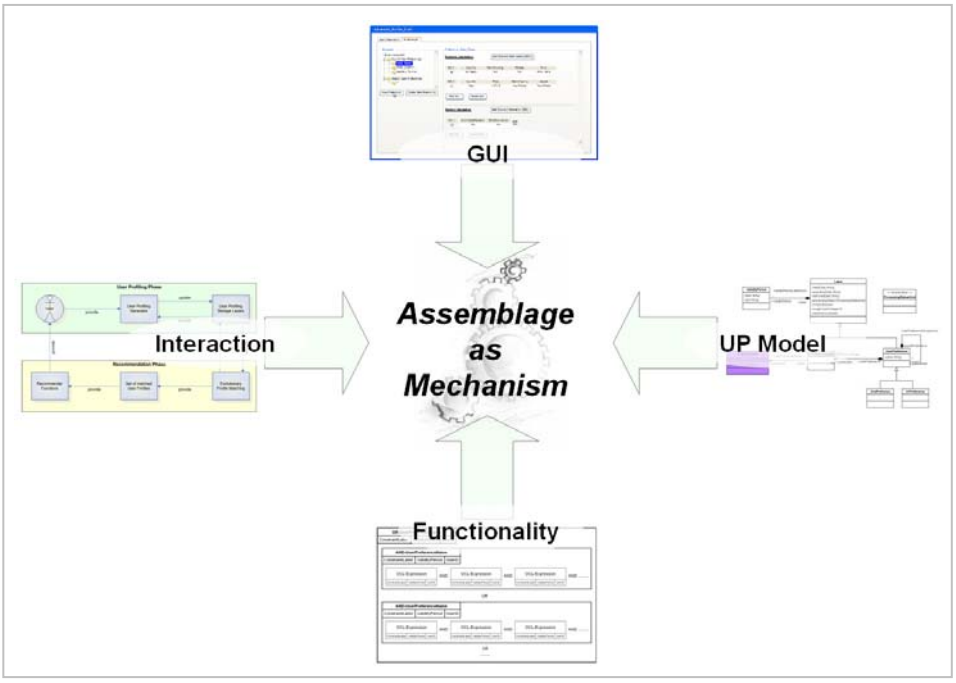



*Figure 1: Profiling Mechanism: Assemblage Interpretation*

A User Profile can consist of arbitrary many different kinds of information like demographical data, usage history etc. The focus in this project was put on the term User Profile as an “assemblage mechanism”. This creates and maintains User Preferences as the user builds the core information source on what the user’s character is represented. This approach required a rather holistic approach for the design and realisation of the User Profile project than just a single algorithm. The interpretation and components of a User Profile in the context of the DBE project is shown in Figure 1.

Referring to this figure, a mechanism is *not* a pure technical aspect using algorithms and code, but more an assemblage between important parts which are necessary to create and maintain a User Profile and the stored User Preferences. Therefore it is also possible to consider an intuitive Graphical User Interface (GUI) to be a part of a good mechanism or an adequate and generally accepted User Profile Metamodel (UPM) [Bart05b, pp.10]. 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.

This deliverable presents the final User Profile design and User Profile implementation.

Chapter 2 shows and illustrates the technical features implemented in the User Profile application based on the aforementioned User Profile Metamodel. Based on these features, the following chapters describe two User Profile applications. Chapter 3 reflects the implementation of the User Profile for the DBE Portal [Admo06] as well as for the Eclipse Plug-in [Bart06, pp. 13] for the DBE Studio [McKi06]. After that a small tutorial in Chapter 4 will show how to use the User Profile for the DBE Portal and the DBE Studio in practice. Chapter 5 provides some Lessons Learnt in order to round off the work of work package 7.

# 2 USER PROFILE - TECHNICAL FEATURES

## 2.1 USER PROFILE METAMODEL (UPM)

The User Profile Metamodel builds the basis for representing User Profile data in an adequate and compatible format in the context of this project. As described in Deliverable D7.2 [Bart05b] the infrastructure designs of the Knowledge Base (KB) follows the OMG's[1] Metadata Object Facility (MOF) approach [GiKM05, pp. 14] [MOF06]. This is why a Metamodel for a User Profile as shown in Figure 2 is necessary to integrate with the KB Peer Architecture [PaKK06].

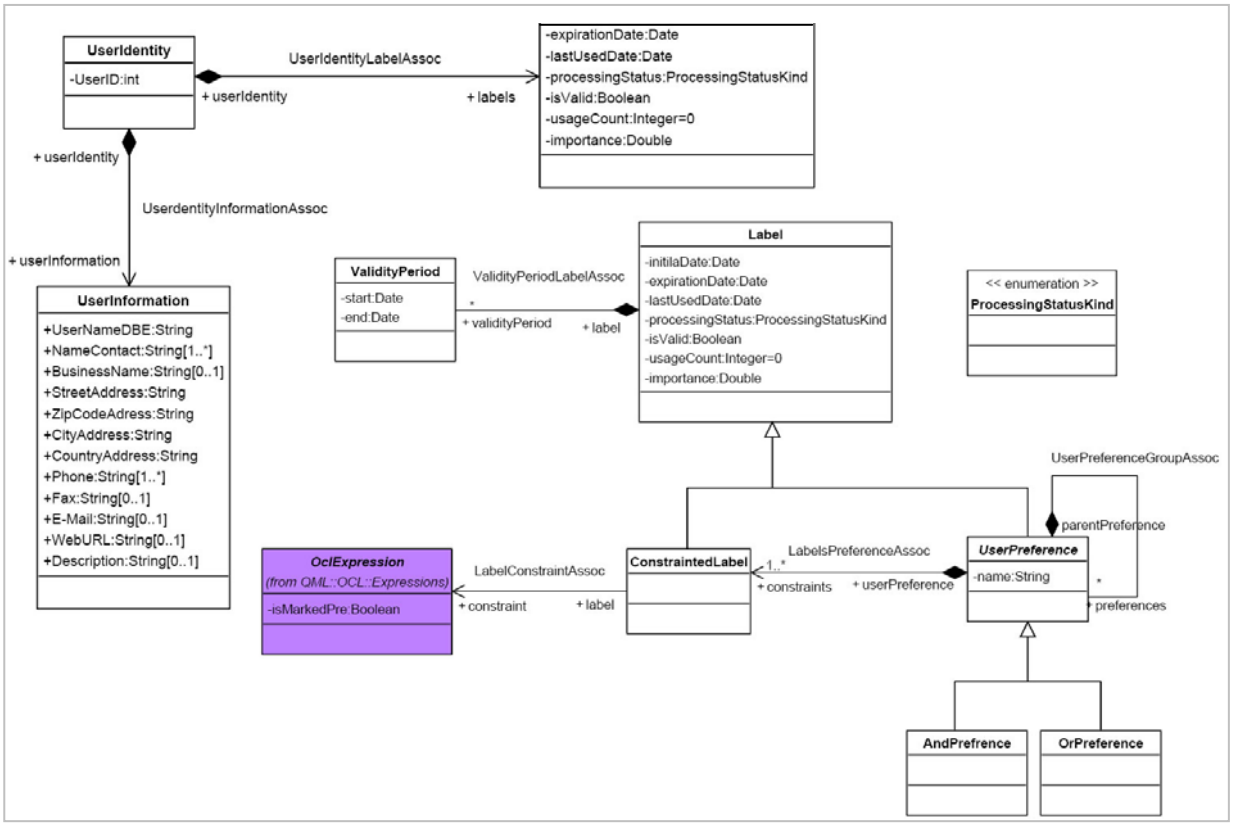



*Figure 2: User Profile Metamodel*

An extensive description for the attributes and their values modelled in the User Profile Metamodel can be looked up in D7.2.

[1] Object Management Group (http://www.omg.org)

## 2.2 UML SEQUENCE DIAGRAMS

The following sections describe the implemented functionality, which is implemented in both the DBE Studio and the DBE Portal, by the User Profile application. The User Profile's actions are illustrated through UML[2] interaction diagrams [UML05], in particular sequence diagrams, which emphasise the flow of control and data. The main processes "behind" the GUI retrieve and store the User Profile, search and retrieve relevant Business Models (BML Models) encoded with the Business Modelling Language [DeTo05]. They finally search and retrieve DBE Services depending on the respective User Preferences.

### 2.2.1 Retrieve User Profile

The retrieval of the User Profile Model is triggered automatically after the log-in process. Identified by the User ID, the User Profile Model (UPM) is requested by the Recommender Service (RC_Service) [KoKK06] which is part of the Knowledge Base Infrastructure. The UPM is retrieved from the Knowledge Base as seen in Figure 3.

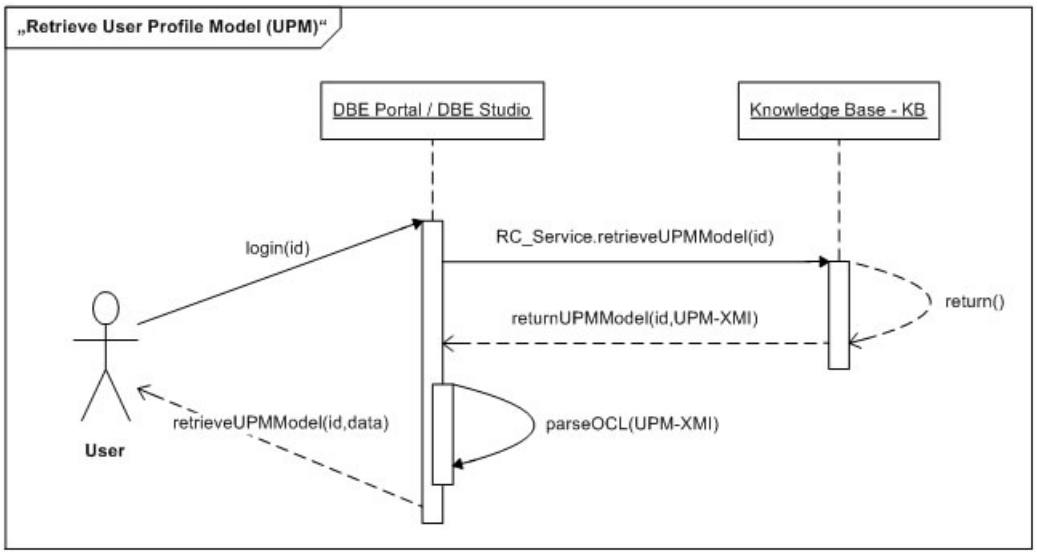



*Figure 3: UML - Retrieve User Profile*

As the retrieved User Profile Model is represented as a XML Metadata Interchange (XMI) [XMI05] file (containing OCL[3]-statements [OCL06] which encode the User Profile Preferences), it needs to be parsed into XML-files to extract the relevant User Profile information that is finally shown to the user.

[2] Unified Modelling Language (www.uml.org)
[3] Object Constraint Language

### 2.2.2 Store User Profile

The procedure for storing the User Profile into the Knowledge Base is very similar to the retrieval process shown in section 2.2.1. The XML representation of the front-end User Profile is converted into OCL statements which are integrated into the User Profile Model XMI file.

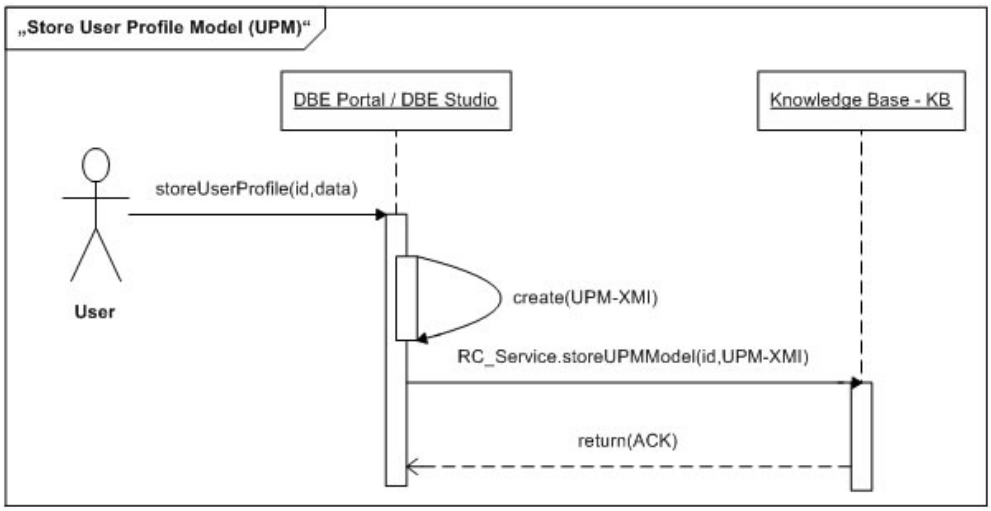



*Figure 4: UML - Store User Profile*

Figure 4 illustrates that this XMI file is sent via the RC_Service to the Knowledge Base where the UPM-XMI file is finally stored. A short acknowledge is returned to the user informing that the storage was successful.

### 2.2.3 Search & Retrieve BML Models

The search for BML Models is an essential activity as these models are the basis for selecting adequate attributes provided by the BML models.

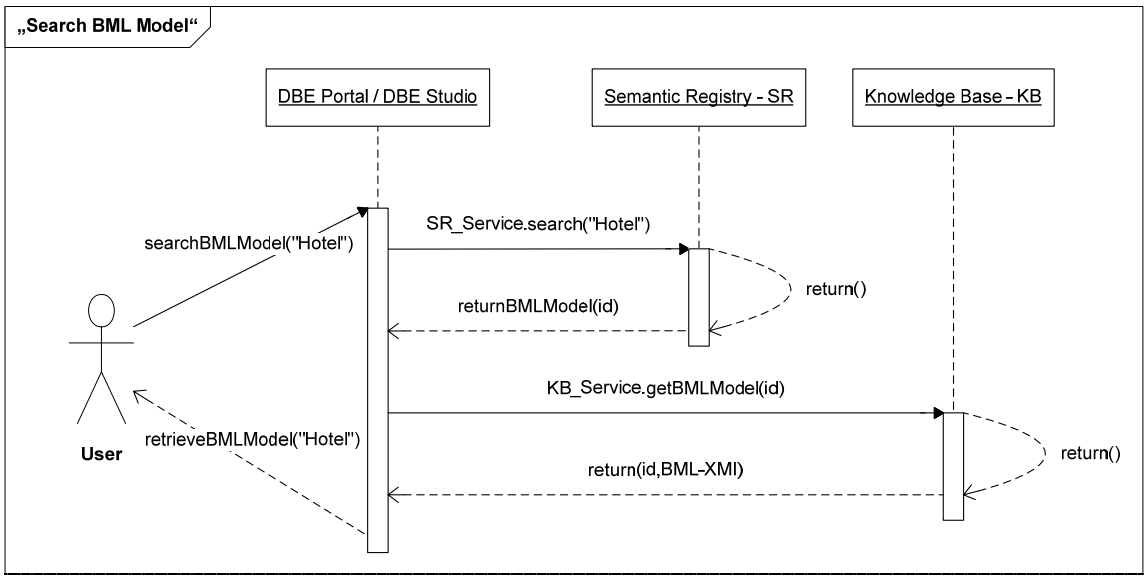



*Figure 5: UML - Search & Retrieve BML Models*

As shown in Figure 5 the users enter several keywords and send the query via the Semantic Registry Service (SR_Service) [Ferr05] to the Semantic Registry[4]. The registered DBE Services are compared with the sent keywords. If the keyword matching process is successful, the identifiers of the potential DBE Services are passed to the Recommender Service (RC-Service) which retrieves the corresponding BML Models from the Knowledge Base. The BML Models found are shown to the user.

### 2.2.4 Manual-Search for DBE Services

The search for DBE Services in the context of the User Profile is guided by User Preferences created by the user. Figure 6 shows how the users select the Preference Sets they want to add to the query. This query is converted into OCL statements which are sent directly to the Semantic Registry using its service.

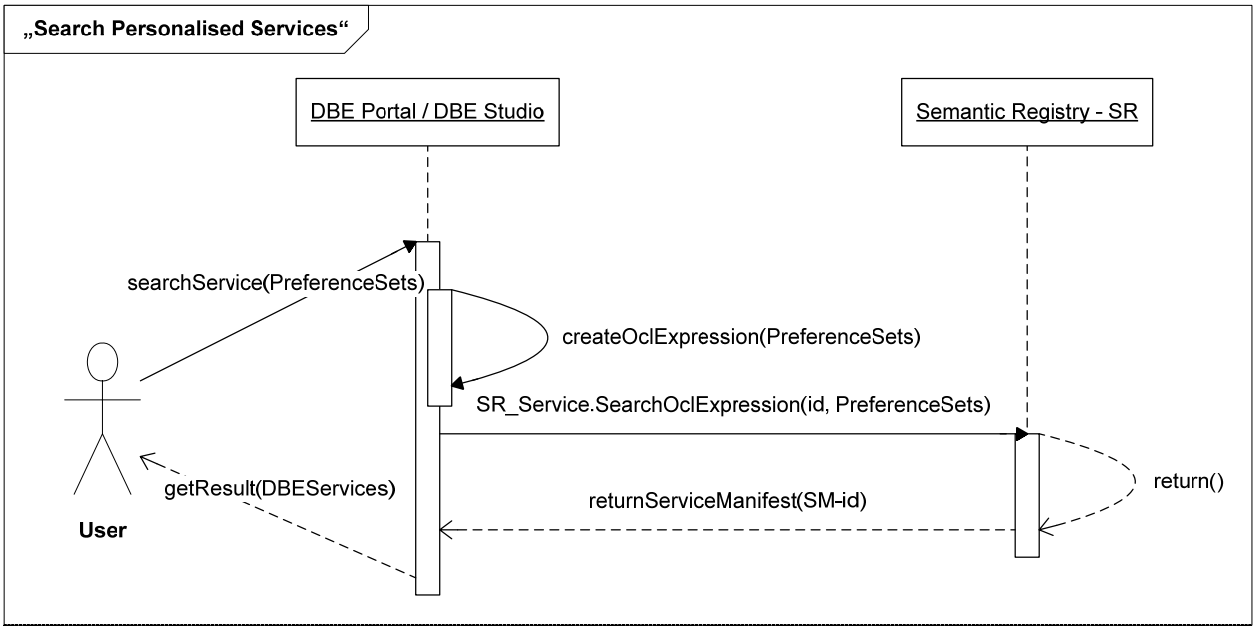



*Figure 6: UML – Manual-Search for DBE Service*

Depending on the matching results with the DBE Services available in the Semantic Registry, the SR_Service returns potential DBE Service Manifests to the user. With these manifests the user can decide which DBE Service fits best to their needs. The pre-selection of potential DBE Services based on User Preference Sets is a useful feature to support the user in filtering and selecting useful DBE Services to their personal needs.

[4] The Semantic Registry is the component of the DBE Knowledge Base that hosts the published services.

### 2.2.5 Auto-Search for DBE Services

The auto-search mechanism as shown in Figure 7 implies that potential DBE Services are collected and pushed to the user automatically by the Recommender Service as soon as the user retrieves the User Profile.

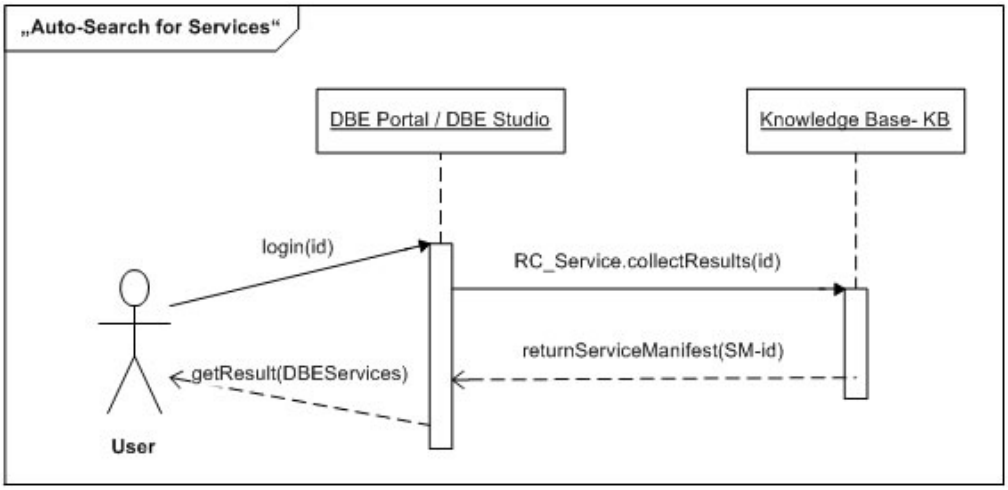



*Figure 7: UML - Auto-Search for DBE Services*

The Recommender Service matches all the Preference Sets from a User Profile with the available DBE Services and “suggests” potential Service Manifests corresponding to the User Profile Preference Sets.

# 3 USER PROFILE IMPLEMENTATION

This chapter describes the implemented User Profile and its functionality in detail. More than 30 buttons, labels, text fields and functions have been implemented within the applications. The following sections illustrate and document every part of the User Profile project. The first section 3.1 describes the User Profile implementation for the DBE Portal whereas section 3.2 shows a similar implementation for the DBE Studio. The resources of both applications can be found in the Appendix 6.1.

## 3.1 USER PROFILE FOR THE "DBE PORTAL"

### 3.1.1 Graphical User Interface

The development of the OpenLaszlo based User Profile GUI consists of a simple Log-In View (see Figure 18) which allows access to the User Profile and the main User Profile frame including the four implemented windows (views) as illustrated in Figure 8:

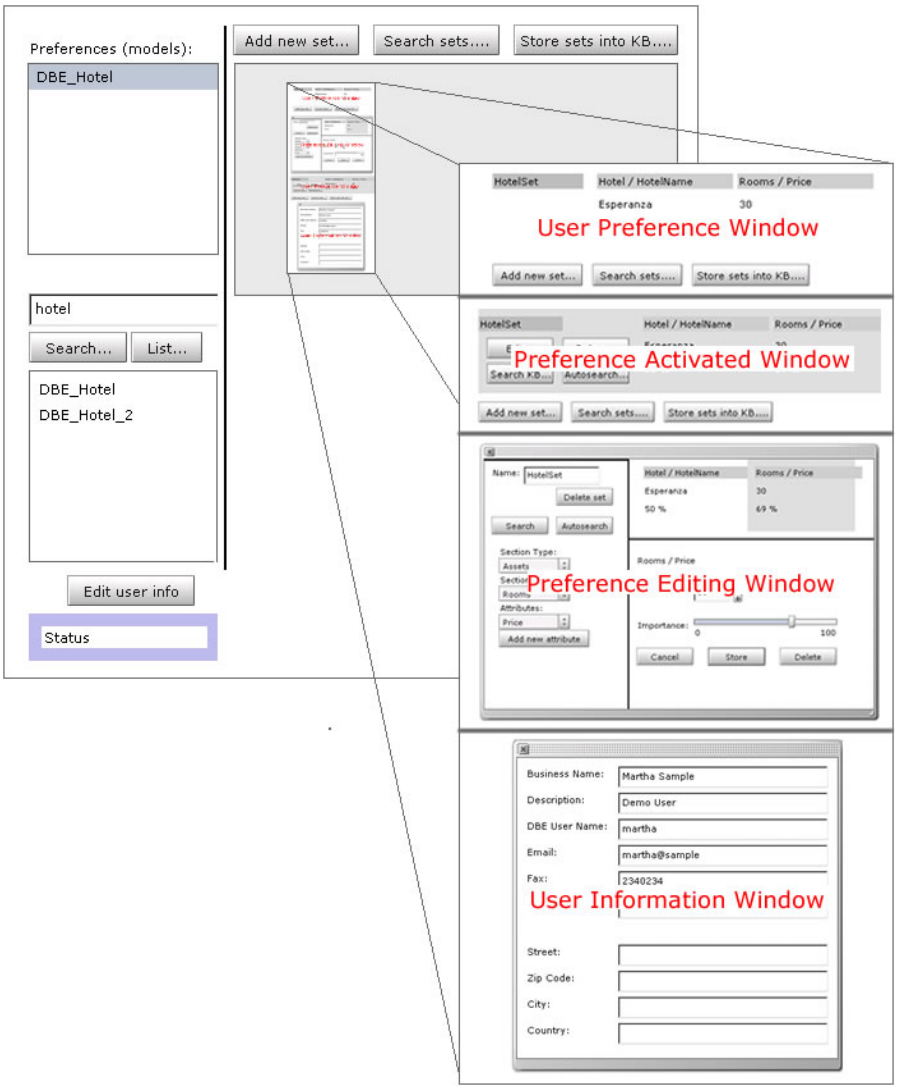



*Figure 8: User Profile Frame with available views (DBE Portal)*

- User Preference Window
  This view shows the Preference Sets within a User Profile Model created by the user.
- Preference Activated Window
  The activated Preference Set provides actions to edit, store and/or delete in this view.
- Preference Editing Window
  This important view allows the user to edit and change values, names, ranges and importance of selected attributes.
- User Information Window
  The window allows the users to enter personal information about them like name, address, email, contact and so on.

All aforementioned windows are described in detail in the next section.

### 3.1.2 Buttons and Functions

The following screenshots are taken from the original User Profile project and represent the finished tasks of work package 7. Figure 9 describes the User Profile frame window which is always visible to the user during a session. Additional graphical dialogs appear in the middle of the screen as shown in the previous Figure 8.

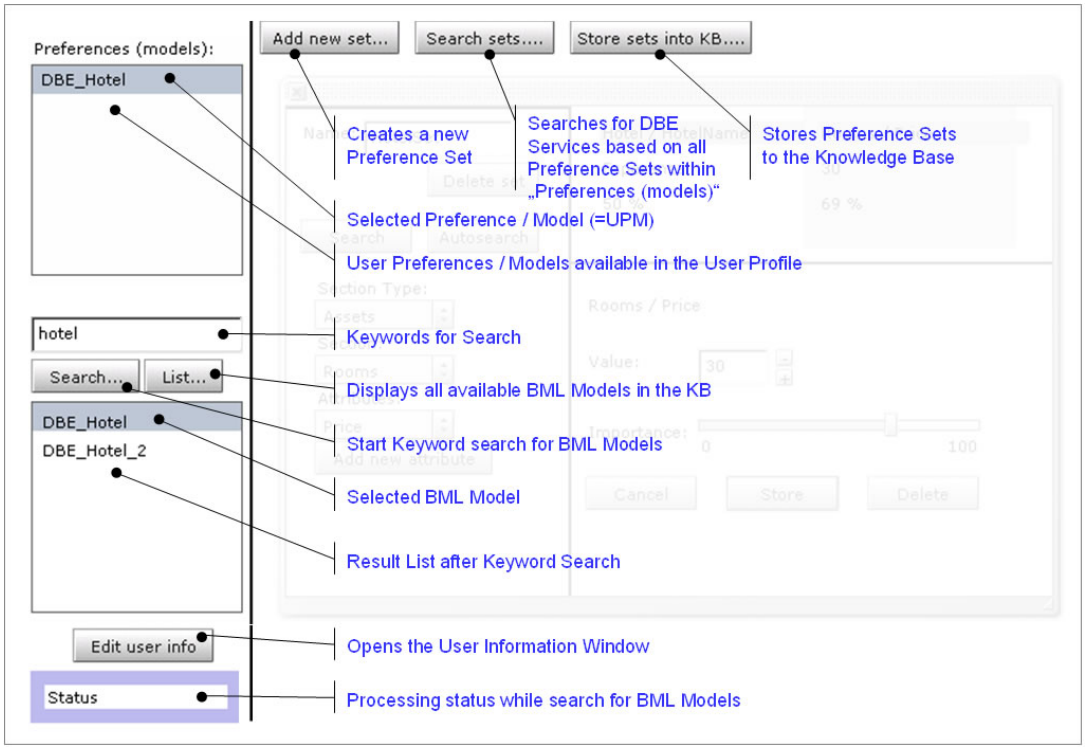



*Figure 9: User Profile Description – User Profile Frame (DBE Portal)*

The User Preference Sets as shown in Figure 10 are part of the selected User Profile Preference / Model. If the user clicks on a Preference Set it is automatically selected and

provides additional functionality like editing, deleting or searching for Preference Set specific DBE Services.

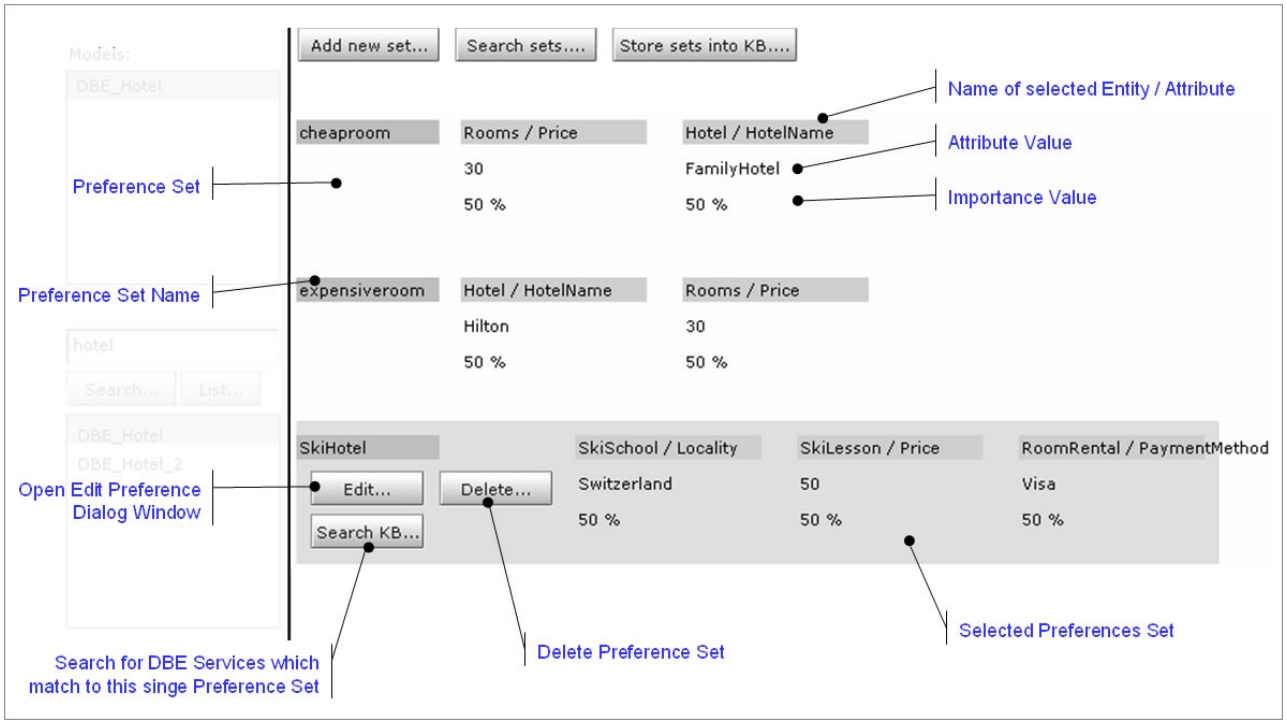



*Figure 10: User Profile Description – User Preference Activated Window (DBE Portal)*

The User Preference Editing Dialog shown in Figure 11 is the core functionality for creating, editing and deleting Preference Sets. Each Preference Set can consist of arbitrary many attributes.

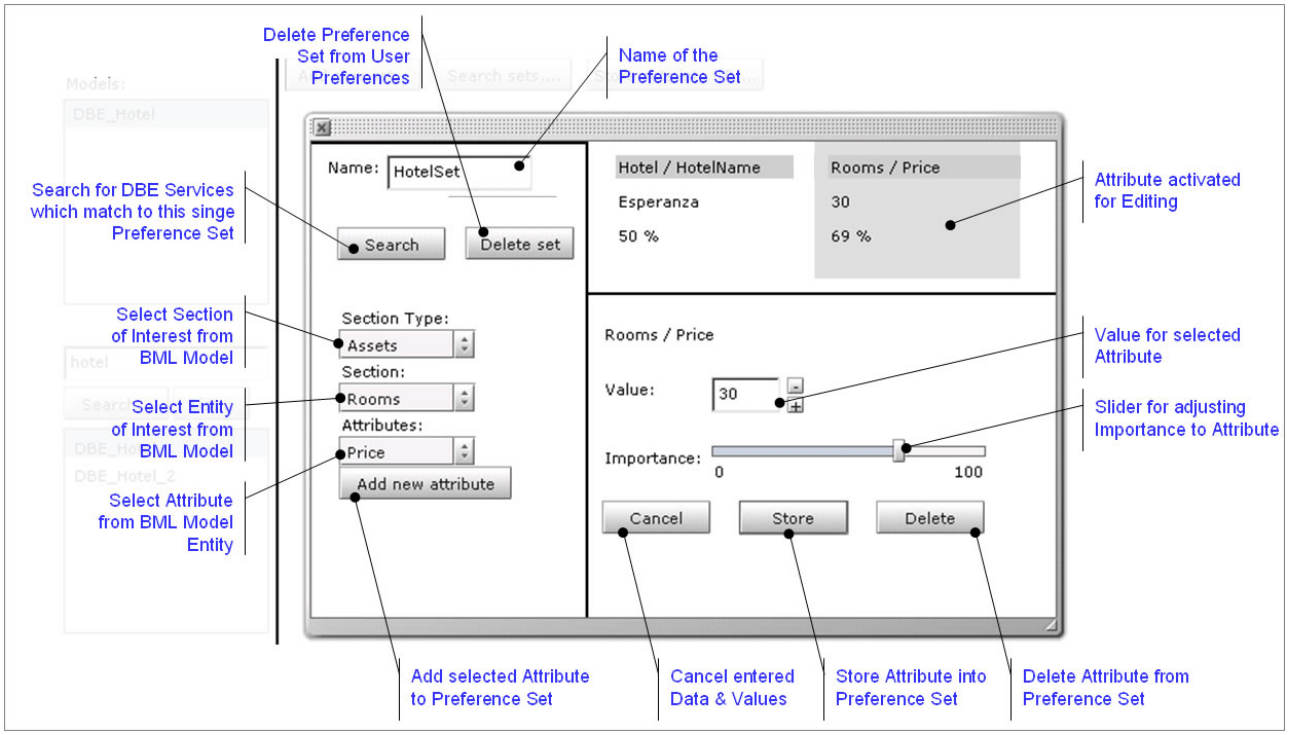



*Figure 11: User Profile Description – Preference Editing Window (DBE Portal)*

# 3.2 USER PROFILE FOR THE "DBE STUDIO"

## 3.2.1 Graphical User Interface

The Graphical User Interface of the Eclipse-based User Profile Plug-In for the DBE Studio is slightly different compared to the OpenLaszlo one. The following screenshots are taken from the original User Profile project and represent the finished tasks of work package 7. The application as shown in Figure 12 provides one view integrating six different Tabs.

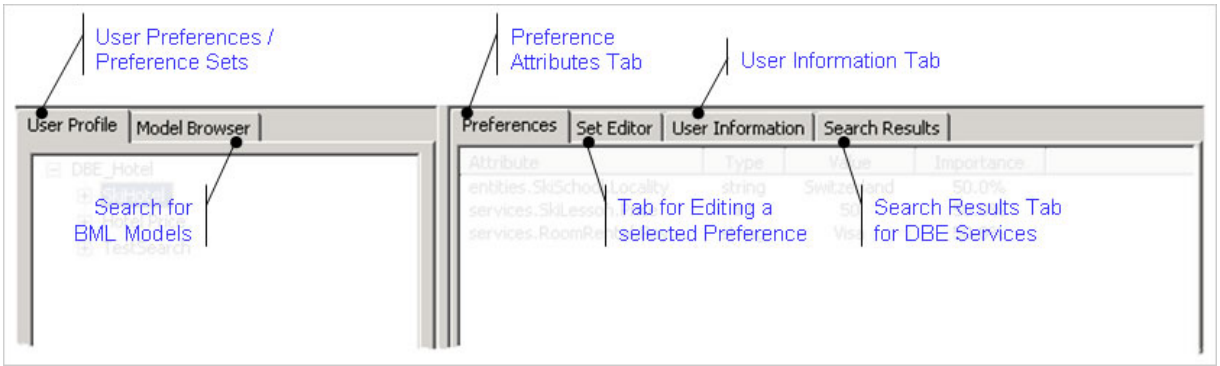



*Figure 12: User Profile Frame with available Tabs (DBE Studio)*

- User Profile Tab
  This view shows the Preference Sets within a User Profile Model created by the user.
- Model Browser Tab
  The user can search for BML models.
- Preferences Tab
  This Tab shows the concatenated attributes of the respective Preference Set
- Set Editor Tab
  Attributes of a Preference Set can be added, deleted or edited.
- User Information Tab
  This tab contains information about a user like name, country, contact, etc.
- Search Results Tab
  If the user initiates a search process considering the Preferences of the User Profile, this view shows a list of matched DBE Service.

The aforementioned Tabs are described in detail in the next section.

## 3.2.2 Buttons and Functions

The User Profile frame window is always visible to the user during a session and combines six Tabs. The first tab 'User Profile' (see Figure 13) informs the users about their currently

existing Preferences and Preference Sets. If a User Preference Set is activated, the corresponding attributes are displayed in the ‘Preferences’ Tab. The same attributes are simultaneously loaded into the “Set Editor” Tab even though the tab hasn’t been activated by the user, yet. If the user switches to this tab all attributes are ready to be edited.

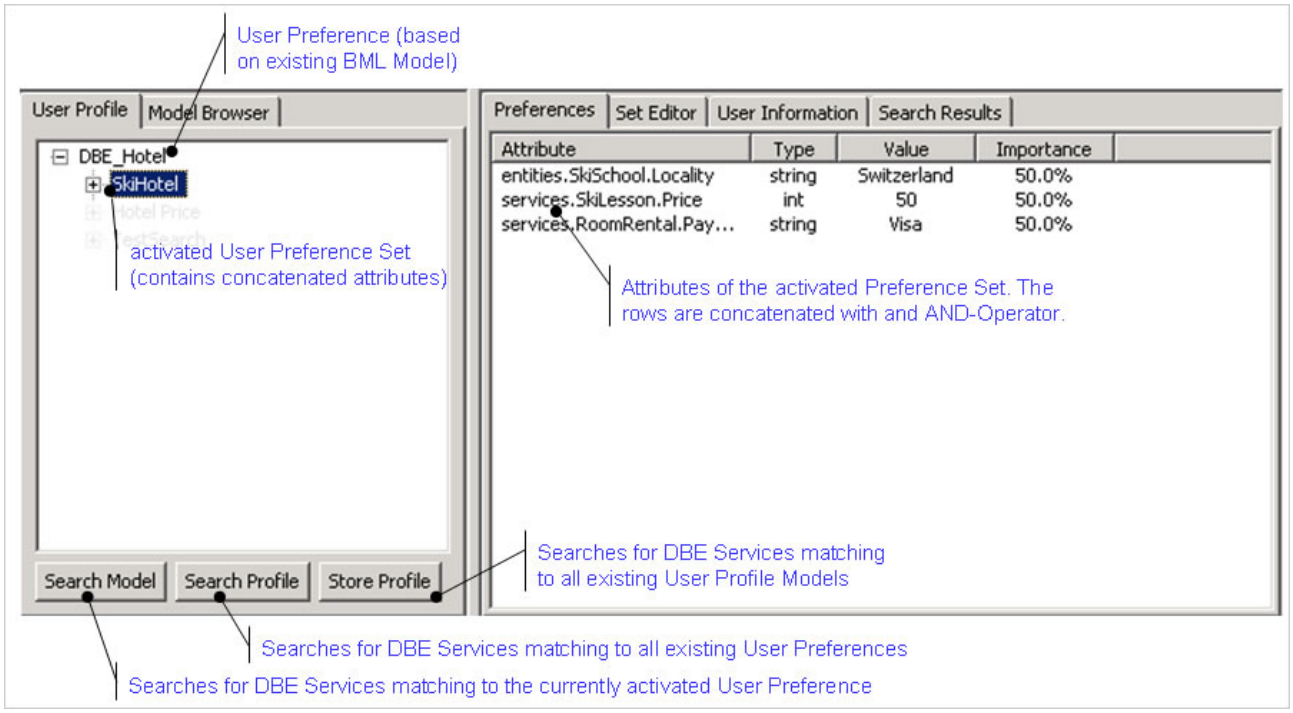



*Figure 13: User Profile Tab (DBE Studio)*

The Model Browser Tab shown in Figure 14 offers the possibility to search for BML Models available throughout the DBE. The keyword search mechanism allows to filtered search to a user’s individual needs.

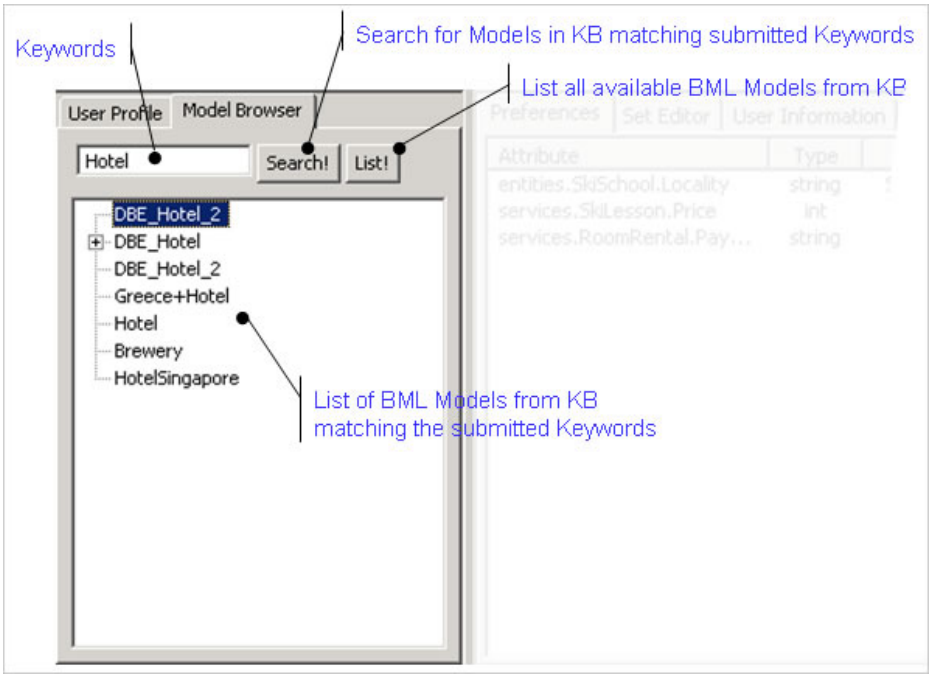



*Figure 14: Model Browser Tab (DBE Studio)*

Either the filtered result list appears or, if the user wants to browse for BML Models in general, the „List“ button shows all available BML Models. The users can add a model to their User Profile by double-clicking on the respective model. After that, the model is available in the “User Profile” Tab and takes the position of a Preference as shown in Figure 15.

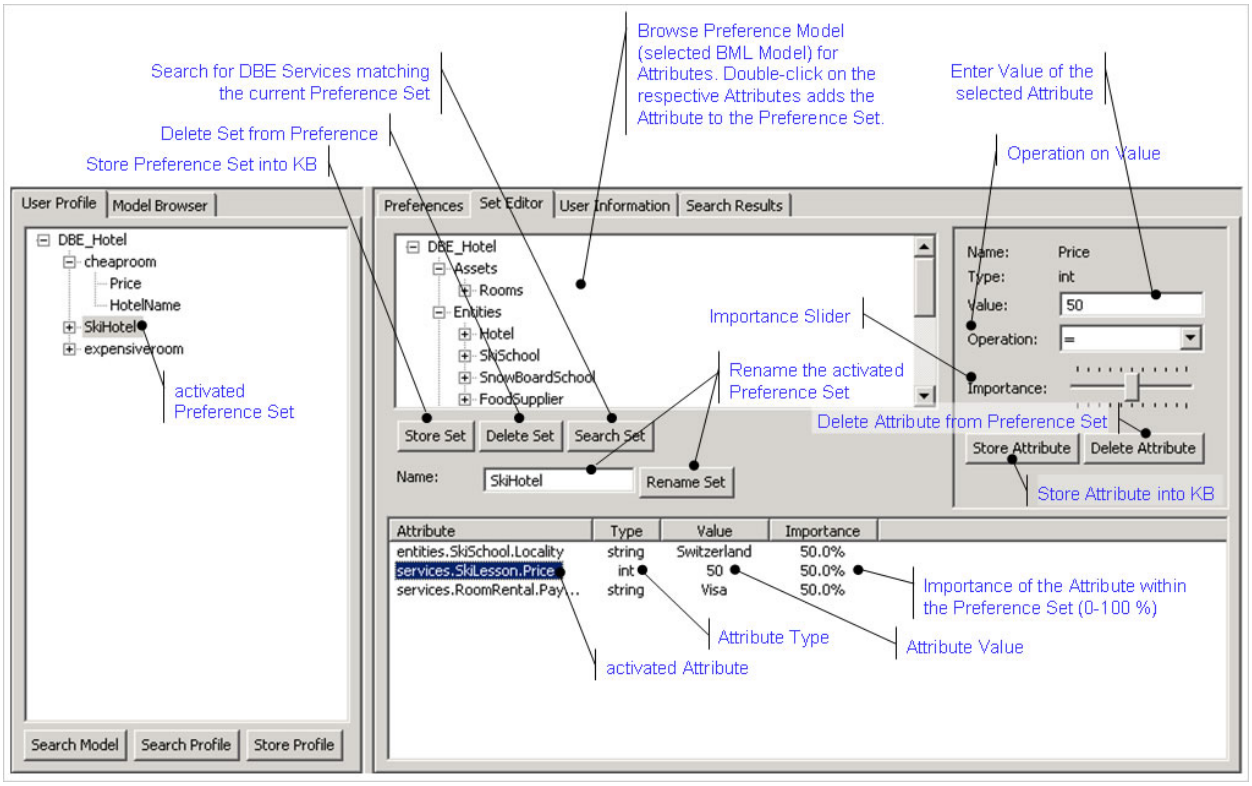



*Figure 15: Set Editor Tab (DBE Studio)*

The “Set Editor” Tab offers a variety of possibilities to the user. Especially the Preference browser is used to search for and select attributes to become part of a Preference Set. Each attribute gets an individual value and is assigned with an importance. After creating and editing Preference Sets in the “Set Editor” Tab, the user gets an overview over existing Preference Sets in the “Preferences” Tab as shown in Figure 16.

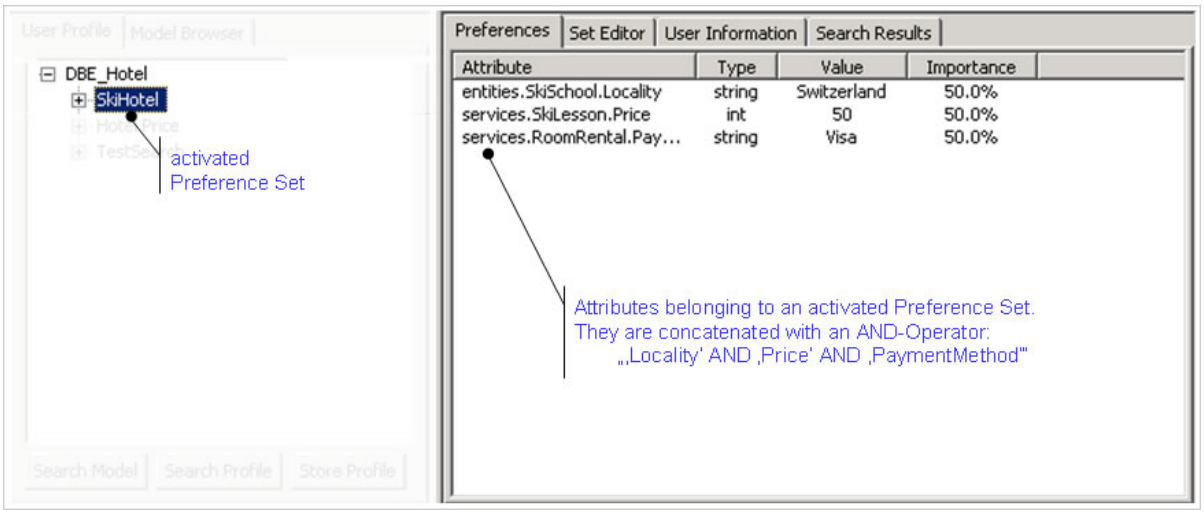



*Figure 16: Preferences Tab (DBE Studio)*

# 4 How to Use the User Profile

This chapter describes the use of the User Profile in practice. As already introduced in Deliverable D7.2 the processes for each step are modelled by using Event-driven Process Chains (EPC) [Bart05b]. The process modelling has been improved and structured better than was in D7.2. The final process modelling in this deliverable also considers the progress of the User Profile development between the Deliverables D7.2 and D7.4.

The method of the Event-driven Process Chain has been developed within the framework of ARIS (Architecture for Integrated Information Systems) [Sche94] and is used by many companies for modelling, describing and analysing 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 organization 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, which are event-driven. As such EPC forms the core technique for modelling in ARIS, which serves to link the different views in the so-called control view, which will be elaborated in section of ARIS Business Process Modelling.

An EPC is an ordered graph of events and functions based on the Petri-net theory by Carl Adam Petri [Mevi06, pp.84] providing several connectors, such as OR, AND, and XOR that allow alternative and parallel execution of processes [Mevi06, pp.68]. 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 the following Table 1 [Sche95, p.11].

| **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 the case of different resultant states can occurring, 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. |  |

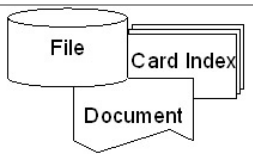
| | | |
|---|---|---|
| *Information Object* | The representation of a real world object which can be input data serving as the basis for a function, or output data produced by a function. |  |
| *Business Organisation Unit* | The representation of an enterprise's organisational units determining which person within the structure of an enterprise is responsible for a specific function. |  |

*Table 1: EPC Elements and Symbols*

As both the User Profile plug-in for the DBE Portal and the User Profile for the DBE Studio basically provide the same features the examples provided in the following sections are always illustrated by screenshots from both User Profile implementations. They are supposed to give an impression of how the User Profile implementation for the DBE Portal and the DBE Studio looks like. The Even-driven Process Chains presented in chapter 4.1 and chapter 4.2 describe the general usage of each activity (load, store, edit, …) and do not represent the technical GUI implementation.

## 4.1 LOG-IN, LOAD & STORE A USER PROFILE

The following Figure 17 describes the main User Profile usage process which consists of the three main activities "Load", "Edit" and "Store". After the DBE Portal / DBE Studio application is loaded the user launches the User Profile application by clicking the respective button. Before the User Profile of the individual user can be launched the user has to identify [SBMC, pp. 40] herself/himself by entering an identity string which is, in this case, the personnel e-mail address of the user (ID.100). This Email address is used as the User ID which identifies the correct User Profile XMI file[5] in the Knowledge Base.
If the Email address is correct the user's profile is loaded (ID.200) from the Knowledge Base using the Recommender Service (RC_Service). After the User Profile has been launched successfully the user can proceed with different editing-actions (ID.300, sub-processes are available) described in the following sections. After editing the User Profile everything is stored back into the Knowledge Base by using the RC_Service again (ID.400).

[5] An example of such a XMI-File can be found in Appendix 6.2 of this Deliverable D7.4.

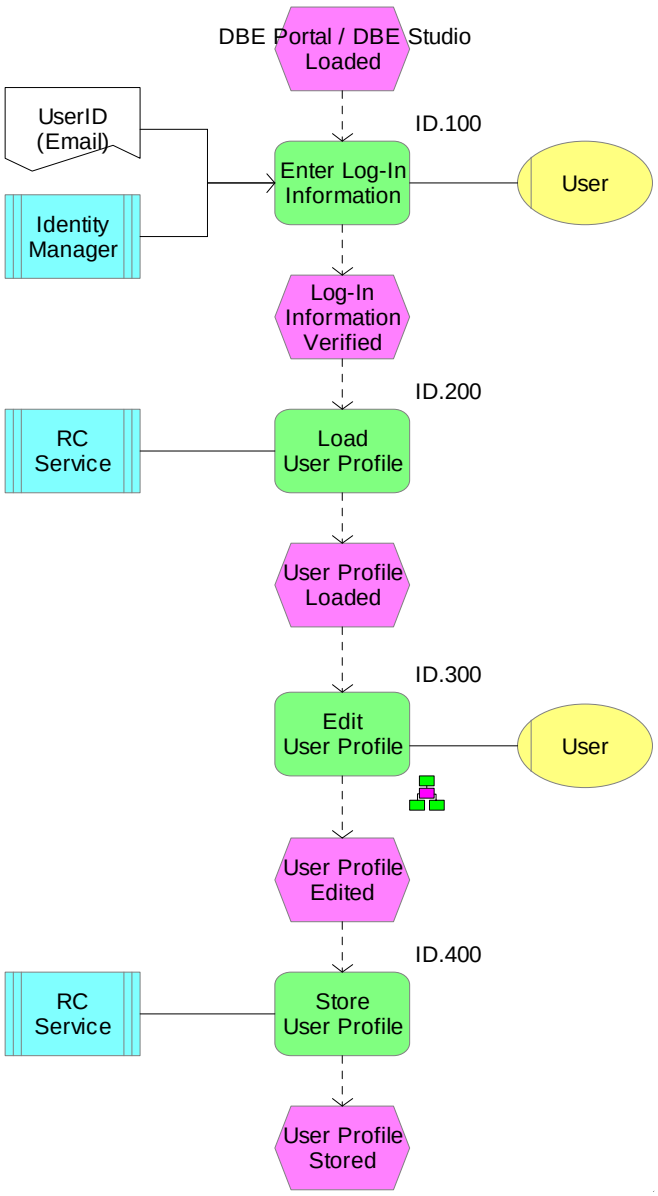



*Figure 17: Main User Profile Process*

The screenshots shown in Figure 18 and Figure 19 give an example of how the Log-In to the respective User Profile might look like.


*Figure 18: Example for Log-In / Identity window (DBE Portal)*

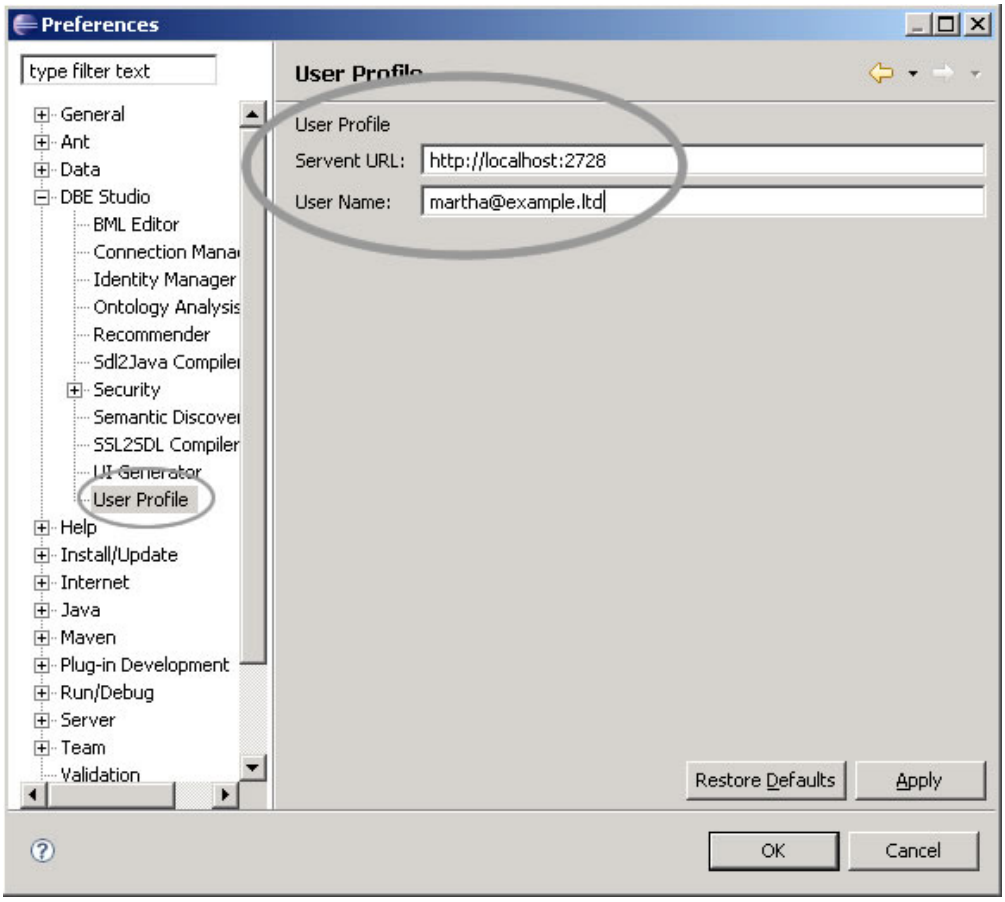



*Figure 19: Example for Log-In / Identity window (DBE Studio)*

The next section 4.2 illustrate aforementioned editing-actions accurately.

## 4.2 EDIT USER PROFILE

The editing process of the User Profile consists of the four main actions:

- Create User Preference
- Edit User Preference
- Add User Information
- Edit User Information

The first two actions contain further sub-processes describing further functionality.

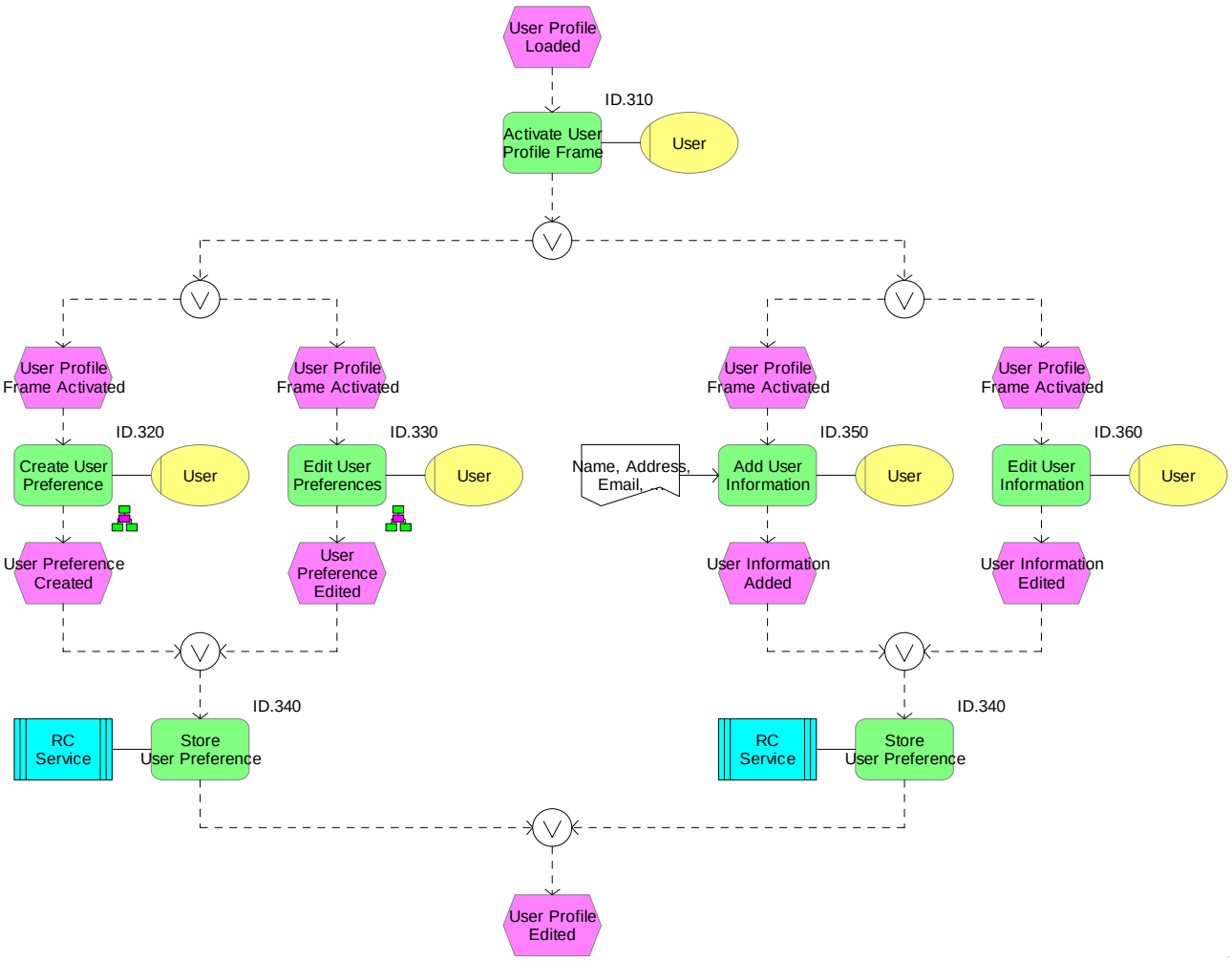



*Figure 20: Edit User Profile Process*

The quite comprehensive editing process is depicted in two steps, User Information (Section 4.2.1) and User Preferences (Chapters 4.2.2, 4.2.3 and 4.2.4) to ensure clearness and tangibility. All aforementioned four main actions are explained as follows.

## 4.2.1 Add / Edit User Information

A User Profile can contain personal information about a user as illustrated in Figure 21. After the User Profile has been launched the user presses the respective button to activate the User Information dialogue window (ID.310). After the window appears information such as name, address, contact information, email and so on can be added (ID350) or if already available be edited (ID.360). If the user is satisfied with the entered information the data will be stored into the User Profile (ID.340).

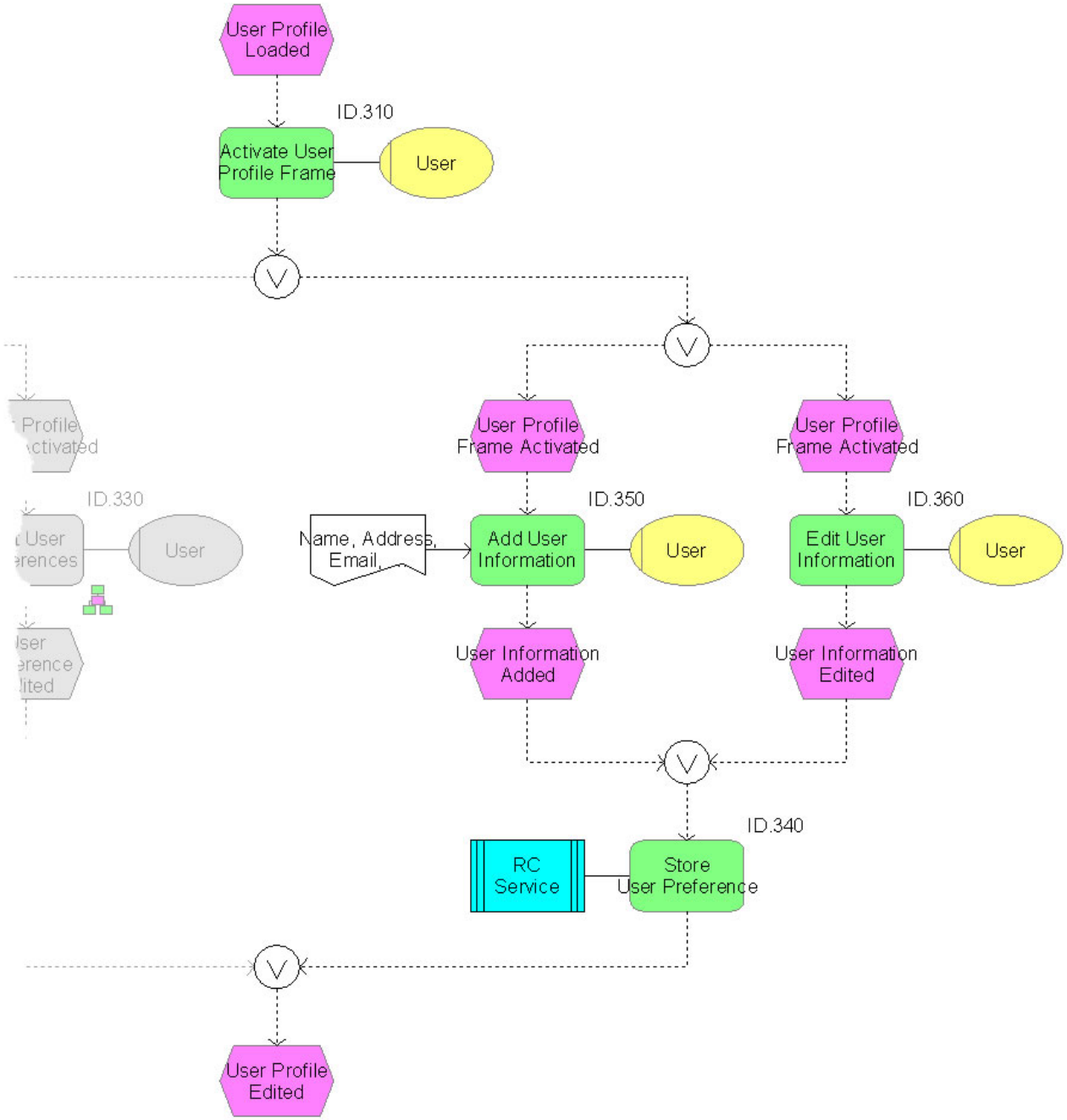



*Figure 21: "Edit User Profile" – Add/Edit User Information*

The screenshots of the OpenLaszlo-based User Profile application for the DBE Portal and the Eclipse-based User Profile plug-in for the DBE Studio are shown in Figure 22 and Figure 23.

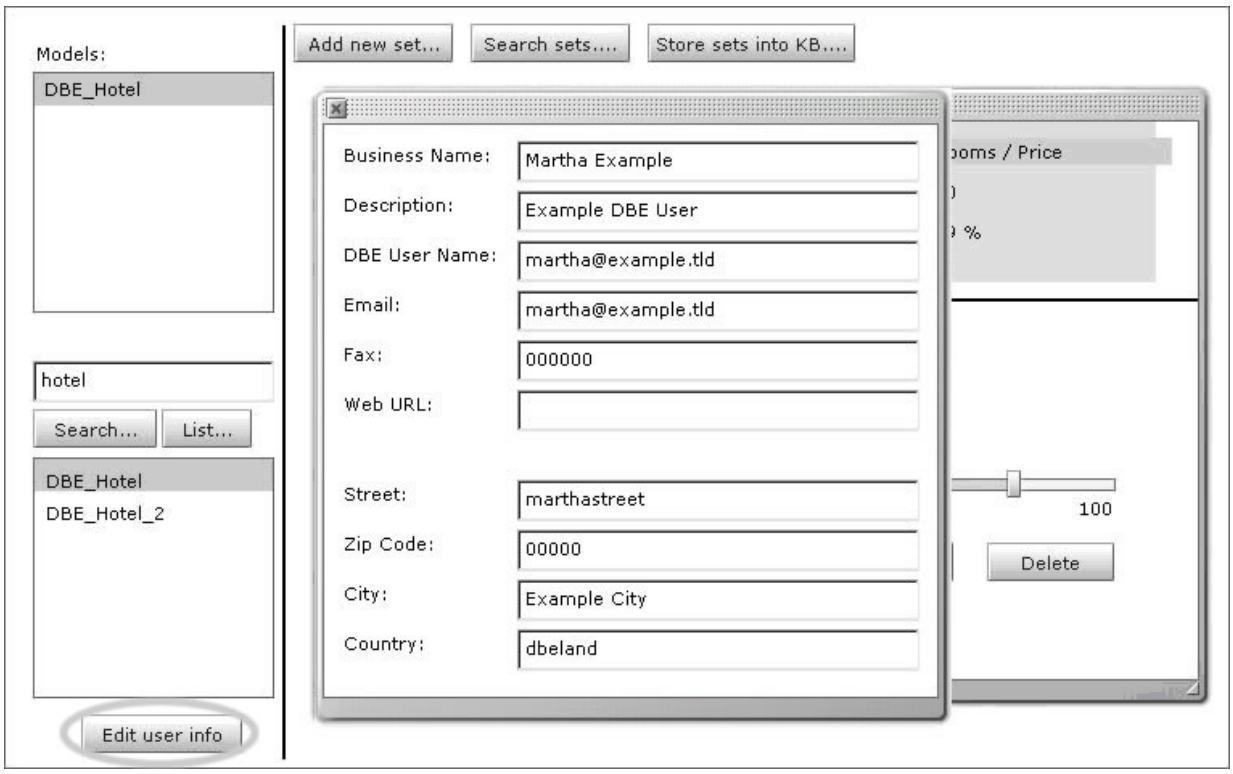



*Figure 22: Add/Edit User Information (DBE Portal)*

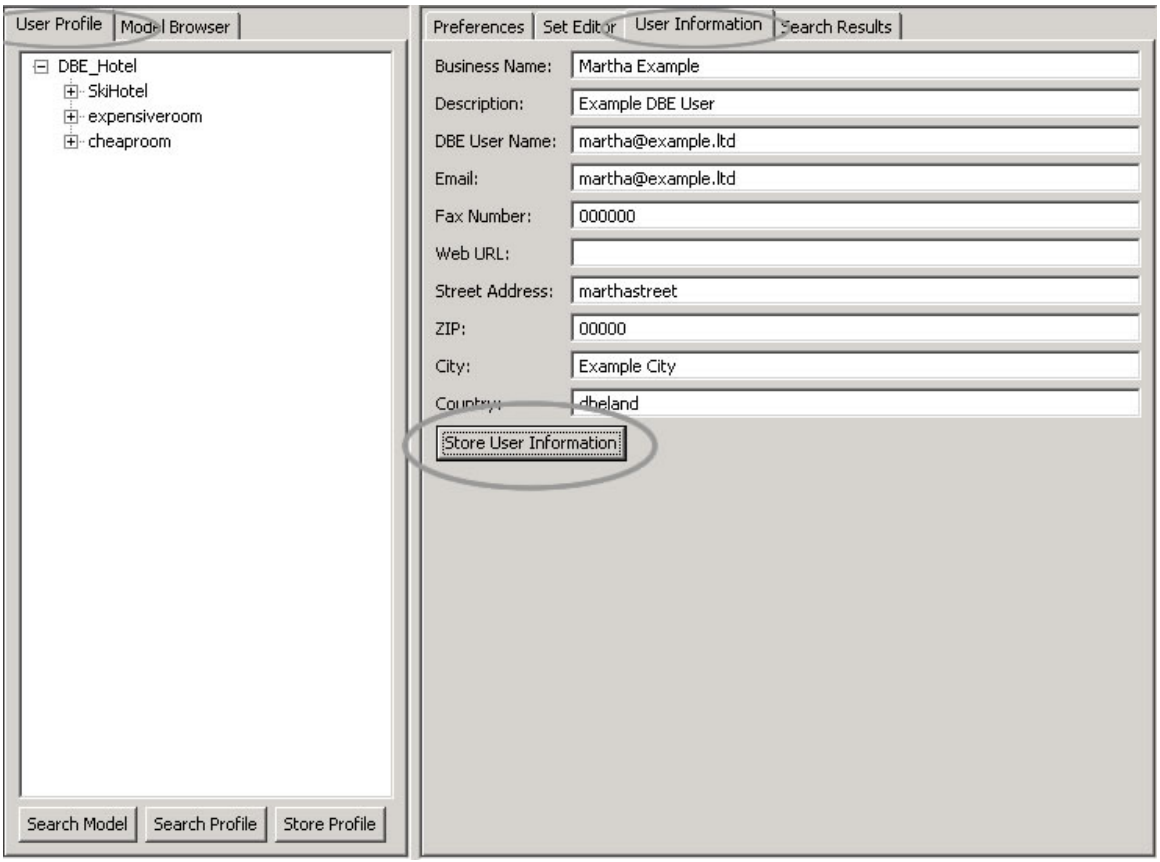



*Figure 23: Add/Edit User Information (DBE Studio)*

After pressing the “Edit user info” button in the lower left corner of the DBE Portal User Profile a pop-up window appears. The “User Information” Tab in the DBE Studio User Profile opens the User Information window. If the window is closed all added and/or edited User Information is stored into the User Profile automatically or by pressing the “Store User Information” button by hand.

### 4.2.2 Create User Preferences / Add Preference Sets

The creation of User Preferences, as shown in Figure 24, is the most important part of the User Profile as this is a main information source for other services such as for example the Recommender Service (RC_Service), to return adequate search results for services offered in the DBE. After the User Profile has been launched the (ID310) user presses the respective button (“Add new set”) to activate the User Preference dialogue window (ID.320). The Preference Set must always be named by the user. After that the Preference Set can be “filled” with attributes and values derived from the main BML-model chosen before. Each added attribute can contain an individual “Importance” value between 0 and 100 percent. This BML model builds the basis for a new Preference. A Preference can contain an arbitrary number of Preference Sets. The selection of a main BML model is explained in section 4.2.3. If Preferences and the corresponding Preference Sets already exist, the user might want to edit these (ID.330). How this can be done is described in section 4.2.4. If all editing activities are finished information is stored automatically and/or by pressing the “Store” button (ID.340).

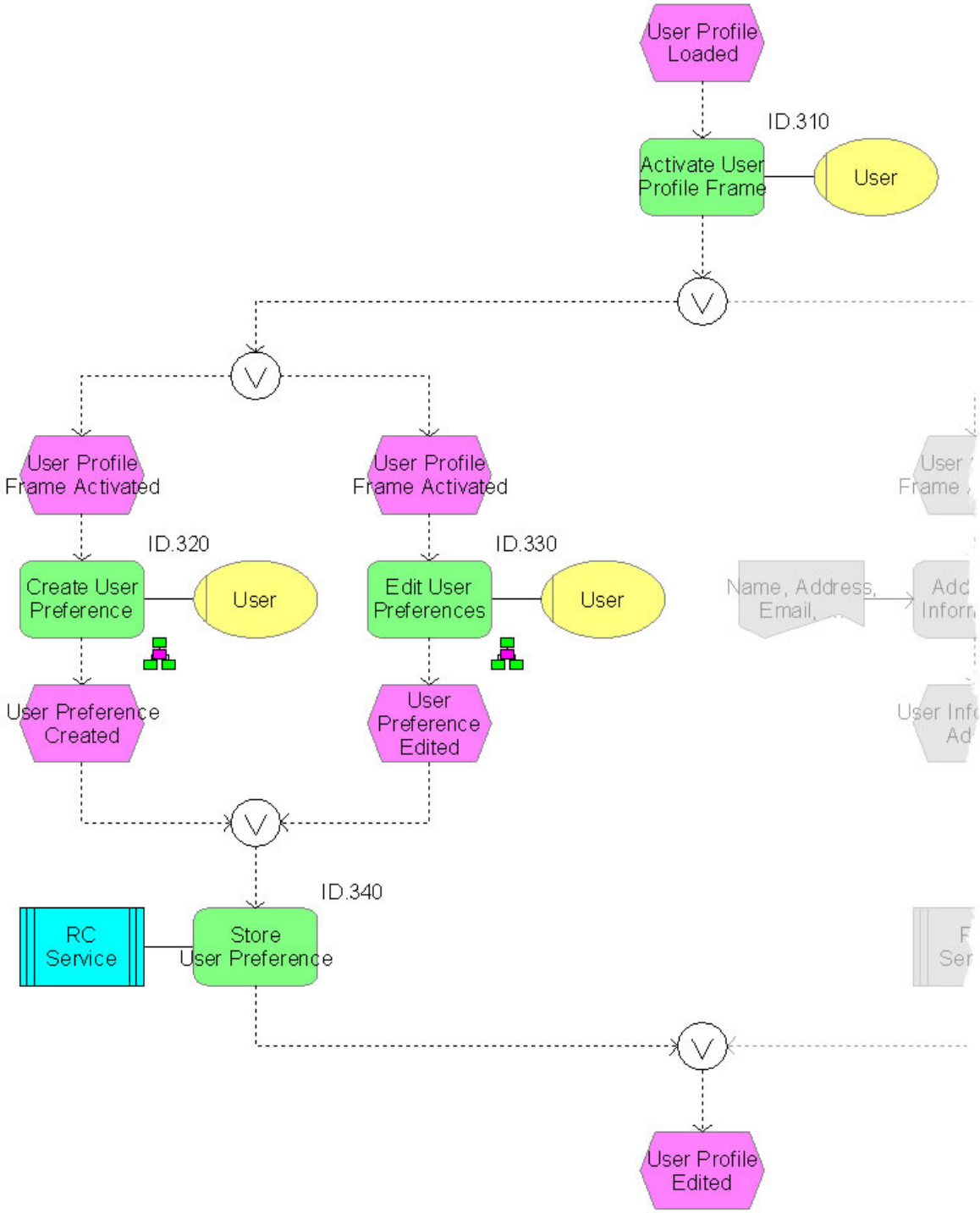



*Figure 24: "Edit User Profile" - Create Preference Sets*

For a better understanding of the aforementioned modelled process some screenshots of the developed User Profile application are shown as follows. Before Preferences Sets can be created the user has to search for a main BML model which can include a vast number of attributes. This procedure is described in detail in Section 4.2.3. In Figure 25 one can see that the Preference "DBE_Hotel" is empty and contains no Preference Sets. If the user wants to add some Preference Sets consisting of attributes and corresponding values to

the DBE_Hotel Preference a new set has to be created by pressing the “Add new set” button.

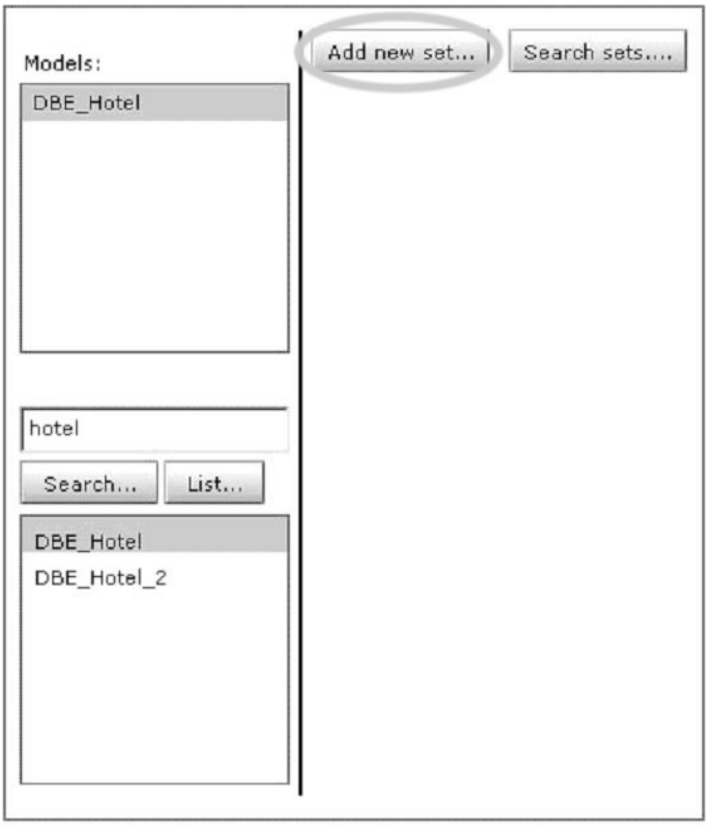



*Figure 25: Empty Preference (DBE Portal)*

No special “Add New Set” button is needed in the DBE Studio application, as this functionality is automatically executed if double-clicking on the respective attributes in the “Set Editor” dialog window.

After the Preference Set container has been created the user can add an arbitrary number of attributes, values and importance ratings to it. All attributes within a Preference Set are concatenated with an implicit AND operator which is an important characteristic when validating requests during a search for services. Figure 26 and Figure 27 illustrate how an attribute is selected, assigned with a value and rated with a certain importance value. The name of the User Preference Set container is “Hotel Price”. The selection of attributes has to follow a special procedure as a result of the structure and characteristic of the main BML model. Therefore the user first selects the type “Assets” which contains the artefacts of interest. The “Section” drop-down menu allows the user to select a specific artefact, in this case “Room”. Finally, the attribute of interest (here: “Price”) needs to be selected to build a meaningful statement. The pair Room/Price is completed after the user has added a concrete value (here: “Value = 30”) and an individual importance value. This is the formal and machine readable representation of the User Preference: “I prefer hotels offering rooms with a price of 30”. The edited attribute is added to the Preference Set by

storing the changes. After that the user can choose another Section and/or Attribute to add further information to the current Preference Set.

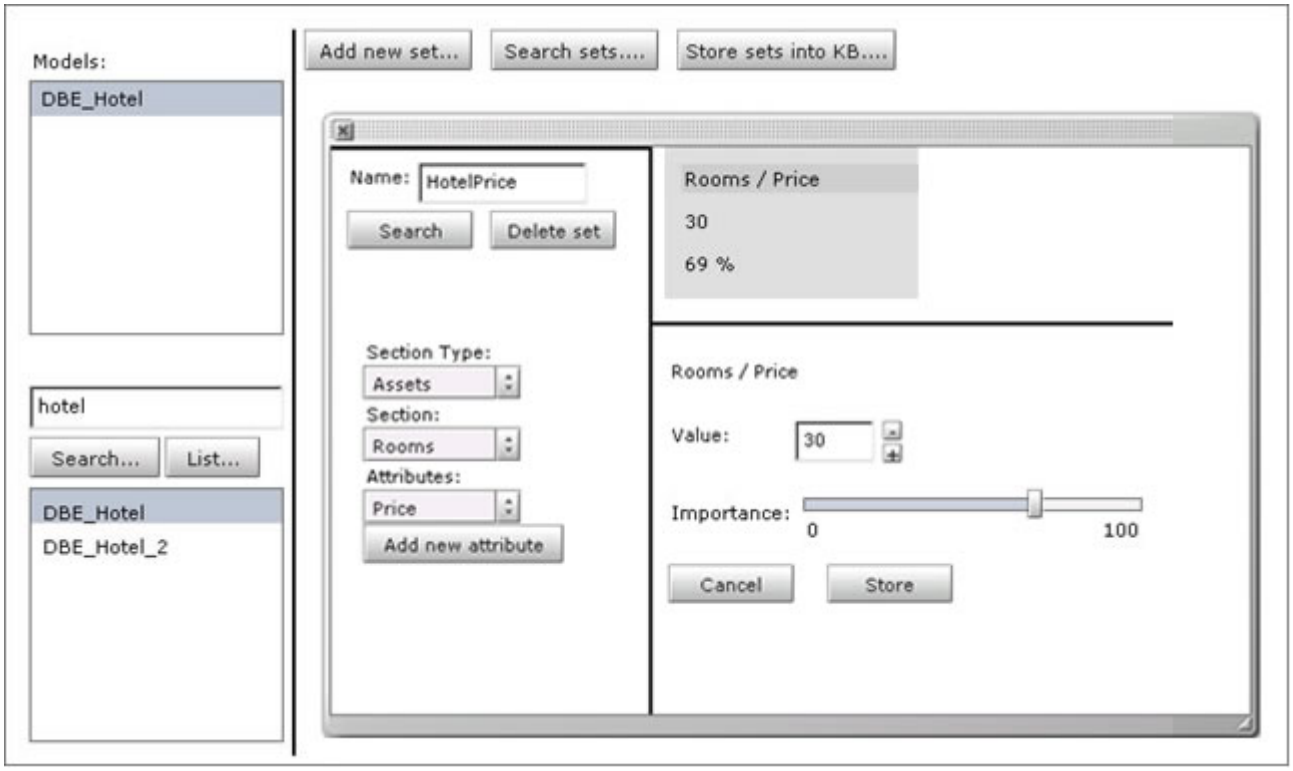



*Figure 26: Naming a Preference Set / Adding Attribute and Value (DBE Portal)*

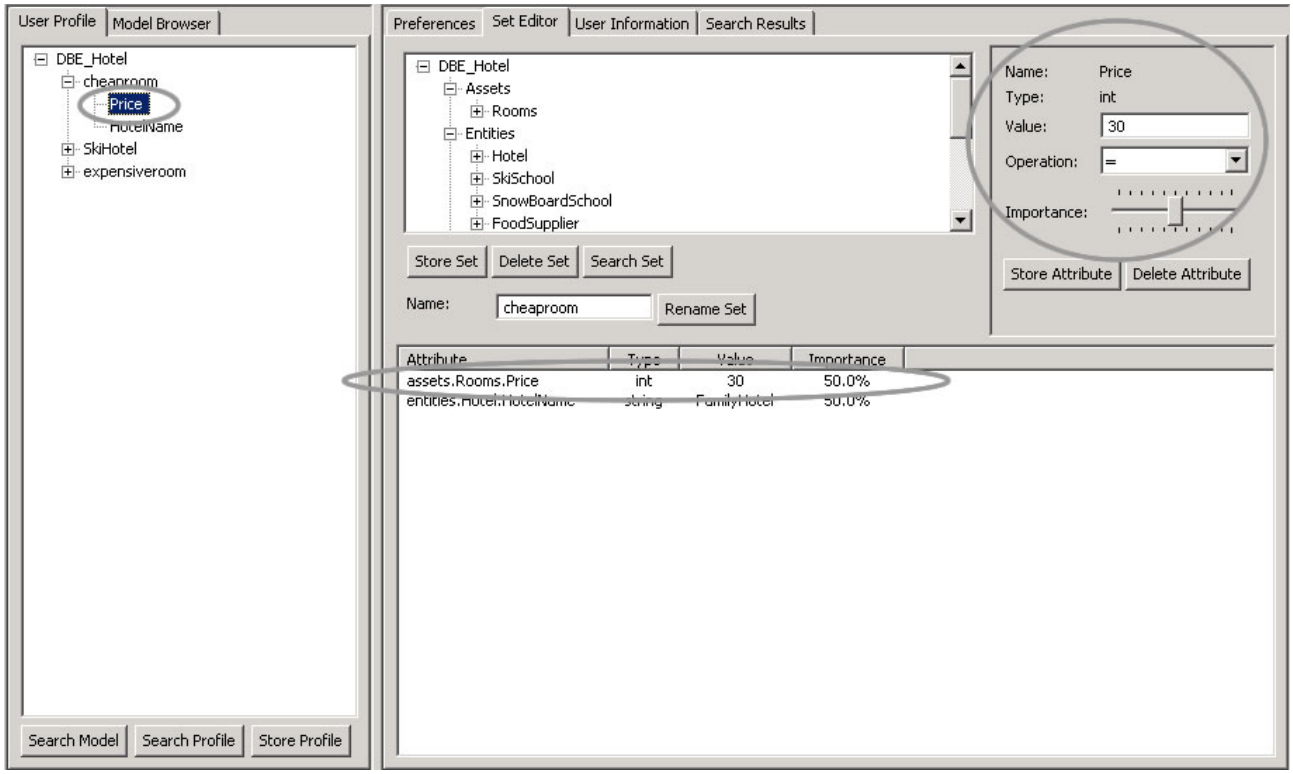



*Figure 27: Naming a Preference Set / Adding Attribute and Value (DBE Studio)*

### 4.2.3 Search & Create User Profile Preference

As mentioned before, the first step before adding/editing Preference Sets and attributes is to create the Preference "container" itself. This means that the user needs to select an appropriate BML model. It is assumed here, that broad ranging BML models referring to any domain of the opportunity space[6] exist in the DBE and that a user can utilise these BML models and their information sufficiently. The process in Figure 28 describes how to search for an initial Preference BML model which will contain the Preference Sets later on.

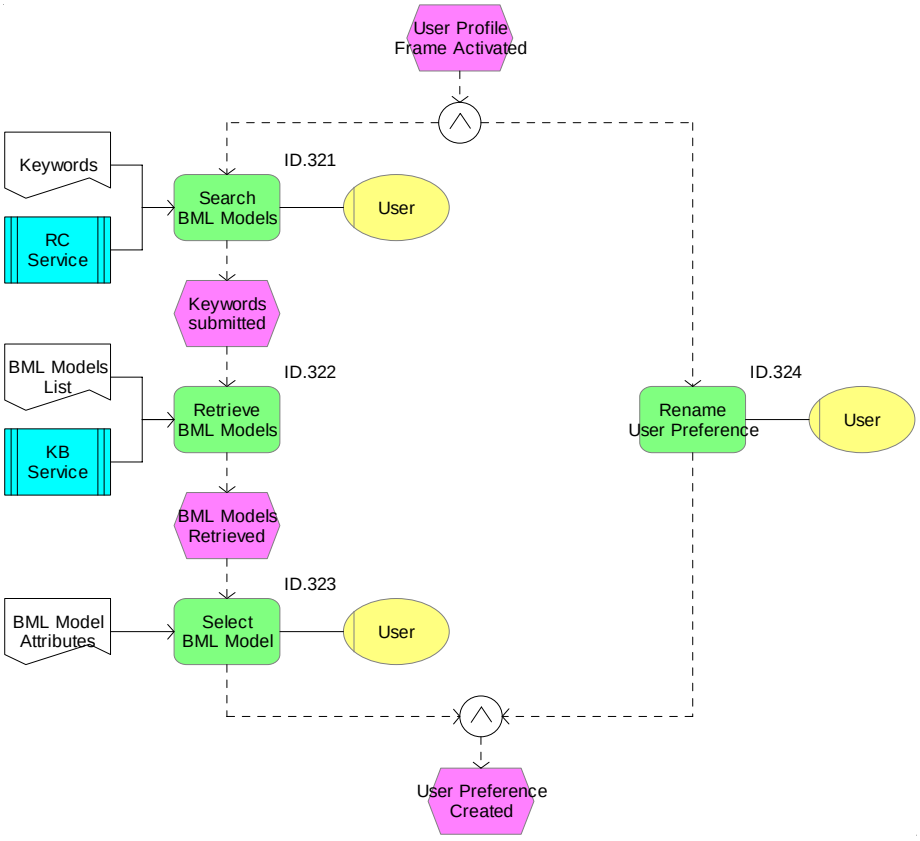



*Figure 28: "Edit User Profile" - Create User Preferences*

The search for a new BML model starts with entering any keywords into the search field (ID.321). After the desired keywords are submitted via the RC Service the matched BML models are retrieved from the Knowledge Base (ID.322). The user is presented a result list with models. If the user finds a BML model which might match to their interests the BML model is selected and automatically appears in the "Preference (models)" window (ID.323). Each BML model which appears in this window represents a pool for attributes which are available for creating Preference Sets. Figure 29 illustrates the composition of a User Preference in dependence on Figure 42.

[6] The idea of the Opportunity Space has intensively been described in Deliverable D7.1 in section 3.2. [Bart05a, pp.11].

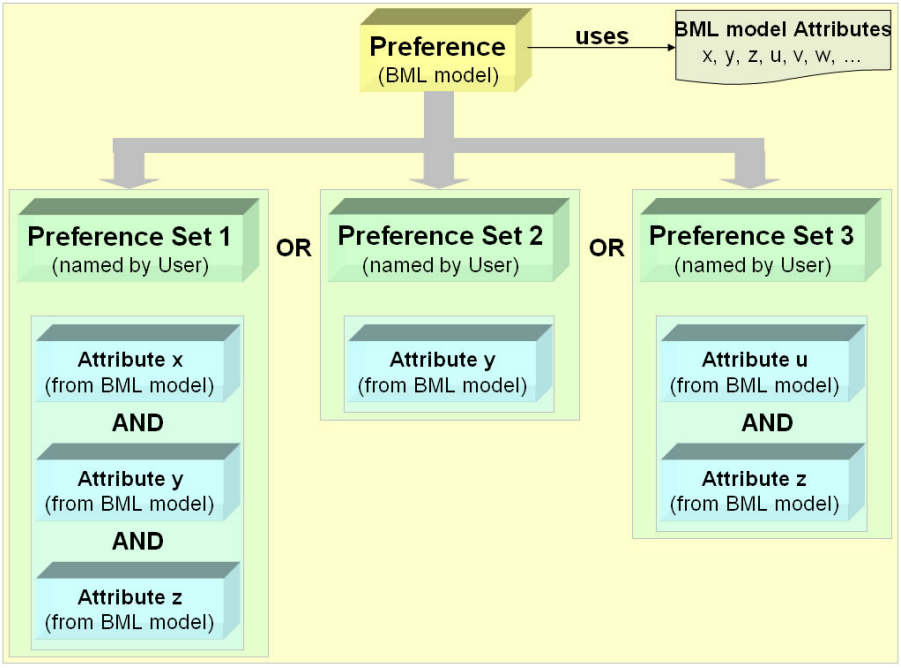



*Figure 29: Preference Structure*

The screenshots from the implemented application shown in Figure 30 and Figure 31 visualise aforementioned steps.


*Figure 30: Keyword Search for BML models (DBE Portal)*

After choosing the DBE_Hotel model from the Knowledge Base, the user can access all sections and attributes contained at the model.


*Figure 31: Selecting Preference from BML model (DBE Portal)*

This step can be repeated as often as necessary to add further Preferences to the User Profile. Of course deleting or editing a Preference is also possible at any time.

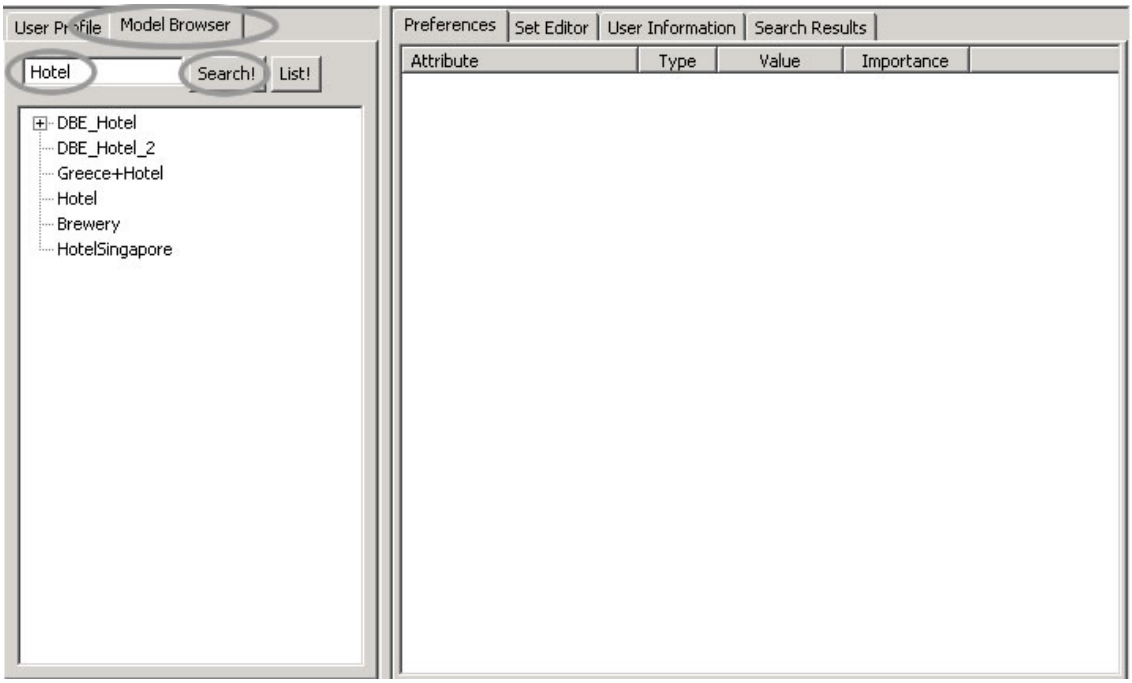



*Figure 32: Keyword Search for BML models (DBE Studio)*

The same functionality is provided in the DBE Studio User Profile implementation as illustrated in Figure 32 and Figure 33.

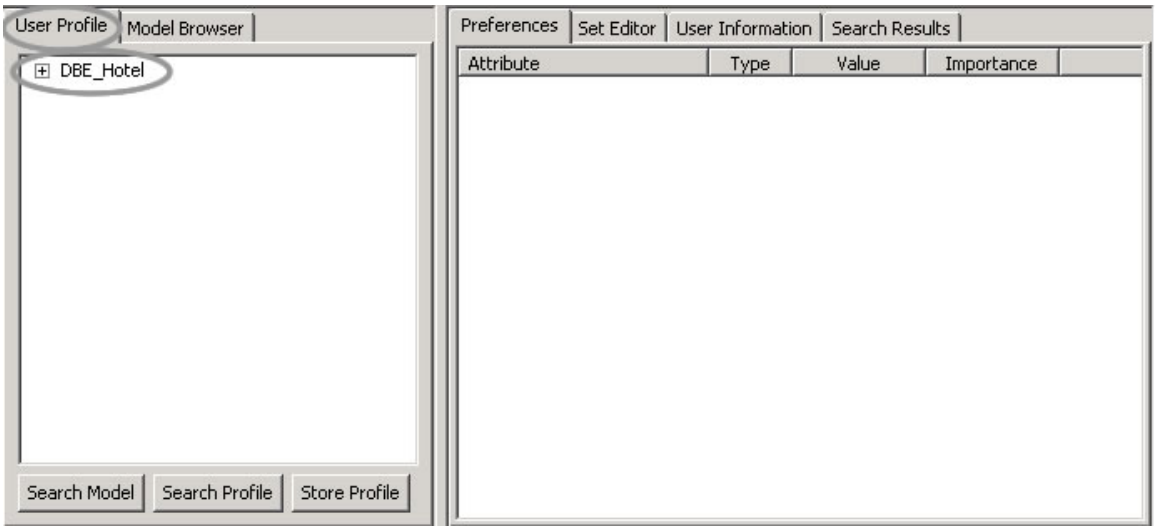



*Figure 33: Selecting Preference from BML model (DBE Studio)*

### 4.2.4 Edit User Preferences

Editing a Preference or a Preference Set will probably be the action most commonly used. This happens if the user wants to change or adapt attributes and their corresponding value. Figure 34 describes the process for editing Preference Sets and the selected attributes. The user selects the Preference Set which needs to be edited and activates the desired entity within Preference Set (ID.331). The entity is now ready to edit and the respective attribute and the value can be changed or deleted (ID.332). The kind of value which can be added to an attribute depends on the attribute type. If the attribute has the type String for example, the user will be able to enter a String value for the attribute. If the attribute has the type Integer, the user can add figures instead of text. Several possibilities are mentioned in the current editing process (ID.333, ID.334, ID.336, ID.337). Each attribute owns a value describing how important this attribute is to the user. The “Importance” can differ between 0% (unimportant) and 100% (required).

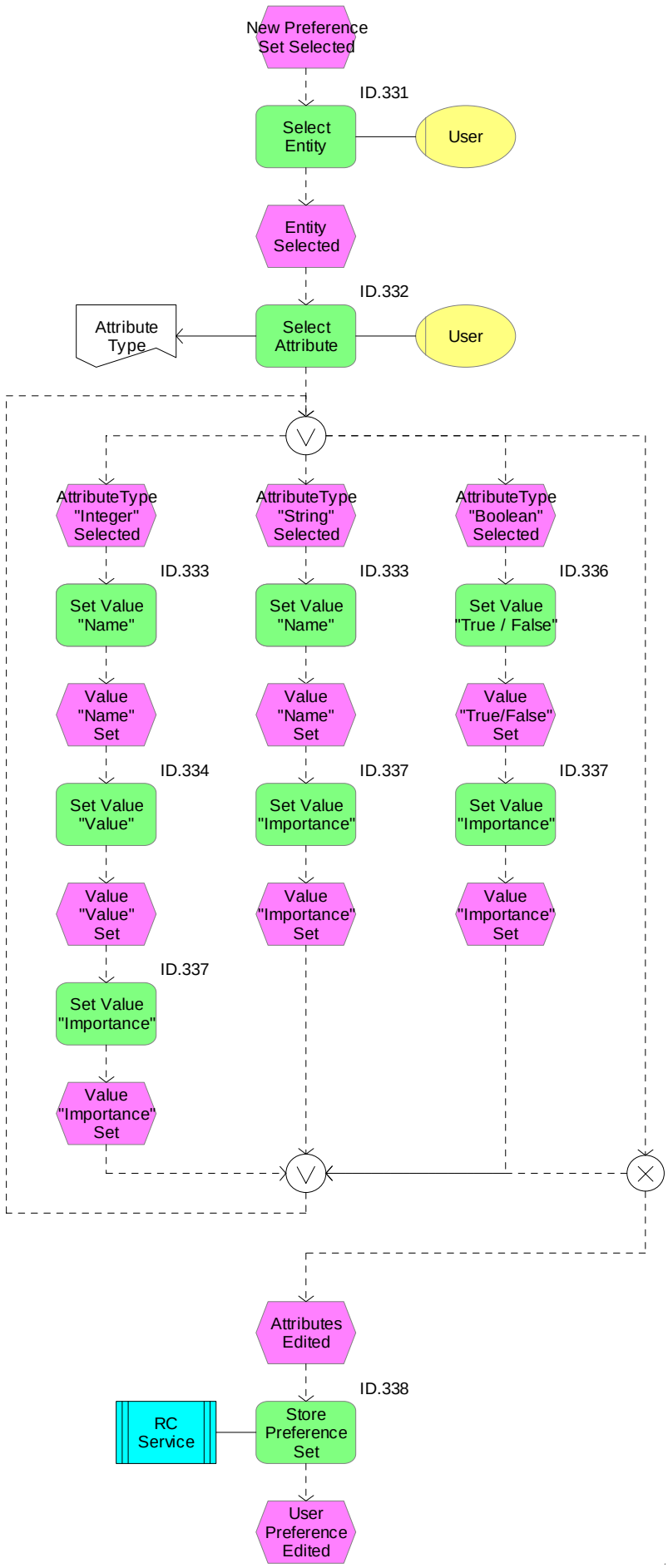



*Figure 34: Sub-Process "Edit User Profile" - User Preferences*

This fuzziness is necessary for matching the User Preferences with available services in the DBE. In the editing process each Preference Set can be extended with further attributes or reduced by deleting attributes. If the user has finished the editing process and is satisfied with the changes, the whole data is stored to the User Profile automatically (ID.338).

The screenshots in Figure 35 and Figure 36 show the Preference “DBE_Hotel” which contains two Preference Sets “Hotel Price” and “Ski Hotel”. Each Preference Set has two attributes with values and an importance rank. The button “Search sets” can now be used for searching a service by matching one of the two Preference Sets. This functionality is described in section 4.2.5.

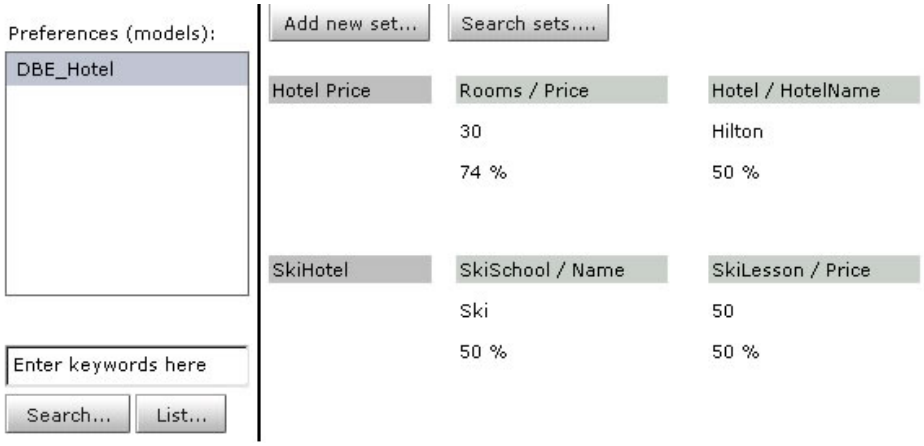



*Figure 35: Available Preference Sets Overview (DBE Portal)*

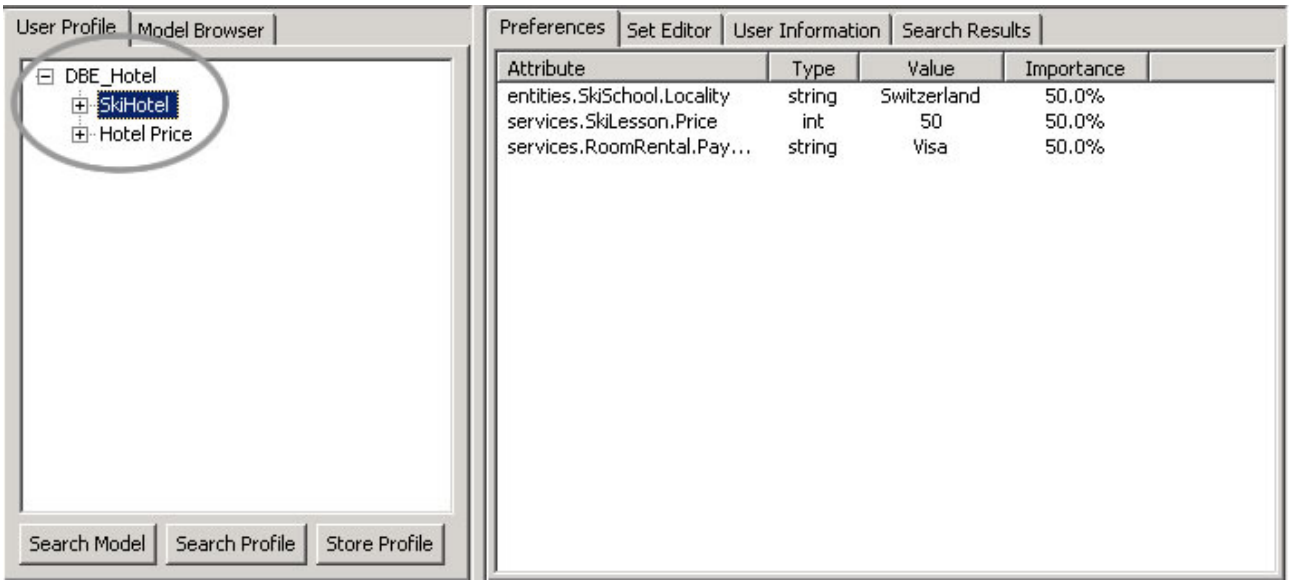



*Figure 36: Available Preference Sets Overview (DBE Studio)*

After the user has selected a Preference Set by clicking on the Preference Set's name several possibilities exist, as shown in Figure 37 and Figure 38. Either the Preference Set can directly be deleted or, by pressing the "Search KB" button, services are looked up by matching only with to this Preference Set. If pushing the "Edit" button, a separate window will open and the particular attributes can be selected and changed.

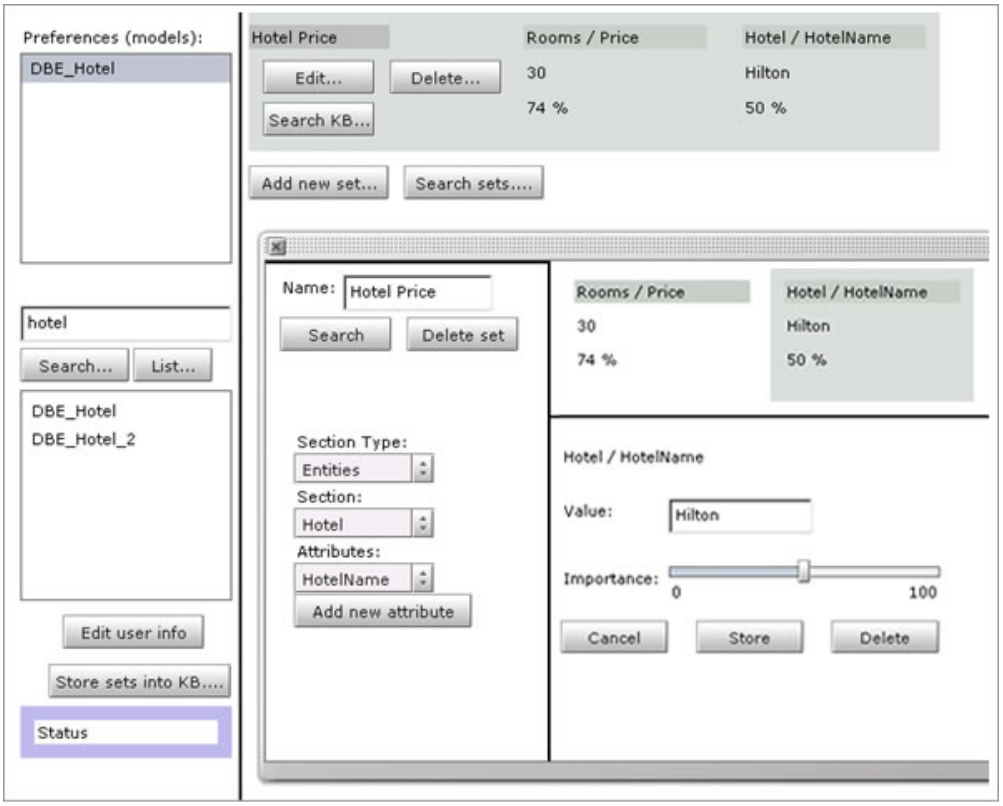



*Figure 37: Editing Preference Set Attributes (DBE Portal)*

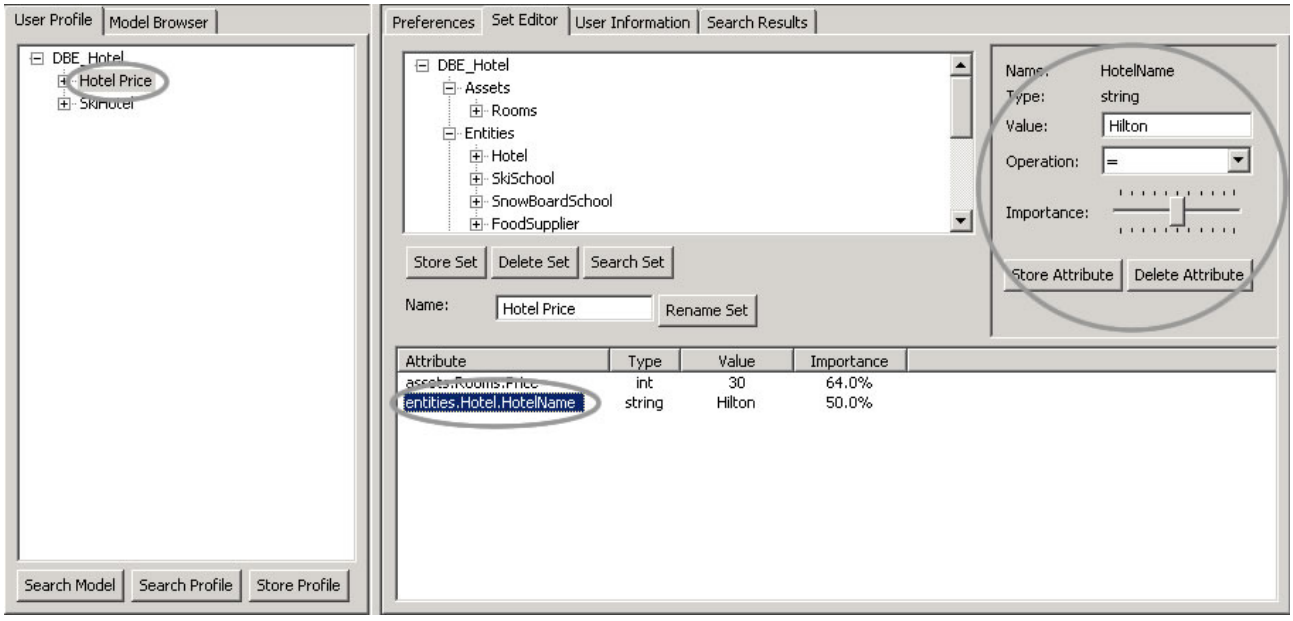



*Figure 38:Editing Preference Set Attributes (DBE Studio)*

### 4.2.5 Search for Services

The search for services can be initiated manually in an ad-hoc manner. The search for services can take into account:

- All Attributes within a Preference Set

or

- All Preference Sets within a Preference

or

- All Preferences within the User Profile

The quality of search results is (as always in matching processes) strongly dependent from the number of restrictions submitted to the search query (number of attribute/value tuples). Therefore it is intuitively evident that the search will provide more results of “lower” quality when matching only one single attribute/value tuple than matching a dozen tuples. Anyway it is assumed that if the number of services in the DBE increases massively in the future the probability for matching many attribute/value tuples successfully will increases, too. Then the users might receive search results even for services which match even to a whole User Profile and not only to a single Preference Set. Figure 39 and Figure 40 show the current possibilities for successful matching as not enough services are available in the DBE at the moment.


*Figure 39: Search Results Preference Set with single tuple (DBE Portal)*

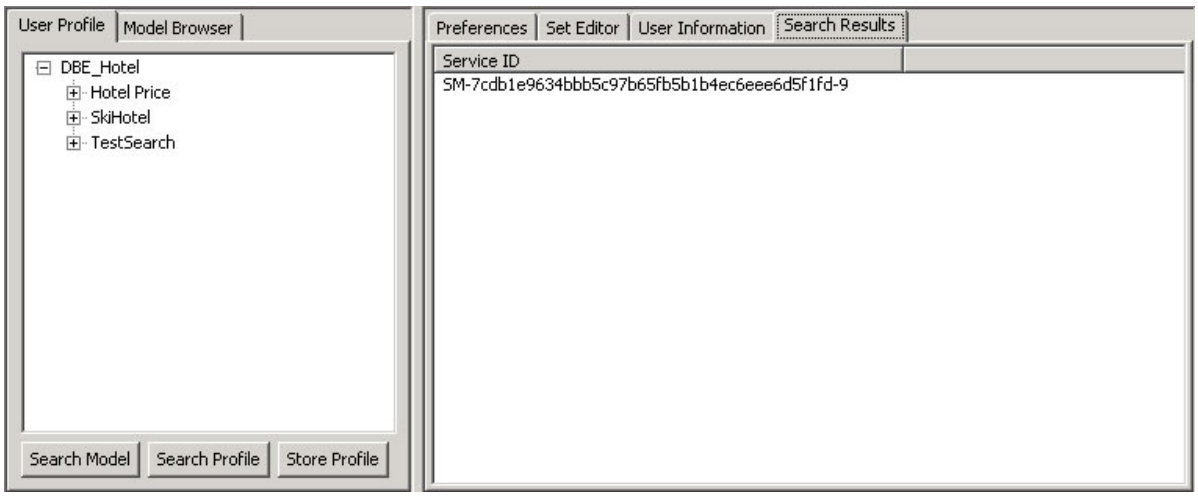



*Figure 40: Search Results Preference Set with single tuple (DBE Studio)*

When searching for a single attribute/value tuple the search matches and retrieves at least one service. The result list shows the Service Manifest Identifier of the found service. If the user double-clicks on this string the service information is loaded and can be viewed.

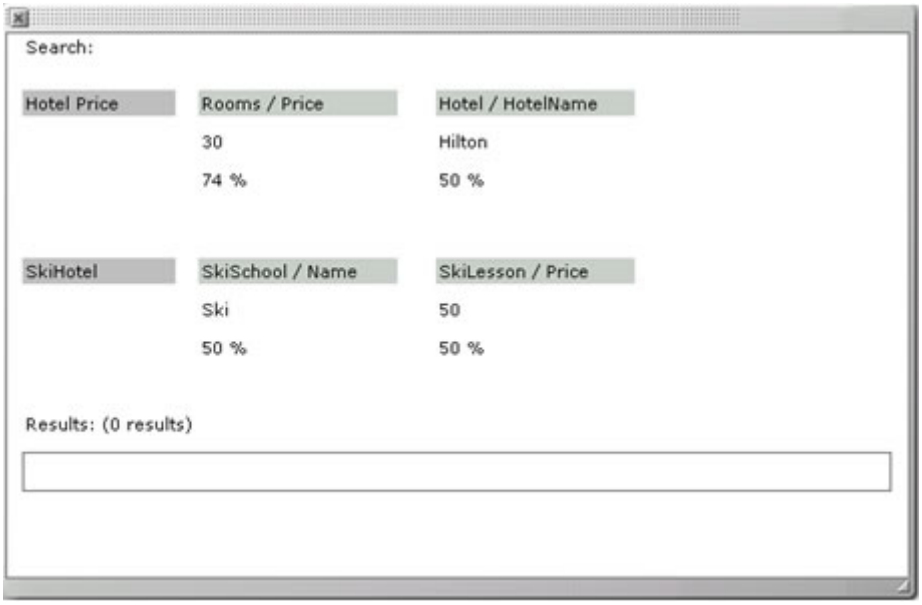



*Figure 41: Search Results Preference with two Preference Sets (DBE Portal)*

If the search is initiated using the complete Preference as shown in Figure 41 and (due to the little number of available services at the moment) the search result list is empty as no services match with the considered attribute/value tuples. Yet another feature is the automated generation of search results as soon as the user starts the User Profile. The user retrieves new recommendations for services as soon as further services have been deployed to the DBE and match to the user's Preference Sets. Of course this is a functionality which only makes sense, if the number of available services is big enough. Basically, the kind of search is the same as the manual one but it is triggered automatically after launching the User Profile.

# 5 LESSONS LEARNT

From the point of view of Work package 7 “User Profiling” the activities in the DBE project mainly resolved in the following “Lessons Learnt”:

- **Sophisticated User Profile modelling and representation increases flexibility**
  The User Profile Metamodel describes the relations and dependencies between attributes and their values available in the User Profile. It also represents the abstract structure of User Preference Sets as shown in Figure 42. This structure allows the creation of an unlimited number of Preference Sets where the users can express their Preferences in the context of specific issues (e.g. Preferences in travelling).

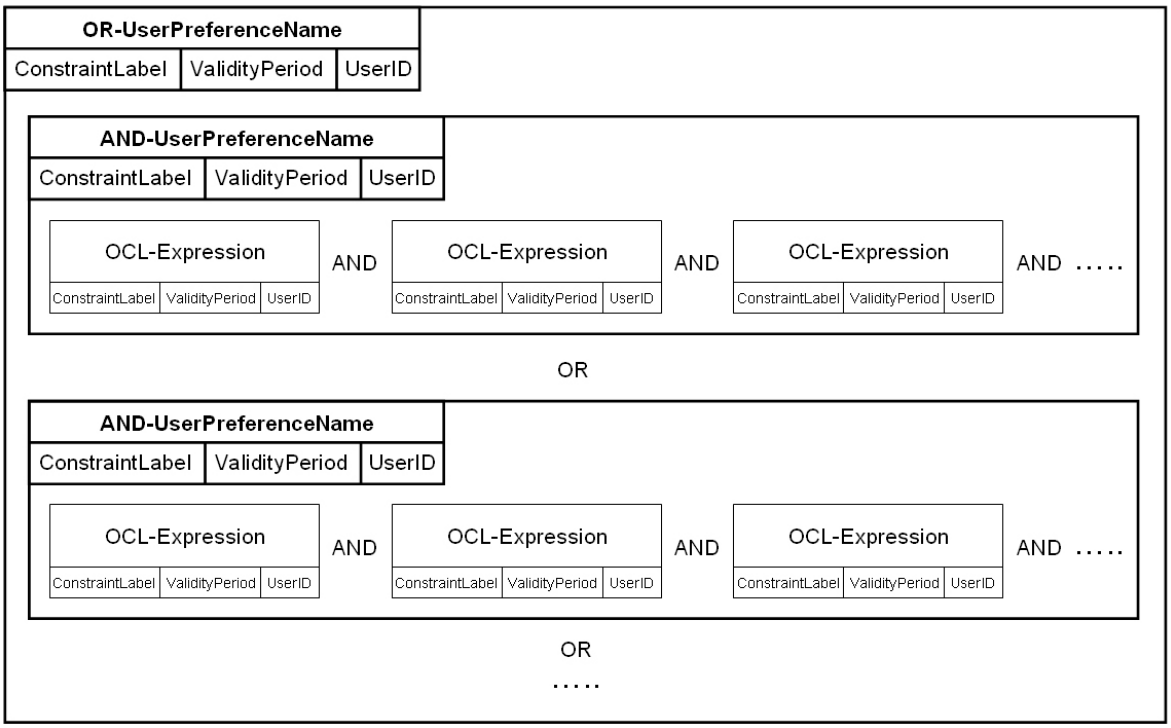



*Figure 42: Visualisation of a Preference Set*

The MOF-based approach allows the representation of a User Profile as XMI files (XML Metadata Interchange). The open and vendor independent format allows the exchange of objects based on Meta-Metamodels following aforementioned MOF-approach. The User Profile has this characteristic with the consequence that all components following the same MOF-based approach (like for example the knowledge base, KB) can easily interact and interchange data with the User Profile. Therefore User Profiles are stored as complex XMI files. An example of how a User Profile XMI file could look like is presented in the Appendix 6.2 of this paper. The relevant user specific information is highlighted in this example.

- **Exact Definition of the term “User Profiling” important**
  In the course of the project the term User Profiling caused some discussions as people interpret this expression quite differently. Therefore it was necessary to define explicitly what the understanding was for this term in order to clarify the work of work package 7 and to dispose any confusion connected to it. User Profiling in this project has been treated as the composition of relevant components working together to represent a user’s requirements and needs properly. Components as the user friendly GUI, the generally accepted Meta Object Facility (MOF) model for User Profiling purposes, search algorithms (e.g. keyword search) and techniques for realising general User Profile features have been considered to be important as part of an integrated User Profiling approach.

- **Extensive use of the User Profile necessary for profiling algorithms**
  The employment of several profiling algorithms has been researched by concept in Deliverable D7.3 and considers the state of the art algorithms in gathering information about a user. As this has not been the main focus (see chapter 1) of the current User Profiling approach further implementation aspects of these algorithms where out of scope. Nevertheless it became clear that all algorithms need a certain amount of information from the users and their behaviour in interacting with the DBE system. And this might be the risk as users could probably setup their profile once, add some Preferences and search for some services in the beginning. After a service is found, it might be possible that the user changes nothing within the profile anymore and no interaction will take place with the system to generate further information for respective algorithms. Personalisation is a hot topic within the research community and it surely is a good staring point for further research if the definition of User Profiling is merely algorithm-based

- **User Profile as resource for other DBE services**
  The User Profile is the most important resource for other services and applications to retrieve information about a user. A service like the Recommender or the Evolutionary Environment project within the DBE needs User Profile data provided by the user to maximise performance and quality of their respective features. Therefore it is necessary that the User Profile contains as much information from and about the user as possible to provide a comprehensive view on the user’s preferences and needs.

- **Expression of comprehensive User Preferences for business services useful**
  The User Profiling approach work package 7 has followed in this project is quite different to other (mostly quite simply) attempts to develop profiles. The approach is completely model-driven and considers the paradigm of the Model Driven

Architecture (MDA) [DeTo05]. As a result it is possible to search for business models on a MDA-Level 1 as well as to search and retrieve concrete services based on User Information coming from MDA-Level 0. So the users can express their business preferences adequately by concatenating an arbitrary number of attributes and values. Nevertheless it has been quite challenging and interesting to adapt and transfer this model-based approach from the DBE architecture into this new kind of User Profile implementation.

# 6 APPENDIX

## 6.1 RESOURCES

### 6.1.1 User Profile for the DBE Portal

The User Profile project is an open-source development project and can be found in the repository of Sourceforge.net. The OpenLaszlo[7] based User Profile application for the DBE Portal, as shown in Figure 43, is available for public access[8].

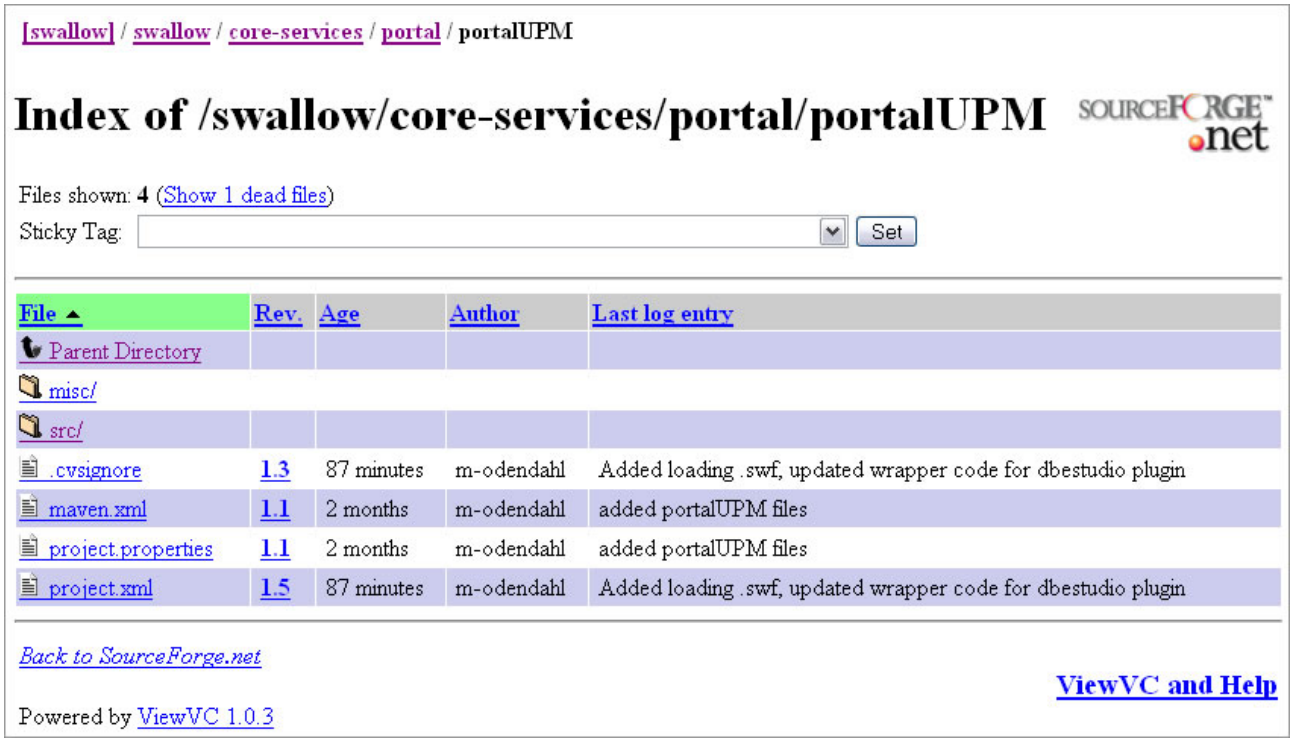



*Figure 43: Repository for the DBE Portal User Profile (Sourceforge Screenshot)*

[7] "OpenLaszlo is an open source platform for creating zero-install web applications with the user interface capabilities of desktop client software. The programs are written in XML and JavaScript and transparently compiled to Flash and DHTML." (http://www.openlaszlo.org)

[8] http://swallow.cvs.sourceforge.net/swallow/swallow/core-services/portal/portalUPM/

### 6.1.2 User Profile for the DBE Studio

The Eclipse[9] based User Profile application for the DBE Studio, as shown in Figure 44, is also available for public access[10]

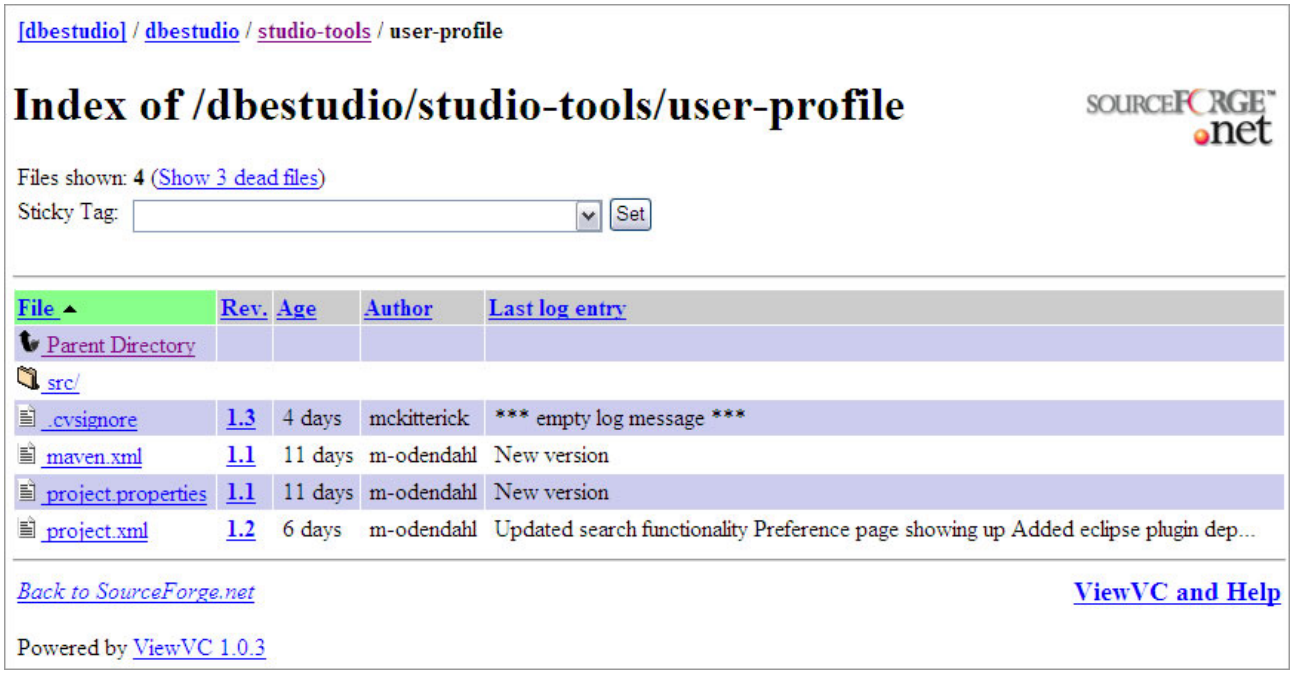



*Figure 44: Repository for the DBE Studio User Profile (Sourceforge Screenshot)*

[9] "Eclipse is an open source community, whose projects are focused on building an open development platform comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the lifecycle." (http://www.eclipse.org)

[10] http://dbestudio.cvs.sourceforge.net/dbestudio/dbestudio/studio-tools/user-profile/

## 6.2 XMI EXAMPLE – USER PROFILE MODEL (EXCERPTION)

```
<?xml version = '1.0' encoding = 'windows-1252' ?>
<XMI xmi.version = '1.2' xmlns:Model = 'org.omg.xmi.namespace.Model' timestamp 'Fri Dec 22 18:04:06 CET 2006'>
 <XMI.header>
 …
 <XMI.content>
  <UPM.Core.UserIdentity xmi.id = 'a1' UserID = 'martha@example.tld'>
   <UPM.Core.UserIdentity.userInformation>
    <UPM.Core.UserInformation xmi.id = 'a2' UserNameDBE = 'martha@example.tld' BusinessName =
                'Martha Example' StreetAddress = 'marthastreet' ZipCod Adress = '00000' CityAddress =
                'Example City' CountryAddress = 'dbeland' Fax = '00000' E-Mail = 'martha@example.tld'
                WebURL = '' Description = 'DBE Example User'/>
   </UPM.Core.UserIdentity.userInformation>
   <UPM.Core.UserIdentity.userPreferences>
    <UPM.Preferences.OrPreference xmi.id = 'a3' isValid = 'false' usageCount = '0' importance = '0.0'/>
   </UPM.Core.UserIdentity.userPreferences>
    …
  <Model:Operation xmi.id = 'a59' name = '=' isQuery = 'false'/>
  <QML.Ocl.Expressions.IteratorExp xmi.id = 'a60' name = 'exists' annotation 'Rooms' isMarkedPre = 'false'>
   <QML.Ocl.Expressions.PropertyCallExp.source>
   …
              <QML.Ocl.Expressions.OperationCallExp.arguments>
               <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a68' isMarkedPre = 'false' stringSymbol = 'Rooms'/>
              </QML.Ocl.Expressions.OperationCallExp.arguments>
              …
         <QML.Ocl.Expressions.OperationCallExp.arguments>
          <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a71' isMarkedPre = 'false' stringSymbol = 'Price'/>
         </QML.Ocl.Expressions.OperationCallExp.arguments>
         …
       <QML.Ocl.Expressions.OperationCallExp.arguments>
        <QML.Ocl.Expressions.IntegerLiteralExp xmi.id = 'a75' isMarkedPre = 'false' integerSymbol = '30'/>
       </QML.Ocl.Expressions.OperationCallExp.arguments>
       …
  </QML.Ocl.Expressions.IteratorExp>
  <QML.Ocl.Expressions.IteratorExp xmi.id = 'a76' name = 'exists' annotation = 'Hotel' isMarkedPre = 'false'>
   <QML.Ocl.Expressions.PropertyCallExp.source>
   …
              <QML.Ocl.Expressions.OperationCallExp.arguments>
               <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a83' isMarkedPre = 'false' stringSymbol = 'Hotel'/>
              </QML.Ocl.Expressions.OperationCallExp.arguments>
               …
         <QML.Ocl.Expressions.OperationCallExp.arguments>
          <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a86' isMarkedPre = 'false' stringSymbol = 'HotelName'/>
         </QML.Ocl.Expressions.OperationCallExp.arguments>
          …
       <QML.Ocl.Expressions.OperationCallExp.arguments>
        <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a90' isMarkedPre = 'false' stringSymbol = 'Hilton'/>
       </QML.Ocl.Expressions.OperationCallExp.arguments>
        …
  </QML.Ocl.Expressions.IteratorExp>
  <QML.Ocl.Expressions.IteratorExp xmi.id = 'a91' name = 'exists' annotation = 'SkiSchool' isMarkedPre = 'false'>
```

```
  <QML.Ocl.Expressions.PropertyCallExp.source>
   …
            <QML.Ocl.Expressions.OperationCallExp.arguments>
              <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a99' isMarkedPre = 'false' stringSymbol = 'SkiSchool'/>
            </QML.Ocl.Expressions.OperationCallExp.arguments>
             …
      <QML.Ocl.Expressions.OperationCallExp.arguments>
        <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a106' isMarkedPre = 'false' stringSymbol = 'Ski'/>
      </QML.Ocl.Expressions.OperationCallExp.arguments>
      …
  </QML.Ocl.Expressions.IteratorExp>
  <QML.Ocl.Expressions.IteratorExp xmi.id = 'a107' name = 'exists' annotation = 'SkiLesson' isMarkedPre = 'false'>
   <QML.Ocl.Expressions.PropertyCallExp.source>
   …
            <QML.Ocl.Expressions.OperationCallExp.arguments>
            <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a114' isMarkedPre = 'false' stringSymbol = 'SkiLesson'/>
            </QML.Ocl.Expressions.OperationCallExp.arguments>
             …
        <QML.Ocl.Expressions.OperationCallExp.arguments>
          <QML.Ocl.Expressions.StringLiteralExp xmi.id = 'a117' isMarkedPre = 'false' stringSymbol = 'Price'/>
        </QML.Ocl.Expressions.OperationCallExp.arguments>
        …
      <QML.Ocl.Expressions.OperationCallExp.arguments>
        <QML.Ocl.Expressions.IntegerLiteralExp xmi.id = 'a121' isMarkedPre = 'false' integerSymbol = '50'/>
      </QML.Ocl.Expressions.OperationCallExp.arguments>
  </QML.ContextDeclarations.InvariantContextDecl>
  <UPM.Preferences.AndPreference xmi.id = 'a130' isValid = 'false' usageCount = '0' importance = '0.0'
                                                                  name = 'DBE_Hotel:::Hotel Price'>
   <UPM.Preferences.UserPreference.constraints>
    <UPM.Preferences.ConstrainedLabel xmi.id = 'a131' isValid = 'true' usageCount = '0' importance = '74.0'>
     <UPM.Preferences.ConstrainedLabel.constraint>
      <QML.ContextDeclarations.InvariantContextDecl xmi.idref = 'a126'/>
     </UPM.Preferences.ConstrainedLabel.constraint>
    </UPM.Preferences.ConstrainedLabel>
    <UPM.Preferences.ConstrainedLabel xmi.id = 'a132' isValid = 'true' usageCount = '0' importance = '50.0'>
     <UPM.Preferences.ConstrainedLabel.constraint>
      …
  </UPM.Preferences.AndPreference>
  <UPM.Preferences.AndPreference xmi.id = 'a133' isValid = 'false' usageCount = '0' importance = '0.0'
                                                                  name = 'DBE_Hotel:::SkiHotel'>
   <UPM.Preferences.UserPreference.constraints>
    <UPM.Preferences.ConstrainedLabel xmi.id = 'a134' isValid = 'true' usageCount = '0' importance = '50.0'>
     <UPM.Preferences.ConstrainedLabel.constraint>
      <QML.ContextDeclarations.InvariantContextDecl xmi.idref = 'a128'/>
     </UPM.Preferences.ConstrainedLabel.constraint>
    </UPM.Preferences.ConstrainedLabel>
    <UPM.Preferences.ConstrainedLabel xmi.id = 'a135' isValid = 'true' usageCount = '0' importance = '50.0'>
     <UPM.Preferences.ConstrainedLabel.constraint>
      …
  </UPM.Preferences.UserPreferenceGroupAssoc>
 </XMI.content>
</XMI>
```

# 7 REFERENCES

[Admo06] Admonds, A.: DBE Portal Specification. DBE-Project (http://www.digital-ecosystem.org), Deliverable D26.6, 2006.

[Bart05a] Bartsch, C.: Description of necessary information about DBE customer, to support a long term evolutionary business relationship. DBE-Project (http://www.digital-ecosystem.org), Deliverable D7.1, 2005.

[Bart05b] Bartsch, C.: Initial Description of Profiling mechanism design and rationale with respect to one or two use cases. DBE-Project (http://www.digital-ecosystem.org), Deliverable D7.2, 2005.

[Bart06] Bartsch, C.: Prototype Implementation of Profiling Mechanism. DBE-Project (http://www.digital-ecosystem.org), Deliverable D7.3, 2006.

[DeTo05] De Tommasi, M.: Business Modelling Language 1.0. DBE-Project, (http://www.digital-ecosystem.org), Deliverable D15.1, 2005.

[Ferr05] Ferronato, P.: Architecture Scope Document Release 2. DBE-Project, (http://www.digital-ecosystem.org), Deliverable D21.2, 2005.

[GiKM05] Gioldasis, N.; Kazasis, F.G.; Maragoudakis, Y.: DBE Knowledge Base Representation Models. DBE-Project, (http://www.digital-ecosystem.org), Deliverable D14.1, 2005.

[KoKK06] Kotopoulos, G.; Kotopoulos, Y.; Kazasis, F.G.: Final Release of the Recommender. DBE-Project (http://www.digital-ecosystem.org), Deliverable D14.6, 2006.

[Mevi06] Mevius, M.: Kennzahlenbasiertes Management von Geschäftsprozessen mit Petri-Netzen. Verlag Dr. Hut, München, 2006.

[McKi06] McKitterick, D.: 2nd Release of the DBE Studio. DBE-Project (http://www.digital-ecosystem.org), Deliverable D26.4, 2006.

[MOF06] OMG Modelling and Metadata Specifications: Meta Object Facility Core Specification Version 2.0. 2006. http://www.omg.org/technology/documents/formal/MOF_Core.htm, Recall: 10.01.2007.

[OCL06] OMG Modelling and Metadata Specifications: Object Constraint Language Specification Version 2.0. 2006. http://www.omg.org/technology/documents/ formal/ocl.htm, Recall: 10.01.2007.

[PaKK06] Pappas, N.; Kotopoulos, G.; Kotopoulos, Y.: Final P2P Implementation of the DBE Knowledge Base and SR. DBE-Project, (http://www.digital-ecosystem.org), Deliverable D14.5, 2006.

[SBMC05] Sacha, J.; Biskupski, B.; Meier, R.; Cunningham, R.: DBE Peer-to-Peer Architecture Design, (http://www.digital-ecosystem.org), Deliverable D24.3, 2005.

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

[Sche95] Scheer, A-W.: Wirtschaftsinformatik – Referenzmodelle für industrielle Geschäftsprozesse. Springer, Berlin, 1995.

[Scrm03] SearchCRM - Definitions: Personalization. 2003. http://searchcrm .techtarget.com/sDefinition/0,290660,sid11_gci532341,00.html. Recall 10.10.2006.

[UML05] OMG Modelling and Metadata Specifications: Unified Modelling Language. 2005. http://www.omg.org/technology/documents/ formal/uml.htm, Recall: 10.01.2007.

[XMI05] OMG Modelling and Metadata Specifications: MOF 2.0 / XMI Mapping Specification, V2.1. 2005. http://www.omg.org/technology/documents/ formal/xmi.htm, Recall: 10.01.2007.