Integrating Web Service and Semantic Dialogue Model for User Models Interoperability on the Web

Integrating Web Service and Semantic Dialogue Model for User Models Interoperability on the Web

Federica Cena

Abstract. Nowadays there is a great number of Web information systems that build a model of the user and adapt their services according to the needs and preferences maintained by the user model (UM). One of the most challenging issues of this scenario is the possibility to enable different systems to cooperate in order to exchange the available information about a user. Our aim is to create rich (and scalable) communication protocols and infrastructures to enable consumers and providers of UM data to interact. Our solution for dealing with such an issue is to exploit Web standards for interoperability (i.e. Semantic Web and Web Services) for implementing simple atomic communication, and a dialogue model for implementing enhanced communication capabilities. In particular, two systems can start a semantics-enhanced Dialogue Game as a form of negotiation to clarify the meaning of the requested concepts when a shared knowledge model does not exist, and to approximate the response when the exact one is not available. We propose a distributed semantic conversation framework based on the Sesame semantic environment for the exchange of user model knowledge on the Web. Systems have to expose their user model data as a Web Service, and to exploit a public dialogue knowledge base to start the dialogue. The main advantage of the approach is to allow systems to deal with difficult situations by starting an appropriate dialogue game instead of stopping the communication as in the traditional "all-or-nothing" Web Service approach. On the basis of a preliminary evaluation, the approach has shown an improvement of the adaptation results provided by the systems we tested.

***Keywords**. User Model, Interoperability, Web Service, Semantic Web, Interaction Model , Dialogue Model, Adaptation*

F. Cena

Department of Computer Science – University of Turin
Corso Svizzera 185 - 10149 Turin (Italy)
e-mail: cena@di.unito.it

1. Introduction

Personalized services for users become crucial when service access takes place in an open and dynamic environment like the Web, since they simplify the user interaction with applications and reduce information overload (Kobsa 2001). Nowadays in the Web many information systems build a model of the user and adapt their services and behaviour according to the user's needs and preferences maintained in a user model. Such user model-based intelligent information systems are known as "user-adaptive systems" (Brusilosvky 1996, Kobsa 2001). Brusilovsky (1996) defines adaptive systems as information systems which can access information (personal data, characteristics, preferences, etc.) gathered in a user model, and use such information to adapt different aspects of the system to the user.
A user model (UM) is a knowledge structure which represents the profile of a single user. A user model usually contains both a set of domain-independent features (e.g. personal information such as age, job, etc.) and a set of domain-related features, representing an *overlay model* of the domain (e.g. how much the user is familiar with a domain concept, or the user interest in the different domain concepts) (Brusilovsky 1996). The former features are usually derived from the registration data, while the latter ones can be derived from explicit user's ratings or can be inferred by the system (for example, tracking user's behavior and reasoning on it) (Balabanovic and Shoham 1997).

Nowadays, in the Web a single user may interact with many different user-adaptive systems. Each of them builds its own model of the user (Montaner et al. 2003). Currently most of them work as stand-alone systems and they are not aware of the existence of other systems in the environment. Thus, instead of cooperating to share the data they store about the user, they are often intrusive with respect to the user, asking her about each piece of information they need. Every time a user interacts with a system for the first time she has to provide data that she already provided to other applications. This makes the interaction between users and adaptive-systems very cumbersome. In fact, especially when the interaction context is critical (e.g., users are mobile, in a hurry, using small devices with small screens and variable bandwidth), users do not want to waste time to explicitly fill in their model for every single application they use (Vassileva 2001). Furthermore, from the systems' point of view, in isolated personalization, information about a user is not reused by other systems. There is no common "memory" of all user activities, preferences and characteristics, which would allow more effective and adequate adaptation to the user's current state. There is the risk that each application has only a partial view of the user and thus many useful data may be missing. Furthermore, a lot of data on a specific user (e.g. characteristics, preferences, knowledge, interests, goals, activities) are replicated over many applications on the Web (Lorenz 2005, Aroyo et al. 2006).

These considerations motivated us to draw a scenario where systems interact with each other in order to share the knowledge about the user contained in their user models, without directly involving the user in the process. This allows, on the one hand, preventing users to get bored in training new systems (Vassileva 2001), and speeding up the phase of user model initialization[1] (Kobsa et al. 2001). On the other hand, this also allows each adaptive system to reach a deeper understanding of the user (Heckman 2005). In fact, sharing user models enables applications to cover more aspects of the user profile, increasing at the same time the level of detail and the reliability of user data. This should lead to adaptation results that are more appropriate for the user (Vassileva 2001, Berkovsky 2005).

[1] This problem is also referred as "cold start problem", which refers to the difficulty for applications to start up the adaptation for new user since there is a lack of information about users in the first interactions (Salton and McGill 1984).

Thus, the overall goal is the *interoperability* of knowledge among user-adaptive applications on the Web. To interoperate system must be able to communicate (*syntactic interoperability* issue), and to correctly interpret the exchanged data (*semantic interoperability* issue) (Veltman 2001). To fulfill both syntactic and semantic interoperability is the *sine qua non* condition to ensure the exchange of knowledge among systems.

Our solution for dealing with such issues is to exploit i) Web standards for interoperability, i.e. Semantic Web and Web Services, for implementing simple communication, and ii) a dialogue model for enhancing the systems communication capabilities in case of difficult situations. More in detail, two systems can start a semantics-based dialogue in order to negotiate the meaning of the requested information (i.e. reasoning on concepts in order to find an agreement), when the request is ambiguous. System may also use dialogue to approximate the response (i.e. reasoning on concepts relations in order to find similar results), when the exact one is not available. Such complex situations occur very frequently during the exchange of UM knowledge in an open environment like the Web (Aroyo et al. 2006, Heckman 2005).

The paper is structured as follow. We first present (Section 2) a scenario in order to motivate the relevance of the discussed problem. Then, in Section 3 we present our approach for solving the problem, describing the model in details. An architecture for the model is provided in Section 4. Section 5 describes the scenario implemented with our interaction model. Section 6 presents some evaluation results, and in Section 7 describes the state of the art in order to have an overview of the existing solutions for the interoperability problem in user model context. Finally, Section 8 concludes the paper discussing the benefits and limitations of the proposed approach.

2. Application scenario

Let us start presenting an application scenario which exemplifies the user models interoperability problem in the Web.

Peter, a University student, is visiting Turin. He interacts with his mobile adaptive tourist guide UbiquiTO[2] (Cena et al. 2006) to gather quickly more information about churches to visit (e.g. locations, creators, styles). In order to provide an answer tailored to Peter's personal preferences and the context of interaction, UbiquiTO needs some information, e.g. information on buildings Peter visited in the past; his expertise in religious art; his interest in the topic; other related topics he is interested in; his constraints in terms of time and locations. UbiquiTO stores a user model of Peter but unfortunately the model does not contain all the necessary information to provide effective adaptation: in particular it does not have information about how much Peter is interested in churches from an artistic point of view. Thus, the system looks for other Web systems used by Peter as well. It finds iCITY[3] (Carmagnola et al. 2008), and it decides to ask it for the needed features. UbiquiTO and iCITY first agree on Peter's identity. Then, UbiquiTO asks iCITY for the values for Peter's interest in "churches". To provide the correct value, iCITY needs to *(i) clarify* the context of the request -i.e., find out whether "church" is considered as a

[2] UbiquiTO is an adaptive application that provides content-based recommendations in tourist domain.

[3] iCITY is a social web-based, multi-device recommender system. It provides suggestions on cultural events in the city of Turin, and allows users to insert new events, to add information about events, to insert comments and tags.

sightseeing site or as a place for attending religious events. Thus, it starts negotiating the concept with UbiquiTO, in order to disambiguate the meaning.
Once the feature has been disambiguated, iCITY is able to discover that its repository does not store a value for such a feature. Instead of stopping the communication as commonly happens, UbiquiTO asks for Peter's interests in other concepts somehow related to the starting one, such as more general concepts. In this way, UbiquiTO is able to *(ii)* obtain an *approximate* answer even if the exact one is not available.

The problem underlined in the scenario described above is how to enable Web-based adaptive applications to interoperate in order to exchange knowledge contained in their user models. In principle, the communication model for supporting the exchange of UM knowledge could be quite simple: a requestor asks for some user model feature and a responder provides the requestor with the value of the required feature. When the object of the request is a concept from a shared knowledge model, a well-defined and structured form of *atomic communication* (request/response) could be sufficient to reach UM interoperability.

However, as the scenario highlights, the exchange of user model data in an open environment like the Web may require more complex interactions, since it can require a very high degree of alignment between the applications. In particular, as emerges from the scenario, there are two main typologies of problems, dealing with ambiguous request or missing response.

Ambiguous request. The problem is exemplified by the situation (i) in the scenario**.** This problem is particularly relevant when the systems do not share the same knowledge model, as it is common in an open environment as the Web, where applications are independently built for different purposes by different organizations. The heterogeneity of user-adaptive applications makes it impossible and unrealistic to have a unified domain model (such as an ontology). Furthermore, even if two systems share the same knowledge model, their internal knowledge organization in repositories could only partially overlap, since the internal knowledge can be structured according to the specific system's needs (Laera, 2006). This can cause some ambiguities in the meaning of the concepts.

Thus, we need some mechanism to disambiguate the exchanged concepts, i.e. to reach an agreement on the exact meaning of the requested features. In case of UM interoperability, to disambiguate a concept means that both systems must be sure that they are referring to the same user model concept. In particular, for domain-related concepts (such as "interest(art)"), it means to refer to the same property of the user model ("interest") and to the same concept in the domain model ("art").

Missing response. This problem, exemplified in the situation (ii) of the scenario, covers two cases. The first one occurs when the value of the requested feature is not available in the provider's knowledge base. The second case occurs when the provided value is not satisfactory for the requestor (for example the level of certainty of the value is too low). Thus, we need some mechanism to approximate the response, e.g. to derive in some way the desired concept starting from some related concepts (more general or more specific concepts).

In both situations, atomic communication is not sufficient. In these cases, the exchange of user model data requires more complex typology of communication among systems.

3. A Model for User Model Interoperability on the Web

The solution we propose for dealing with the problems described in the previous section is to exploit *Web standards* for interoperability (Semantic Web and Web Service) for implementing a simple atomic communication, and a *dialogue model* in order to enhance the communication capabilities of the systems.

A *dialogue* represents a communicative action between two agents interacting with a specific goal. A dialogue is an evolution of traditional messaging mechanism used by distributed systems to communicate (Wooldridge, 1966), enriched with some aspects derived from human conversational conventions, such as flexibility, context-dependence, mixed-initiative, action-reaction mechanism, turn-talking management (Levinson 2001). For its characteristics, a dialogue can be useful to support both syntactic and semantic interoperability issues.

In the following, we will present separately our solutions to reach syntactic and semantic interoperability of UMs in the Web.

Syntactic interoperability. Our model for syntactic interoperability of UMs is based on i) *Web Service* technology for implementing atomic communication, and ii) *dialogue techniques* on top of Web Service infrastructure when more complex interactions are necessary.

The loosely coupled structure and the well accepted stack of standards underlying *Web Services* represent a solution to the integration of UM knowledge and personalization services from various systems. The use of Web Service techniques for reaching UM interoperability means that user-adaptive systems must make their UMs available as a public interface described in the standard language WSDL[4], accessible via HTTP by means of SOAP[5] messages. Thus, each user-adaptive system provides a sort of *user model service* specifying in a WSDL file the user model features the system provides and the methods to access them. For instance, the UM of a recommender system could be extended with a public interface that enables other systems to invoke its user model features. With this approach, a software agent which needs user features can search for Web Services which provide such features for a specific user and decide to invoke the service which better satisfies its needs.

Web Service interface and the emerging service publication standards support only the specification of static interfaces of elementary services. They allow only simple "one-shot" interactions, structured as "request–response" pairs, but they are not expressive enough to define conversations including more than two turns (Ardissono et al. 2004). In fact, such standards cannot support rich interactions requiring a more proactive, dynamic modality of communication with complex patterns of interaction (Hanson et al. 2004). In problematic situations they usually stop the communication. Instead, in our model, when a communication problem occurs, systems may play a *dialogue* in order to start a negotiation. In particular, by means of a dialogue, systems can approximate the response when the exact one is not available or can find better results when the available ones are not reliable enough, as explained later.

Semantic interoperability. Our model for semantic interoperability of UMs is based on the use of i) *Semantic Web* languages to guarantee common basic understanding among systems, and ii) *dialogue model* when a negotiation of meaning is needed.

Resources from the *Semantics Web* (Berneers-Lee at al. 2001), such as ontologies, metadata and specification languages, are based on open standards and have been

[4] WSDL - Web Service Description Language (http://www.w3.org/TR/wsdl)

[5] SOAP - Simple Object Access Protocol (http://www.w3.org/TR/soap/)

developed to allow common understanding among distributed systems in open environments. For these reasons, in our model, we exploit Semantic Web techniques to solve semantic interoperability. This implies that the user-adaptive systems must represent the knowledge in the user model (and in the domain model related to the user model) with standard xml-based formal languages (like RDF[6], RDFS[7], OWL[8]) , and they must organize this knowledge in structures such ontologies which must be public and shared among applications. Even though in the last years suppliers and consumers of user profiles have shown increased awareness of the need for standards for representing and exchanging user profile data, the heterogeneity of user-adaptive applications in the Web makes the use of such standards very difficult (Heckmann 2005). Currently, in such an environment there cannot be a single universally shared ontology that is agreed upon by all the parties involved. Instead, every system typically uses its own private ontology, which may not be understandable by other applications. Thus, some form of alignment among ontologies is needed. When some problems occur, systems should be able to negotiate the meaning. In our model, we propose the use of dialogue to solve such problems.

In summary, dialogue plays crucial role in our interaction model, to solve both semantic and syntactic interoperability. In the following section, we will focus on the detailed description of the proposed dialogue model.

3.1 The Dialogue Model

As a conceptual model of the interaction, we have applied the dialogue model introduced by Dimitrova (2001), based on multi-agents[9] dialogue models of the *Speech Acts* (Searle 1972) and the *Dialogue Game* (Levin and Moore 1977).

The *Speech-Act* theory[10] considers communication as an action: a Speech Act can change the state of the environment in the same way physical actions do. A dialogue is a set of Speech Acts performed by speakers with the intention to accomplish certain purposes. Every communication requires that the participants apply some decision making mechanisms to organize the dialogue in order to achieve their goals.

Levin and Moore (1977), starting from interactions between humans, proposed a theory that describes regularities in conversational interactions as *Dialogue Games*[11]. Dialogue Games are knowledge structures for representing multiple turn dialogue patterns organized around a specific goal.

[6] RDF - Resource Description Framework (http://www.w3.org/RDF/)

[7] *RDFSchema* (http://www.w3.org/TR/rdf-schema/)

[8] OWL - Web Ontology Language (http://www.w3.org/2004/OWL/)

[9] A multi-agent system (MAS) is a system composed of several agents, collectively capable of reaching goals that are difficult to achieve by an individual agent. “The characteristics of MASs are that (1) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (2) there is no system global control; (3) data are decentralized; and (4) computation is asynchronous” (Sycara, 1998).

[10] Speech-Act-based approaches are widely used to describe system-to-system interaction since, on the one hand, they represent the interaction at a conceptual level, independently of the physical implementation of the systems. On the other hand, they allow considering the use of a language by a system like any other action that it might take, simplifying the interaction management.

[11] A Dialogue Game defines “the kind of language interactions in which people engage, rather than the specific content of these interactions” (Levin and Moore 1977). Thus, the dialogue game approach is particularly useful to model system-to-system interaction (Jarred et al. 2003, McGinnis at al. 2003, McBurney and Parsons 2002), allowing to separate the management of “how” the interaction occurs from the management of exchanged messages.

Dimitrova (2001) put together these two interaction paradigms and presented a model for human-computer interaction in the e-learning field. This is a closed educational environment, with one student interacting with one system at a time. It is very different from the context of UM interoperability in open environment like the Web where many systems and many users can mutually and contemporarily interact with each other. Thus, we extended the approach by adding features needed in such a context, and omitting some features specific to educational task (see Cena and Aroyo 2007 for a description of the changes in the model).

In the following we present the definition of the main components involved in the model: *communicative acts*, *dialogue rules*, and *dialogue games*.

3.1.1 Communicative Acts

A Communicative Act (CA) (Baker, 1994)[12] consists of a proposition representing the intention of the system. Each participant in a dialogue produces CAs at its turn. A CA is defined as a quadruple <*Requestor, Responder, Move, Statement*> where the *Requestor* and the *Responder* identify the dialogue participants, the *Move* is a performative verb describing the illocutionary force[13] of the CA, and the *Statement* expresses a claim over the knowledge models.

Requestor/Responder. In the context of UM interoperability, the *requestor* is the user-adaptive system asking for information about a user, while the *responder* is the provider of UM data.

Statement. The requestor and the responder communicate by means of statements over concepts of the knowledge models (user and domain models). Statements are dependent on the specific application domain, but independent of the representational language used for the knowledge representation. A statement on a knowledge model may be represented in different ways, for example using *conceptual graph* (Sowa 1984) like in Dimitrova (Dimitrova 2002). We chose a representation by means of linear parameters (Rich 1983) as <*property(topic), value, belief*>, where *property* is a feature of the user model (and *topic* is a concept of the domain model[14] to which the property refers), *value* indicates the value assumed by the property, *belief* measures how much the system believes that its assumption about the specific value is correct.

Furthermore, in our model, statements have to be semantically expressed, i.e. expressed in a non ambiguous way (in formal semantics) and in a machine-processable way (in formal syntax) with some XML-based Semantic Web languages (such as RDFS, OWL), and have to refer to some shared model. Semantically expressing the statement is very useful in a dialogue since it provides a common understanding of the exchanged data among systems, but it also gives the systems the possibility to consider the semantics of data when deciding their behaviour in the dialogue. In other words, systems can reason with the hierarchy and the attributes of the knowledge models (*ontological reasoning*[15]) for identifying the concepts involved in the dialogue (see Section 3.1.3).

Move. Moves express the intention of the CA. Starting from the definition of moves by Dimitrova, 2001, we identify a set of moves appropriate for the UM interoperability context. By means of moves, a system may express the following intentions:

[12] To avoid association with exclusively verbal interactions, Baker (1994) proposes to use the term *Communicative Acts* (CA) rather than *Speech Acts.*

[13] The *illocutionary force* of a communication act is the intention it is exchanged (e.g. inform, inquire, deny) (Levinson, 2001).

[14] In case of simple domain-independent features (such as demographic features) the topic is *null.*

[15] Ontological reasoning means deducing subclass and superclasses of a concept, to find similar concepts and so on.

- to make a statement:
    - *inform*(s) – a system believes *s*;
- to suggest actions to perform over the statement:
    - *suggest(p,s)* - the requestor suggests p as a new conversation topic instead of *s*;
    - *modify(v,s)* - the requestor suggests a different value *v* for the property in the statement *s*;
- to make request about s:
    - *inquire(s)* - the requestor asks for the value of the property in the statement s or for the belief of this value;
- to express opinion on s:
    - *challenge(s)* - the requestor doubts the value or the belief of the property in the statement *s*;
    - *disagree(s)* - the requestor disagrees with the value in the statement *s*;
    - *agree(s)* - the requestor agrees with the value in the statement *s*;
- to explain *s:*
    - *justify(s)* - the requestor explains the value of the property in the statement *s*;
- to express the capability to provide s:
    - *deny(s)* - the requestor does not accept *s* as a topic of the discussion;
    - *accept(s)* - the requestor accepts *s* as a topic of the discussion;
    - *skip* - the responder skips its turn and passes the initiative to the requestor.

We conclude by providing some examples of Communicative Acts.
An example of a CA with a domain-related feature in the statement could be:

"system1, system2, inquire, ("interest(art)", value=?, belief=?)"

which means that system1 inquires system2 about its value and belief on the user model property "interest" in the domain concept of "art".
Instead, an example of a CA with a domain-independent feature in the statement could be

"system2, system1, challenge, ("school_level", value=0.7, belief=0.3)"

which means that system2 challenges system1's belief on the value of the user model property of *"school level"*.

3.1.2 Dialogue Rules

Dialogue rules define the communication protocol of the dialogue. They express the allowed moves in Communicative Acts and how to sequence them. A dialogue rule is defined as *$(move_1,statement_1)$ → $(move_2, statement_2)$* and postulates that a CA with move$_2$ and statement s$_2$ is permitted if the previous turn has included move$_1$ and statement s$_1$ (Dimitrova 2001). A dialogue rule does not specify the involved conversational participants.

Dialogue rules are domain and application-dependent: in our scenario of interoperability interaction between UASs, possible dialogue rules could be:

- *(inquire, s1)* → *(deny)* allows a participant to deny to provide a response to the request of another participant.
- *(inform, s1)* → *(challenge, s1)* allows a statement made by one agent to be challenged by another agent.

Dialogue rules are public and all the agents involved in the dialogue have to follow them in order to be able to communicate.

3.1.3 Dialogue Games

A *Dialogue Game* is a template defining the sequence of Communicative Acts to be exchanged for reaching a particular goal. Dialogue Games can be seen as “pre-programmed interaction patterns”, and they correspond to conversation policies as defined by Hanson et al. (2004). A dialogue game, from a conceptual point of view, is characterized by three parts (Dimitrova 2001): specifications, parameters, and components.

i) Specifications are the elements that remain constant during the game and are represented as predicates on the game parameters. They represent the *pre-conditions* of the game, i.e. the specific situation triggering a dialogue game.

ii) Parameters are the elements of a dialogue which can vary during the same game. They express:

- what the requestor wants to reach through the dialogue game (*goal*);
- a list of concepts and properties related to the topic that can be discussed during the game (*focus space)*.

iii) Components are the elements changing in a systematic way during a game. They are a set of sub-goals that determine the sequence of Communicative Acts to be generated during the game. They are defined by three typologies of *dialogue strategies***:**

- strategies to retrieve the necessary pieces of information for the construction of each dialogue game. There are two methods: *find_focus()* for retrieving all the concepts that will compose the “focus space” of the game, and *choose_scope()* for extracting from the set of concepts in the focus, i.e. the concepts to use as statement of each CA.
- strategies to determine the Communicative Acts to produce. For example, a system may apply the following rule: if the belief in the statement is under a particular threshold, the next CA must be a challenge of the concept.
  if (belief < 0.7) then
  {CommunicativeActs_List = add <q,r,challenge,c>,CommunicativeActs_List}
- strategies to determine how the dialogue participants’ knowledge base changes at the end of the dialogue (*post-conditions).*

The first typology of strategies strictly depend on the goal of the game: each Dialogue Game implies specific strategies to collect the information. They are the same for all the dialogue participants and must be public.

The latter two strategies, instead, depend on the specific internal policies of each system, since every system can implement its specific strategies. Thus, they are kept private.

All the elements of the dialogue game model are represented in Figure.1.

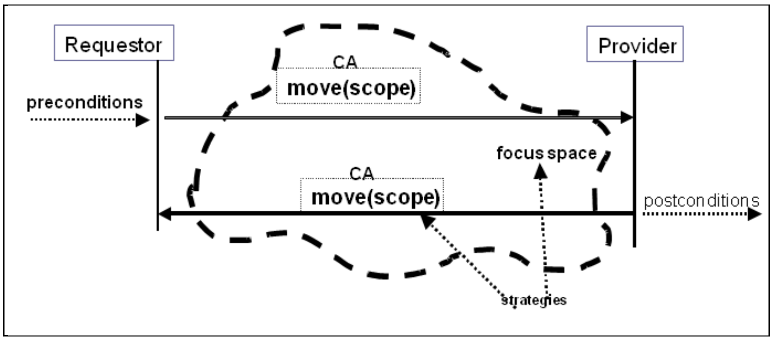

Figure 1. A graphical representation of the dialogue game model

The formalization of the dialogue game can be expressed also as a 5-tuple (Aroyo et al. 2005)

<P,U,C,S,R>

where

- *P* are the *pre-conditions* necessary to trigger the game;
- *U are* the *strategies* for selecting the concepts to exchange in the game;
- *C* is the *focus*, the set of all possible concepts that can be exchanged during the dialogue;
- S are the *scopes*, a subset of focus concepts;
- *R* are the *post-conditions*, the actions to perform at the end of the game.

Dialogue Games are the building blocks of the model, since they are the templates describing the communication behaviour the systems can follow to reach a particular goal. All the information about a dialogue is maintained in a public template called *dialogue schema*, containing the preconditions P, and the specific strategies (U) for the construction of the focus space (C) and the choice of the statement (S). We decided not to maintain public neither the strategies for producing the CA nor the post-conditions strategies, leaving each system free to implement its own strategies to decide what to do after the end of the game.

Dialogue Games for UM Interoperability

As seen in the scenario in Section 2, "ambiguous request" and "missing response" have been identified as the two main problems in the context of UM interoperability. They correspond to the goals of (i), "clarifying a request" and of (ii), "approximating the response". Thus, two specific *dialogue games* have been defined to reach such goals (Cena and Aroyo 2007, Cena 2007) [16]:

- an *explorative game* supports the *approximation of concepts* by collecting information about the *concepts* and *relations* in the knowledge base. The requestor can use this game to explore the user model of the responder in order to find an approximate answer, when an exact one is not available or the belief of the answer is too low. The rationale of the game is the use of related concepts in case the exact concept is not available;
- a *clarification game* supports the refinement of the request. The responder can use this game to know the context of the request in order to identify the granularity of the response in a "query refinement"[17] task (Stojanovic 2005). The rationale of the game is the use of concept properties to disambiguate the meaning of the requested topic, since two concepts sharing the same properties have a high probability of being the same concept.

In the following we describe in detail the *dialogue schema* that determines the content of the dialogue and the behaviour of the dialogue participants.

[16] In the UM interoperability context another goal has been identified as "to justify a response". The correspondent *explicative game* has the aim to clarify why a certain value or belief is present. This dialogue game starts when there is a discrepancy in the participants' beliefs that needs a justification. The focus of the game will be constituted by the concepts from which the value is derived. These concepts are here considered as *endorsements*, i.e. reasons for believing or disbelieving the statement to which they are associated (Cohen 1985). The evaluation of the final value depends on the evaluation of the intermediate values (Carmagnola and Cena 2006b). In the rest of the paper, we will not analyze this type of game. For a more detailed description, refer to Cena 2007.

[17] A "query refinement" task is the process of searching for queries that are more relevant for an entity's needs than the initial query, by specialization of the query's scope.

Explorative Game

An explorative game has the *goal* of exploring the conceptual space of a concept. This means finding all other related concepts and the values of their features. The situations which trigger the game *(preconditions)* are two: the responder does not store a value for the requested property, or the belief in the value provided by the responder is too low for the requestor. The *dialogue strategies* that determine the content of the explorative dialogue are the following ones:

- *find_focus.* The aim of the game is to approximate the response. The strategy is to retrieve the information related to the context of the concept, thus, the focus space will be composed by all the concepts related to the topic. In fact, the rationale of the game is that if the an exact match of the requested concept does not exist, the value of the properties of related concepts can be used instead. We can consider that if the user expresses interest in a subclass, this interest can be extended with high degree of certainty to the upper class, since children concepts are subclasses of a class and the subclasses inherit attributes of the upper classes. Thus, the interest of a user in “ancient art” can be assumed to be similar to the interest in its parent concept “art”. For example, if a starting topic is the concept of “cubism” (an artistic movement) a possible focus could be composed by other artistic movements with similar principles (e.g., “elementarism”); sharing the same historical period (e.g., “futurism”) , or all the concepts that are more generic than the starting one (for example, “ contemporary art”).
- *choose_scope.* The focus concepts are selected to be statements of CA according to their similarity with respect to the starting topic: the more properties with the same values they have in common, the more they are similar. In the selection of a scope, only the concepts that have the same upper concepts and properties in common are selected and ranked according to their similarity.

Table 1 represents the schema of the explorative game among a requestor (q) and a responder (r) exchanging statements in the form <p(c),v,b> . Notice that the notation *“v_q (p_c)”* in the table means “the value (v) of the property (p) of the topic (c) stored by the requestor (q)” .

Game structure for explorative game(c)			
Parameters	*Goal*	topic c	Explore a concept c
	*Focus**	*focus = find_related_concepts(c)*	Find all the concepts related to c
Specifications	*Pre-conditions*	*(v_r (p_c) = ’null’)* *OR* *(b_r (p_c) = ’low’)*	i) The responder does not store the value for the property p of the concept c. ii) The responder’s belief on the value is too low
Components	*Scope**	*scope = choose_scope (focus)*	Select concepts in the focus to be the scopes of the CA according to their similarity
	*List-CA***	*list_CA = assign_communicative_act(move, scope)*	Produce a list of CAs

Table 1. The structure of explorative game
(* The *find_related_concepts* and *choose_scope* methods are illustrated in Section 5.
** The list of CAs derived from the application of internal strategies)

At the end of the game tha system applies some post-conditions: for example, it could add to its repository the new values only if their beliefs are higher than the previously available ones.

Table 2 provides an instance of such a game where topic=cubism.

Parameters	*Goal*	(cubism)
	Focus	*focus=find_related_concepts(cubism)* *focus=[elementarism, naturalism, futurism, gothic, artistic_movement, painting]*
Specifications	*Pre-conditions*	*(v_r (interest(cubism)) = 'null')*
Components	*Scopes*	*s_1= elementarism, s_2= futurism..*
	List-CA	*List_CA= [< inquire, interest(s_1), v?>, ..]*

Table 2. An example of an instance of the explorative game

Clarification Game

A clarification game has the *goal* of disambiguating a concept starting from its properties. A concept is defined, in terms of the model, as a combination of properties: *concept(c) = <p1, p2, p3, .., pn>*. Two concepts can be considered the same concept if they have the same properties. Thus, in order to disambiguate the topic concept, the requestor asks the responder for the properties of the topic, in order to compare them with the properties it stores.

The *precondition* of this game is the statement of the CA containing a label which refers to more than a concept in the responder's knowledge base.

The *Dialogue Strategies* that determine the content of the clarification dialogue are the following:

- *find_focus.* The strategy is to disambiguate the concept starting from identifying its features. Thus, the focus will be constituted by the properties of the concept. For example, if we want to identify the concept of painting, a possible focus could be composed by all the properties of the concept: style, author, date of creation, title, techniques.
- *choose_scope.* The concept to be the topic in the statements is selected from the focus space according to the level of importance for identifying a concept: thus, statements are ordered from the ones with the highest importance to the lowest.

Table 3 represents the schema of the clarification game among a requestor (q) and a responder (r) exchanging statements in the form <p(c),v,b> .

Game structure for clarification game(c)			
Parameters	*Goal*	*c*	Identify a topic concept (c)
	*Focus**	*focus = find_properties(c)*	Find the properties of the concept (c)
Specifications	*Pre-conditions*	(c = x) AND (c = y)	The CA contains a label (c) which refers to more than a concept (x and y) in the responder's knowledge base

Components	*Scope**	*scope = choose_scope(focus)*	Select concepts in the focus to be the statement of the CA according to the level of importance
	*List-CA***	*List_CA = assign_communicative_act(move, scope)*	Produce a list of CAs

Table 3. The structure of clarification game
(* The *find_related_concepts* and *choose_scope* methods are illustrated in Section 5.
** The list of CAs derived from the application of internal strategies)

At the end of the game, the requestor updates the repository for the properties with the new value according to its internal strategies. Table 4 shows an example of a possible implementation of a clarification game for the concept rock in sense of music genre.

Parameters	*Goal*	(rock)
	Focus	*focus=find_properties (rock)* … *focus = [genre, artists, singers, period, concert_dates]*
Specifications	*Pre-conditions*	*[c(rock)= concept(C002)] AND [c(rock)=concept(C004)]*
Components	*Scope*	p_1= *genre*, p_2= *artists*, p_3= *period*
	List-CA	*<inquire,* p_1*, >* …

Table 4. An example of an instance of a clarification game

In the following section we describe the architecture of the framework we propose to implement the interaction model presented so far.

4. The Semantic Conversational Framework

We propose a semantic conversational framework for user model interoperability on the Web. The most important requirement for systems to partake in the framework is to expose the interface to access the user model as a Web Service. It is not required that adaptive systems are built as Web Service at all, but they only have topublish in a WSDL file the methods to access the user model knowledge.

The second requirement for systems to partake in the framework is to have a common understanding of exchanged statements. To do so, systems should express the knowledge model (the user model and the corresponding domain model) with a semantic languages (in RDFS[18]) and should refer to some external knowledge model (GUMO for user model and other domain-specific models).

We chose to develop a distributed peer-to-peer architecture to organize the interaction management. The main components of the framework architecture are i) a *semantic environment* to store the semantic knowledge models, ii) a UM Web Service, iii) some *dialogue management tools* (a public *dialogue knowledge base* , local *dialogue modules* and a local *dialogue working memory)*. Figure 2 represents the main components of the framework.

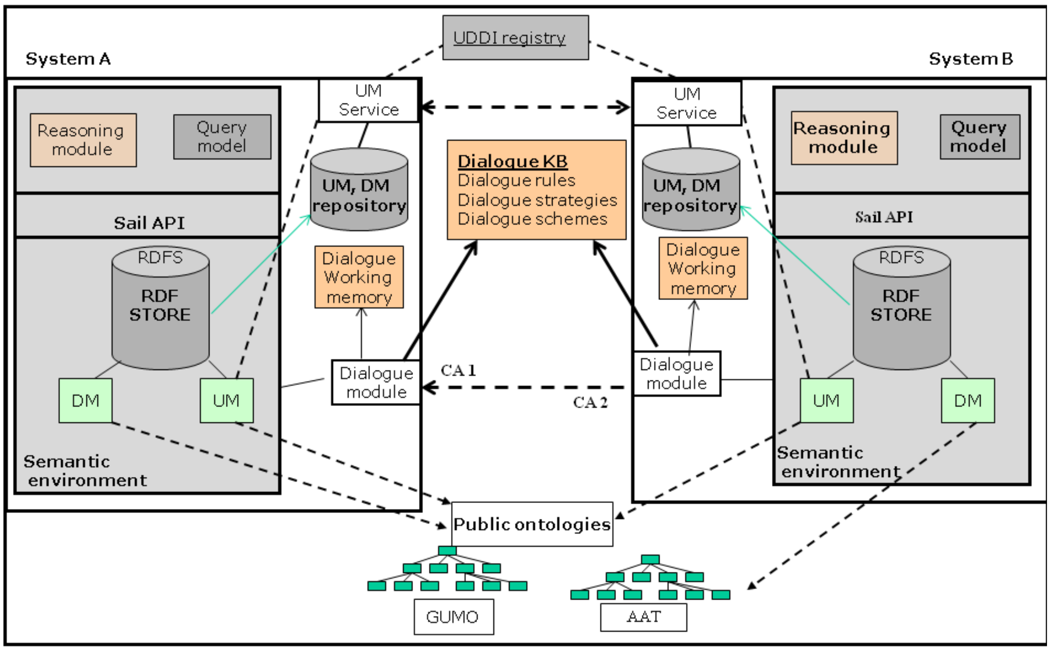

Figure 2. Architecture of the semantic conversation framework

4.1 The Semantic Environment

Systems need a semantic environment to store the semantic knowledge models, to query them, and to reason on them. We implemented our model within the semantic environment provided by Sesame-1.2.5[19]. The environment is composed by:

- a *RDF store*. It contains all the system knowledge about the user (UM) and the domain (DM) in the form of RDF/S data models (linked to some external public ontologies).
- the *SAIL (Storage And Inference Layer)*. SAIL is an API for 'wrapping' storage backend. It allows implementing Sesame on top of a wide variety of repositories. All the other Sesame functional modules (like the query module) are clients of the *SAIL API***.** This allows high portability of the framework to different systems.

[18] We chose the W3C recommendation standard for RDF/S since it is supported by Sesame environment and it is expressive enough for our purpose. In fact RDF-Schema allows to describe classes and properties of resources, to constrain properties in their domain and range, to build hierarchies of classes and resources. It has a formal semantics which supports basic reasoning capabilities like class and properties subsumption, and domain and range inference. This is sufficient for the goal of our model. Thus, we chose not to use OWL because in the trade-off between expressive power and efficiency of reasoning mechanism (Antoniou and van Harmelen 2004), we opted for efficiency.

[19] http://www.openrdf.org/

- a *SeRQL query model*. It supports declarative querying at the semantic level, about explicitly asserted and inferable statements. Sesame supports SeRQL[20], a RDF query language. Giving an RDF triple to query, SeRQL accesses the RDFS specific contents of the triple, retrieving all classes, properties or instances of a class (and also all instances of subclasses).
- a *reasoning module.* Sesame provides tools for reasoning capability on the knowledge models. This module allows: i) RDF Schema entailment (subsumptions of classes and properties that are not explicitly stated in the RDF schema); ii) class membership inference (with distinction between subclasses and direct subclasses); iii) classification of a concept (inserting a concept in the correct level of the hierarchy), iv) inheritance (inferring the properties of a concept based on properties of its parents); v) inverse reasoning (definition of relations in both directions, from domain to range and vice versa, e.g., to deduce from the statement "Mario owns a car" that "A car is owned by Mario"). All the more powerful reasoning capabilities needed for the internal tasks (e.g., user modeling and adaptation strategies) and interoperability interaction (e.g. strategies for producing Communicative Acts) have to be expressed by means of rules.

4.2 User Model Service

All systems need to publish the method to access their user model data as a Web Service interface. This means that the systems must express in the WSDL language the methods for accessing the user model values (e.g " *public String getValue(String userFeature, Int UserId)").* When a system wants the data in the User Model, it must implement such an interface.

4.3 Dialogue Management Tools

In order to be able to implement a more complex form of interaction, all the systems must refers to a shared *dialogue knowledge base*, containing all the knowledge needed to implement a dialogue game. Furthermore, every dialogue participant needs to locally maintain a component in charge of dialogue management (*dialogue module*), supported by a local *dialogue working memory*. In this way we avoid the use of centralized agents in charge of dialogue managing, differently from Levin and Moore (1977) and Dimitrova (2001) dialogue processing models. In our model, the functionalities for maintaining a dialogue are performed by an internal module of each user-adaptive system. This requires more computational efforts, but it gives more flexibility and freedom to each system.

Dialogue Knowledge Base. Systems must share the communication protocol of the dialogue (*dialogue rules*, Section 3.1.2), and the patterns for implementing the dialogue (libraries of *dialogue schemes*, Section 3.1.3). We store a representation of the knowledge the participants use during the dialogue, implemented as Java APIs, in a shared *dialogue knowledge base.* All the systems that want to participate in the dialogue only need to access this public knowledge base and execute the APIs to start a communication.

Dialogue Working Memory. It contains all the private strategies used in the dialogue, such as rules that compute run-time belief values for each concept according to context constraints - time, location - that will determine the future steps of the dialogue. The

[20] SeRQL – Sesame RDF Query Language (http://www.openrdf.org/doc/SeRQLmanual.html) combines the best features of query languages like RQL, RDQL and of data languages like N-Triples and N3, adding some other features: graph transformation using *construct*-queries; optional path expressions, nested optional path expressions.

Working Memory maintains also all the temporary dialogue results, such as the current level of belief of the dialogue participants for the discussed concepts, constraints from context that can have influence on the dialogue decision and the dialogue history, containing all the previously performed Communicative Acts. Moreover, the Working Memory stores the strategies for producing the Communicative Acts and for deciding which actions to perform at the end of the dialogue (post-conditions).

Dialogue Module. It is in charge of the following activities. First of all, it decides whether a dialogue is needed. Furthermore, it accesses the Dialogue Knowledge Base and chooses which game to play according to the trigger conditions. Afterwards, it exploits the dialogue strategies of the chosen dialogue game in order to retrieve the focus and the scopes concepts. Then, the Dialogue Module uses some internal private strategy to decide which Communicative Act is appropriate to perform. This strategy matches one of the possible moves (derived from the application of Dialogue Rules) with one of the possible statements (derived from the application of dialogue strategies). To do this match, it can take into account the previous CAs produced by the other dialogue participant and its beliefs.

The Dialogue Module produces CA which will be exchanged according to the communication protocol defined by dialogue rules in the dialogue knowledge base. It analyses the CA of the responder, and fills in parameters of the current dialogue situation of the Working Memory. At every step, it updates the Working Memory, and, according to the current context maintained in the Working Memory, it decides the next step in the dialogue.

5. Use Case as a Sequence of Dialogue Games in Our Interaction Model

In this section, we illustrate the scenario presented in Section 2 as a sequence of Dialogue Games, providing some implementation detail.

The starting point is Peter selection of "church" in the UbiquiTO interface. Since UbiquiTO does not have in its UM repository all the required information to satisfy Peter's request (in particular it lacks Peter's interest in churches), it accesses the UM Web Service interface of iCITY[21] (Section 4.2). Exploiting the public methods, it sends a SOAP message to retrieve Peter's value of the feature:

```
public String getValue(String interest, String church, String peter[22])
```

UbiquiTO and iCITY refer conceptually to the same public user model ontology (GUMO[23]) and to the Art and Architecture Thesaurus (AAT)[24] for the art domain. However, they structured their internal repositories in different ways according to their specific adaptation goals. Their repositories only partially overlap and some ambiguity in the concepts meaning is possible. In this case the request is ambiguous. "Church" is a label which in the UbiquiTO repository refers exclusively to a place to visit (i.e. a sightseeing site)(#001 in Figure 3). In iCITY repository, instead as it possible to see in

[21] Searching external systems does not influence the interaction management. Hence, we focused only on what happens when the system is found and the interaction starts. For a possible approach for systems discovery in User model interoperability context, see Cena and Furnari (2008)

[22] Before starting to exchange data UbiquiTO and iCITY should reach an agreement about the user identity, in order to be sure that they are speaking about the same user (see Carmagnola and Cena 2009)

[23] http://www.ubisworld.org/ubisworld/documents/gumo/2.0/gumo.owl

[24] http://www.getty.edu/research/conducting_research/vocabularies/aat/

the Figure 3, it can refer both to a sightseeing site (#C007) and to a place for attending religious events (#C003)[25].

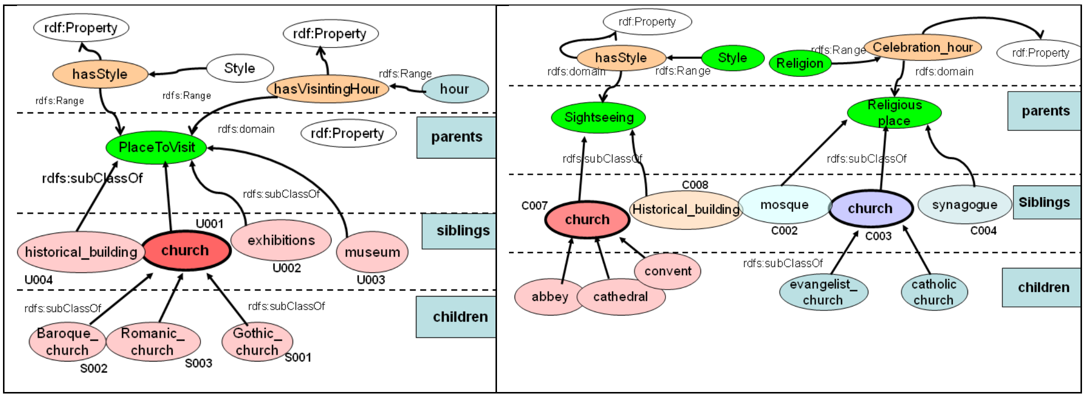

Figure 3. Difference in domain models of UbiquiTO (left) and iCITY (right)

iCITY accesses the Dialogue Knowledge Base (Section 4.3), and initiates a **clarification game** to clarify the request. Thus, it implements the strategies for collecting the information (the *find_focus* and *choose_scope* methods; Section 3.1.3) specific to such a game:

- *find_focus()*. In this dialogue game, the properties of the concept to disambiguate constitute the dialogue focus. Thus, iCITY queries (with SeRQL) the domain model in order to retrieve all the properties related to "church" as concept C007. In the following, the code for collecting the focus in an explorative game (topic = "church") is presented:

```
      public String findProperties(String topic) {
      String queryString = new String(
      "select properties from " +
      "{Topic} um_rdf_:has_" + property + "_" + {BlankNode}, " +
                  "{BlankNode} um_rdf_:has_properties {properties} " +
       "where User=http://www.di.unito.it/~cena/um.rdf#" + User +
        "using namespace " + Namespaces.NamespaceString);
-----------------------------------------------------------------------
Focus = [has_style, has_architect, has_visiting_hour, has_historical_period]
```

  As seen, the query results as focus: *has_style, has_architect, has_visiting_our, has_historical_period.*

  Then, iCITY performs the same query for collecting the properties of the concept C003 in its repository. The query results as focus: *has_celebration_hour, has_religious_view, has_priest_name.*

- *choose_scope()*. The properties to be the topic in the statement are selected from focus space according to the level of importance for identifying the concept: thus, the scopes are ordered from the highest importance ones to the lowest ones. To this purpose, iCITY checks the property "importance_factor" , which associates each property to a concept and indicates how much the property is important for identifying such a concept.

```
public  String  checkImportanceFactor(String User_dimension,  String  topic,
String User) {
String queryString = new String(
```

[25] Notice that each domain concept in the ontology is identified by means of a numerical code, but what is exchanged is the concept's label.

```
          "select CL from " +
          "{User} um_rdf_:has_" + User_dimension + "_" + topic + " {BlankNode}, " +
                    "{BlankNode} um_rdf_:has_CL {CL} " +
          "where User=http://www.di.unito.it/~cena/um.rdf#" + User +
            "using namespace " + Namespaces.NamespaceString);
 -----------------------------------------------------------------------
          Scopes = [s1 = has_style, s2 = architect, s3 = visiting_hour]
```

iCITY is now able to ask UbiquiTO whether it also stores the collected properties as well in its repository. Implementing its own strategies to produce Communicative Acts (see Section 3.1.3), it generates the following sequence of CAs:

CLARIFICATION GAME

iCITY : Do you have property = “has_celebration_hour” for church”? (<inquire move>)
UbiquiTO: No, I don’t. (<deny move>)
iCITY : Do you have property = “has_style” for church”? (<inquire move>)
UbiquiTO: Yes, I have. (<accept move>)

Then, according to its internal strategies, taking into account the number of properties in common, it determines if the concepts can be considered the same (for a detailed description of the algorithm, see Cena 2007). In this way, it is able to establish that the concept (U001) in the UbiquiTO’s repository refers with a high probability to the concept (#C003) in its repository since they share a large number of (important) properties. Table 5 illustrates the whole structure of such a game filled with the specific values.

Parameters	*Goal*	*“church”*
	Focus	*find_properties (church)* *focus(C001) = [has_style, has_historical period, has_architect, has_visiting_hour]* *focus(C003) = [has_celebration_time, has_priest_name, has_religious_view]*
Specifications	*Pre-conditions*	*(church)= concept(C001) AND (church)=concept(C003)*
Components	*Scope*	*p_1= has_style, p_2= has_architect, p_3= ..*
	List-CA	*<inquire, p_1, >* …

Table 5. Implemented structure of the clarification game

After having clarified that the concept refers to a typology of places to visit, iCITY can check its UM repository and discover that it does not store a value for Peter’s interest in it.

However, instead of stopping the communication, UbiquiTO accesses the *dialogue strategies* in the public Dialogue Knowledge Base to decide which dialogue game to play in this situation. It wants to achieve similar results, and this is a precondition that triggers an **explorative game**. Thus, UbiquiTO starts an explorative game for exploring the domain model, finding other possible related concepts to address the user request. It exploits the public strategies to follow for collecting information:

- *find_focus()*. To find the *focus* of the dialogue, UbiquiTO explores in its domain model the semantic space of the topic the feature refers to. It aims at gathering all the concepts related to the topic: all the ancestors, descendants and siblings concepts related to "church". Such concepts will constitute the dialogue focus. An example of the code of the strategy for collecting descendant (children) concepts is the following one (topic = church)

```
find_children(topic)
 "select children from " +
 "{children} serql:directSubClassOf {X} " +
 "where X = dm_rdf:" + topic +
 using namespace"+NamespaceString;}
-------------------------------------------------------
query results[26]
children = [abbey, cathedral, convent]
```

In this way, UbiquiTO has the complete view of the concept relations in the domain model, as it could be seen in Figure 4.

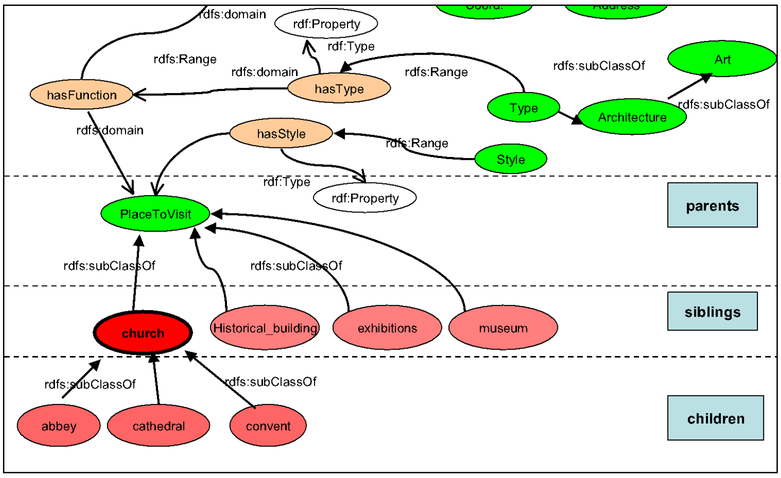

Figure 4. The extracted portion of the domain model representing the focus space of the explorative dialogue

○ *choose_scope()*. To choose scope concepts, UbiquiTO selects from the focus space concepts according to their similarity with the topic. The level of similarity depends on the number of properties in common. In the following the portion of the code for the extraction of the scope from focus is reported. Notice that *user_dimension* is "interest", *topic* is "church" and *user* is "Peter id".

```
public String checkSimilarity(String user_dimension, String topic, String
user) {
String queryString = new String(
"select CL from " +
"{User} um_rdf_:has_" + User_dimension + "_" + topic + " {BlankNode}, " +
           "{BlankNode} um_rdf_:has_CL {CL} " +
"where User = http://www.di.unito.it/~cena/um.rdf#" + User +
  "using namespace " + Namespaces.NamespaceString);
----------------------------------------------------------------------------
Scopes = [s1 = cathedral, s2 = abbey, s3 =convent
s3 = cathedral, s4 = historical_buildings, s5 = place_to_visit,]
```

At this stage, UbiquiTO can ask for Peter's interest in all the related concepts ranked by similarity with the starting topic "church". The most similar concepts are the children concepts since they share exactly the same properties. Thus, UbiquiTO performs the first CA, starting from the children concepts (e.g., abbey, cathedral, convent).

[26] It is a declarative querying at the semantic level, since giving an RDF triple to query, SeRQL accesses the RDFS specific contents of the triple, retrieving all classes, properties or instances of a class.

Then, it continues inquiring for the interest values of the sibling concepts (e.g., "historical building", "museum", "exhibition", etc). Among them the most similar with "church" is "historical_building", since they share the property "has_style". Figure 5 shows the strongest relation among "church" and "historical building" with respect to "church" and "exhibition".

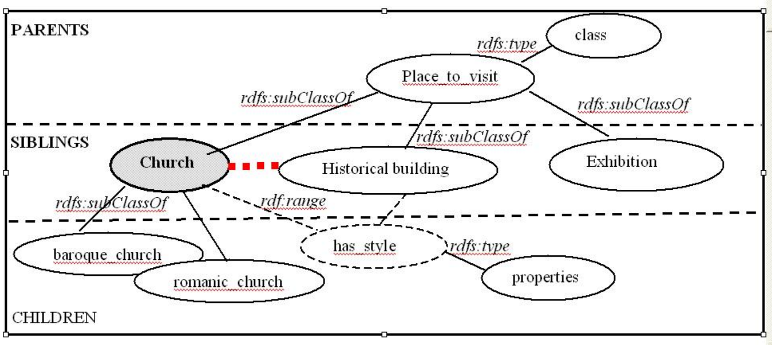

Figure 5. A portion of the domain model showing the strongest relation between "church" and "historical building"

EXPLORATIVE GAME

UbiquiTO: Which is Peter's interest in abbey? (<inquire move>)
iCITY : The value is 0.4 with belief 0.6 (<inform move>)
UbiquiTO: Which is Peter's interest in cathedral? (<inquire move>)
iCITY : The value is 0.9 with belief 0.7 (<inform move>)
UbiquiTO: Which is Peter's interest in convent? (<inquire move>)
iCITY : I have no value (<deny move>)
UbiquiTO: Which is Peter's interest in historical_building? (<inquire move>)
iCITY : the value is 0.2 with belief 0.9 (<inform move>)
UbiquiTO: Which is Peter's interest in place_to_visit? (<inquire move>)
iCITY : the value is 0.4 with belief 0.5 (<inform move>)

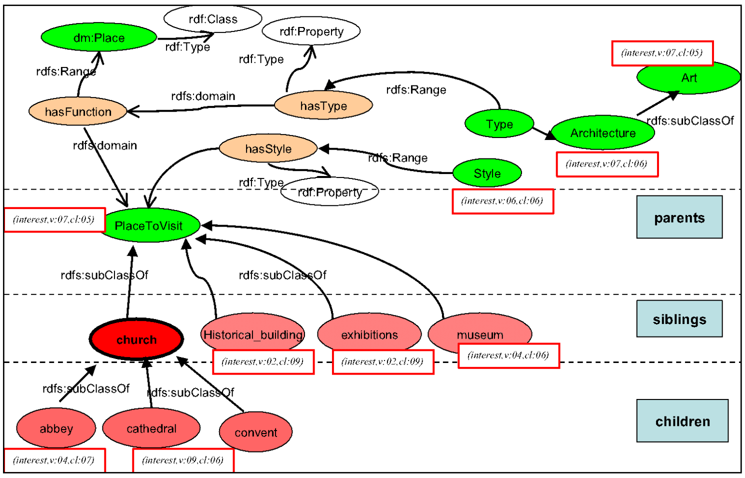

Fig 6 The extracted portion of the domain model relating of "church" after the interaction with iCITY

UbiquiTO has got from iCITY a list of values of Peter's interest in some domain concepts related to the topic, as it could be seen in Figure 6. In this way, even if UbiquiTO does not have the exact value of interest in the concept "church" from iCITY, it can infer it as an average of the values of interest in related concepts, weighed on the confidence level of each value. This is done according to the assumption that the interest in a concept can be assumed to be similar (with a degree of similarity) to the interest in its parent concepts. In this case, we can deduce for Peter a medium interest in church (0.54), since he shows a medium-high interest in the children concepts of the concept.

As regards the post-conditions of the game, UbiquiTO updates its repository with the new values derived from iCITY. For the concepts it has not yet stored any interest value for, UbiquiTO will import the values from iCITY only if they have a high confidence level. For concepts it has already stored an interest value for, it will combine the value derived from iCITY (weighted with a trust value) with the correspondent value stored in its repository (weighted on the confidence level).

Table 6 provides the schema of the explorative game implemented with the specific value of the game.

Parameters	*Goal*	(church)

	Focus	*find_children(church)*
		children= [abbey, cathedral, convent]
		find_parents(church)
		parents= [place_to_visit, place, architecture]
		find_siblings(church)
		siblings= [historical_buildings, museum, exhibition]

Specifications	*Pre-conditions*	*[v_r (interest(church)) = null]*
Components	*Statement*	*s_1= abbey,* *s_2= cathedral,* *s_3= convent,* *s4= historical_buildings,* *s5=place_to_visit*
	List-CA	*< inquire, interest(s1), v?>,* *< inquire, interest(s2), v?>,* *< inquire, interest(s3), v?>,* *< inquire, interest(s4), v?>,*

Table 6. The implemented computational model for the game

Figure 7 presents the sequence of CAs of Dialogue Games performed in the use case by means of a UML sequence diagram.

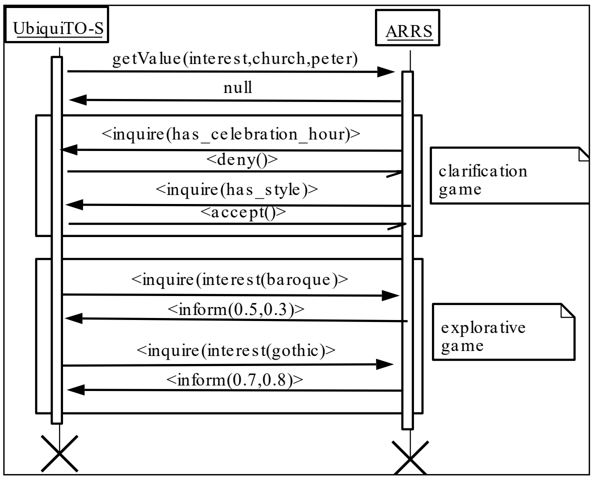

Figure 7. The UML diagram of the dialogues in the use case

As a consequence of the dialogue, UbiquiTO has a wider range of data about the user, that it can use to provide more appropriate results. In fact, before starting the dialogue, UbiquiTO did know that the user did not like visiting historical buildings. Thus, it could happen that it suggested visiting the baroque building of the ancient University Palace. After the dialogue, UbiquiTO knows that Peter: likes cathedral; does not like visiting

historical buildings; likes museums and sightseeing. Therefore, now UbiquiTO is able to correct its results, and it does not recommend visiting an historical building, rather it could recommend a museum about Baroque furniture.
We can conclude that, thanks to the interaction with iCITY , UbiquiTO is able to achieve more reliable knowledge about user. In this sense, the enhanced interaction model supports UbiquiTO in reaching more effective adaptation results. This claim is what we wanted to test: the tests results are described in the following section.

6. Evaluation

In this section we present a preliminary evaluation of the enhanced interaction model using as test bed the interaction between UbiquiTO and iCITY systems, and measuring the results in UbiquiTO.

The starting assumption (derived from literature: Berkovsky 2005, Heckman 2005, Vassileva 2004) is that user model interoperability leads to a qualitative improvement of the user model and consequently of the adaptive behavior of the system. Interoperability of user models improves the coverage and accuracy of the integrated user models (this is quite important since bad input data would result in "misadaptation"), and thus allows for better functions of adaptation: "the more information is available, the more adequate the user model and consequently the adaptation process will be" (Vassileva 2001). According to such a hypothesis, we were expecting that adaptation results would be increased after a user data exchange. In particular, we expected that this increase would be higher if systems interact using our model enhanced interaction model, since it allows to reach UM interoperability also in difficult conditions.

In Section 5, we claimed that, thanks to the interaction with iCITY, UbiquiTO is able to i) achieve more accurate data, and ii) provide the user with better adaptation performance. Now, we want to experimentally support these two aspects separately, to give evidence of the role of our model with respect to adaptation. In fact, even if, as seen, the latter is a consequence of the former, adaptation results can be influenced not only by the data quality but also by the inferences made on such user data, which could be incorrect also in presence of very accurate user model data.
In particular, what we want to evaluate is both the

a) *the accuracy of the UM values*,
b) *the accuracy of adaptation results*

after the exchange of user model data performed by means of

- *a standard interaction model*
- *our enhanced interaction model.*

Hence, we wanted to evaluate if the use of our enhanced interaction model allows to obtain better results.

To measure both the accuracy of UM values and adaptation results, we followed the approaches and the metrics proposed to evaluate an adaptive Web system (Herlocker et al. 2004, Gena and Weibelzahl 2007). In particular, we exploited the *Mean Absolute Error* (MAE) metric. MAE measures the distance between the system's predictions and the user's preferences by means of rate vectors. The smaller the value is, the more accurate the predictions are. Good et al. (1999) suggested that, in the evaluation of a recommender system, a satisfactory value of MAE should be about 0.7 in a range of 0-5.

6.1 Evaluation of the Enhanced Interaction Model with respect to the User Model Data

The **accuracy of the user model** is measured by the difference among the values declared by the user and those stored by the system. Thus, we compare the preferences expressed by the users with the corresponding predictions generated by UbiquiTO and maintained in the user model in three situations:

i) UbiquiTO as stand-alone system.
ii) UbiquiTO and iCITY interoperating with a standard interaction model
iii) UbiquiTO and iCITY interoperating with our enhanced interaction model.

We selected 30 users following a proportional layered sampling strategy[27]: 15 males and 15 females, 20–45 aged, which are quite expert in using Web applications and user-adaptive systems in particular.

We conducted the following experimental setting consisting of 5 main steps:

- *Step 1. User interaction*. We asked the users to interact with iCITY performing a certain number of tasks in order to make the system able to create a user profile of them.
- *Step 2. User assessment*. We presented to the users a printed list containing all the dimensions in the UbiquiTO user model (interest in places to visit, restaurants, etc). We asked them to indicate a value for each dimension on a scale from 0 to 5 (for example, interest in art equals 3 means medium interest)[28].
- *Step 3. Stand-alone system*. We asked the same users to interact with UbiquiTO. We asked them to perform a set of experimental tasks. For example, they were asked to perform the task of "finding some place for dinner and finding something to do in the evening". As many tasks as the number of categories and subcategories were defined. As a consequence of such actions, UbiquiTO is able to create a model of each user.
- *Step 4. UbiquiTO and iCITY interoperating with a standard interaction model.* UbiquiTO starts to exchange user data with iCITY, and provides new results to the users.
- *Step 5. UbiquiTO and iCITY interoperating with our enhanced interaction model.* UbiquiTO exchanges again the user data with iCITY, but using our model, and provides new results to the users.

Table 6 presents the MAE values derived from the comparison of the user's assessment and UbiquiTO's assumptions about them.

	MAE
UbiquiTo alone	0.89
UbiquiTO + iCITY with standard interaction model	0.80
UbiquitTo-S + iCITY with enhanced interaction model	0.67

Table 7. MAE results of the first evaluation

[27] In the proportional layered sampling strategy the population is divided into layers, related to the variables that have to be estimated, and each one containing a number of individuals proportional to its distribution in the target population.

[28] A well-known limitation of the approach of directly asking the user is that people are not used to be completely sincere, for social reasons, in the declaration of personal data (Ardissono et al. 2004). It is quite rare for a user to provide false data to user-adaptive system whose performances are directly related to the data about user herself. People are conscious that bad data results in "misadaptation", i.e., errors in adaptation results (Spooner and Alistair 1997).

With respect to the situation without interoperability, the dialogue leads to a small decrease of the MAE value. This indicates that after a dialogue the user model maintained by UbiquiTO is closer to the real values provided by the users.
The use of the enhanced interaction model leads to lower MAE results.

6.2 Evaluation of the Enhanced Interaction Model with respect to the Adaptation Performance

In the second phase of evaluation, we aimed at measuring the performance of a system that interoperates with our interaction model with respect to the *accuracy of the adaptation results*. Thus, we needed to evaluate UbiquiTO recommendations. We asked the same group of users about their real preferences, and we compared them with the corresponding system recommendations generated the first time the user data were inserted into the system.
In particular, we asked the users to evaluate the first 10 items for each category provided by UbiquiTO.
Then we calculated MAE. The average MAE results were 0.92, with 12 users with a MAE between 0.38 and 0.68, and 10 users with a MAE between 0.45 and 0.78.
Afterwards, we repeated the test (comparing the user assessments with adaptation results) after UbiquiTO has interacted with iCITY by means of a standard interaction model, and our enhanced interaction model.
In Table 8 we can see the results: using the enhanced interaction model the MAE value is smaller, which means better performance.

	MAE
UbiquiTO alone	0.92
UbiquiTO + iCITY with standard interaction model	0.86
UbiquiTO + iCITY with enhanced interaction model	0.71

Table 8. MAE results of the second evaluation

Hence, the value derived from the use of our approach are satisfactory and, in this context, confirm our hypothesis about the validity of our enhanced interaction model to obtain better adaptation performance. However, in order to get more general experimental results, we are planning a new set of experiments with different adaptive systems working in other domains using our model.

We are also working on a *performance evaluation* of our solution, in order to measure qualitative and qualitative attributes (such as modifiability, portability, extensibility and scalability) of the proposed framework. In particular, we aim at verifying how the dialogue model impacts on the communications for exchanging UM data, both in term of efficiency and effectiveness of communication. To evaluate the first aspect, it is possible to measure the number of the exchanged massages in our enhanced interaction model with respect to the use of a standard model. To evaluate the second aspect, it is possible to measure the quality of the exchange (QoE, Chen et al. 2008), i.e., the number of exchanges that succeeded out of the total number of exchanges. According to the features of our model, we expect that the efficiency of communication will be little lower but that the effectiveness will increase.

7. Related work

The problem tackled in this paper is the user model interoperability among web-based user-adaptive applications (Kobsa 2007). This problem is not new and it has been variously addressed in the literature from the beginning of the Nineties. In this paper we proposed a *decentralized* solution (Vassileva 2001) for UM interoperability. Each system maintains a user model for its own purposes of adaptation, and may interact with other systems to collect information about the user at a particular time and with specific purposes, in a peer-to-peer way. Similar decentralized solutions for user model interoperability are the ones of Lorenz (2005), Heckman (2005), Dolog and Schafer (2004). A different approach proposed in the literature is the 'centralized solution' (Kobsa 1990), where the aim is to gather as many data about a user as possible in a central shared space. The knowledge about the user is maintained by a server (called User Modeling Server) which hosts the information. User data are delivered to different applications through a client-server architecture. This solution is widely adopted in the user model community for its advantages (it allows different applications to access the same user knowledge and to make consistent adaptation; it allows mobile applications on devices with limited memory and computing power to exploit a user model). Among the others, we can cite AVANTi (Fink et al. 1996), CUMULATE (Brusilosky et al. 2005), Personis (Kay et al. 2002). Despite the advantages of centralized user modeling, we believe that current and future Web scenarios will require a more flexible and modular architecture. In fact, while the integrated-designed systems can rely on centralized user model servers, the applications working in the Web (and especially in the market) can not, since the centralized UM model is too restrictive. It imposes a set of user features that should be present and a non-negotiable format of representation, APIs, and protocols. Our intention is to exploit the advantages of the distributed approaches, i.e. flexibility in managing privacy[29], efficiency in obtaining only the needed data about the user, modularity of application knowledge that can be spread in different physical devices (Vassileva 2001) .

In this decentralized setting, as seen, our solution for UM interoperability is to exploit the Web Service and Semantic Web standards.

Regarding the use of *Web Service*, the loosely coupled structure of Web Service seems perfect to easily integrate personalization service from various information systems for the sake of cross-system personalization (Niederee 2005). An example of the use of the structure of Web Service to reuse existing user model is provided by (Zhang et al. 06). The user model data are exchanged according to the user schema the service provider registered in registry and the communication between user models from different personalization systems is implemented by binding an XML document to the SOAP protocol. Another example of Web Services to exchange user's (context) features is provided by (Nurmi et al. 2006) in "MobiLife Context Management Framework". The key component of the framework is a generic consumer/producer model given by Context Providers (a software entity that exposes interfaces to provide context information), and Context Consumers. All such examples regard distributed user models as in our solution. However, Web Services technologies can be used not only in decentralized approaches but also in centralized ones, as we can see in (Dolog and Schaefer 2005 and Musa et al. 2006). In Dolog and Schaefer, the user model fragments can be accessed by the use of a Web Service which acts as a learner model server. The server maintains the main model with the data about the learner gathered from different sources, and manages the requests from the clients. Instead, Musa et al. proposed a Web-service based centralized architecture for the integration of learner model among e-learning systems in the Web.

[29] The distributed approach is also more flexible to manage privacy issue than centralized approach, since each system may define which parts of user model to be shared and which ones to keep private (Vassileva 2001, Lorenz 2006).

Each e-learning system which has student's data to share specifies their respective services using WSDL and registers it at the central repository, which acts as a broker system.

Regarding the use of *Semantic Web* languages, there was an increasing interest in the exploitation of semantic Web to make internal knowledge of adaptive systems sharable (Antoniou et al. 2004, Baldoni et al. 2005b, Henze and Kriesell 2004, Dolog et Nejdl 2003). In the direction of interoperability, suppliers and consumers of user model profiles have showed the need for a standard way to represent and exchange user model data. To answer such needs, several models have been introduced : GUMO - General User model Ontology (Heckmann 2005) for uniform interpretation of distributed user models in intelligent semantic Web environments; UUCCM - Unified User Context Model (Niederée et al. 2004) which can be used for modeling characteristics of the user and her situation. Beside such attempts for a general user model, standard for representing user related to particular applications domains has been proposed, especially in learning field. For example, Dolog (2003) introduces a domain and learner ontology for sharing learner model in the learning network.

Another relevant characteristic of our solution is to enrich Web Service interaction by means of a *dialogue model* to face the *syntactic interoperability* on the Web. Our need was to have dynamic interfaces changing in relation to the content of the exchanged message, and the communication protocols defining the order of messages. The problem of an explicit specification of the possible sequences of messages has been addressed in the research field of multi-agent systems. In this area, different dialogue models and languages have been proposed to coordinate the interaction among agents (for example, KQLM -Knowledge Query and Manipulation Language[30], and FIPA[31] ACL[32] - Agent Communication Language). Since we considered Web Services as interacting agents[33], we have applied the same approaches for modeling their conversational behaviour. The use of Speech-Act based approach to specify the conversation flow between Web Services and their consumers in open environments allows for extending the interface in a direction of more flexibility and dynamicity, managing more-than-two-turns interactions, defining communication behaviour as response to messages (Hanson 2004, Ardissono et al. 2003). Hanson propose some enhancements to the existing Web Services architecture by means of the adoption of "conversation-centric" interactions. The same approach is followed by Ardissono et al. The exchanged messages during the service fruition can be seen as a conversational action performed to carry out a task-oriented dialogue between entities with "provider" and "consumer" roles. The model is derived by simplifying Speech-Act theory, while the management of conversation turns is server-side and it is based on the communication techniques proposed by emerging standard for Web Service. Baldoni et al. (2003) proposes an approach to face the problem of the automatic selection and composition of Web Services, where the communicative behavior of the services is modeled as a complex action based on Speech Acts. They proposed to include, in the Web Service declarative description, the high-level communication protocol, used by the service for interacting with its partners.

The last relevant feature of our approach is the use of a *dialogue model* not only for reaching *syntactic* interoperability, but also *semantic* interoperability. This is particularly

[30] http://www.cs.umbc.edu/kqml/

[31] Foundation for Intelligent Physical Agents, http://www.fipa.org/about/index.html

[32] http://www.fipa.org/specs/fipa00061/

[33] In terms of how they interact, differences between agents and rich web service interactions are largely a matter of scale and emphasis. But from a Web Services architectural standpoint, they can be considered as synonymous

useful when systems work in open dynamic environments such as the Web, where cannot be a single universally shared ontology. Thus, some form of ontology mapping is needed. The different techniques and formal methods proposed in the literature for ontology mapping can be manly divided in instance-based and schema-based approaches (see Ehrig and Sure 2005 and Kalfoglou, Y. & Schorlemmer M. 2003 for overview of the different approaches). The negotation protocol we propose in the Clarification Game for concepts disambiguation is based on schema-based matching (Shvaiko and Euzenat 2005) and take inspiration from the graph-based techniques (Maedche and Staab 2002).

Beside the specific theoretical approach exploited, usually the mapping among different ontologies is performed by a centralized service. However, in our approach, the mapping is not performed by a centralized entity, but each application can negotiate for reaching an agreement over not shared concepts by means of dialogue. We chose such dialogue-based solution since it represents an easy tool that is particularly effective, as demonstrated by our evaluation, in our scenario, a small set of systems working on similar domains and exchanging simple data (user model feature and related domain concepts). It enables to perform a basic form of ontology mapping at a run time, and it does not require a preprocess task on ontologies or complex form of reasoning.

Several other approaches in the literature have proposed the use of negotiation through a dialogue between agents in order to enhance standard mapping approaches. Among the others, we can cite Orgun et al. (2005) which propose the resolution of semantic differences at run-time through a semantic ontology negotiation protocol and using the Word Net lexicon[34] to find semantically similar concepts in different ontologies. Another relevant related work is the one proposed by Laera at al. (2006). The agreement is achieved by means of an argumentation process, where there is a set of candidate mapping correspondences among concepts of different ontologies. These correspondences could be accepted or rejected by the agents, according to their knowledge models and their preferences and belief. For example, agents might have different perspectives on the acceptability or not of a candidate mapping correspondence depending on the particular context. Finally, we can mention Bailin and Truszkowski (2002) which present an ontology negotiation protocol which enables agents to exchange parts of their ontology, by a process of successive interpretations, clarifications and explanations.

8. Discussion and conclusions

In this paper we discussed the benefits of a proper integration of the Web standards for interoperability and multi-agents dialogue models with respect to sharing user model knowledge. We presented a scenario where systems provide personalized interaction with the user, which is not aware of the back-end interactions among systems needed to satisfy her request. This approach requires flexible and agile communication between systems which interoperate to exchange user model knowledge. Our aim was to create rich (and scalable) communication protocols and infrastructures to enable consumers and providers of UM data to interact with each other. To this aim we propose an enhanced interaction model based on Web Service and semantic dialogue as an appropriate communication form for UM interoperability in the Web, in order to support to the clarification and negotiation of the data when needed. The framework we propose is based on a Web Service architecture for managing simple queries to user models, and by a dialogue supporting structure, dynamically involved when the request is not clear or the response is missing. The main advantage of our approach is the possibility to allow systems dealing with difficult situations to start the appropriate dialogue game instead of stopping the communication as in the traditional “all-or-nothing” Web Service approach.

[34] http://wordnet.princeton.edu/

We instantiated the framework to exchange domain-dependent user information; in particular we considered applications exchanging the value of the UM feature of Interest (as defined in GUMO) in the domain concepts. However, the same approach could be used to exchange other user domain-dependent features (such as such the user knowledge in a particular topic).

We propose the use of dialogue model to negotiate a mapping of ontologies in our scenario of applications working in similar domains. However, the solution we propose is less suitable with respect to mapping among completely different domains or for very complex and vast domain models. Other modalities of ontology mapping can be included in the framework to face such situation. For instance, an (or a set of) OntologyMapper service can be added in order to offer some specialized ontology mapping abilities (implementing any of the methods proposed in the literature) to the systems in the framework. The systems can publish their mapping requests (in a given shared format) and the specialized OntologyMapper can perform this complex task and provide an answer.

According to the preliminary evaluation discussed in Section 7 we can suggest that the approach is promising in a small context of adaptive systems working in similar domains. It allows to obtain more user data, and thus it improves the adaptation results. However, it is quite difficult to distinguish how good an interaction model is from how good the exchanged values are. We can conclude that the effectiveness for adaptation of the interaction model is very data-dependent. Thus, an interaction model for improving adaptation results should be supported by some mechanism for the evaluation of the reliability of the exchanged values (Carmagnola and Cena 2006b). To evaluate data, systems could use some heuristics that take data properties into account (for example, the last update, the temporal validity, the dependency on the context, or metadata about the reasoning strategies that have lead to the data such as <derivation> – whether the data are directly stated by the users, or inferred by the system, or imported from external systems, or observed by user behaviour). Furthermore, another prerequisite which affects the results is that the applications involved in the dialogue are reliable and cooperative, and provide truth confidence levels. Thus, some mechanisms for the evaluation of the reputation of the provider systems are required (Maximilien and Singh 2002).

Even if promising as demonstrated by our tests, some **open issues** remain unsolved in our solution and need further investigation.

The first one regards the *systems discovery*. One of the main obstacles that makes interoperability a hard issue is that an application has to discover other applications able to interoperate. In order to participate in a conversation, the requestor must know in advanced the system which it interacts with. Thus, a mechanism to support system discovery is necessary. Framework for discovery in the Web which systems store the needed UM knowledge have been proposed by (Cena and Furnari 2008, Chepeing et al. 2005, Specht et al. 2005).

Together with the issue of system discovery, a *user identification task* must be performed among interoperating systems before exchanging the data. This task is necessary for systems to be sure that they are exchanging data regarding the same user, even in absence of a unique user identifier (see Carmagnola and Cena 2009, and Dolog and Schafer 2005 for different solutions for such a problem).

Another relevant issue to be considered regards the management of *user privacy*, i.e., how the propagation of user data among applications can be regulated with respect to the user privacy. User model data have to be exchanged according to some privacy policy (Schreck 2003, Kobsa 2003), in order to fulfill both user preferences and legal requirements. The user can have her personal preferences on privacy dimensions, e.g. about which part of the user model to make available to other applications; which applications can access the user data, the purpose of the data sharing, etc. Legal requirements also affect the possibility of sharing user model data in an interoperable context. They may forbid user model server to supply data to other user-adaptive systems if they use information for different purpose with respect to the starting applications; they can compel service provider to obtain the consensus of the user to transmit the data to third parties. A promising solution to be studied is the P3P standard (Platform for Privacy Preferences Project)[35]. This standard provides the formats by which two parties, client and server, describe and enforce their privacy policy.

Some further **considerations** regard the dialogue management. What we propose in this paper is the use of an existing semantic environment (Sesame) plus some public APIs to manage the dialogue. This solution has the advantage for the applications to be based on a solid widely-adopted environment for the management of the semantic data as Sesame; furthermore, to provide UM data as Web Service interface is simple for existing application, since they can use simple Web Service wrappers to provide WSDL interface. Finally, this solution gives application a very high degree of freedom in managing the conversation. Such freedom implies, however, some implementation efforts on systems in order to implement the dialogue dedicated local modules. To lower the dialogue implementation efforts, we are working in adding to the framework some tools and facilities to facilitate the adoption of this solution.
Several other implementations of our dialogue model are, however, possible. The dialogue could be implemented in any other semantic environment (like Jena[36], Pellet[37]) and also can be encoded in different ways. In fact, we implemented a representation of the dialogue as Java APIs available in the shared dialogue knowledge base. To participate in a dialogue, systems only need to implement the APIs. This choice is a practical simple solution to make the start of dialogue easier, and it is feasible in our case since our dialogue protocol is not too complex. However, the dialogue model could be differently coded, for example making the communication protocols public available in some process-based language as, for instance, abstract BPEL[38] languages. This would allow to represent more complex protocols for communication.

An interesting **evolution** of the work can be to add semantics to all the components of the Communicative Act in order to increase the possibility of understanding, and thus make the dialogue more efficient. In the presented approach, only the "statement" in the Speech Act is semantically represented. A possibility of semantic enrichment could be to add metadata about the dialogue participants. Every dialogue participant could provide in the Communicative Act a meta-description about modalities of its internal knowledge representation, its reasoning strategies, the input necessary to provide output. This information can be used for evaluating the providers and to choose one among them in the discovery phase. Furthermore, it would be interesting to enrich the model with metadata about the move. Each move could be enriched with different set of meta-information according to sub-goals, which can help the disambiguation of the request (e.g., knowing that a user feature is required for the goal of user identification can help in the provision of the response).

Another interesting extension of our work is the sharing not only of the user models data but also the adaptation reasoning (Carmagnola and Cena 2006a). Thus, we are investigating how to include the sharing of adaptation strategies (for example in the form of adaptation rules) in our interaction model. This may be useful for i) evaluating the data, ii) providing a proof of the reasoning of the system[39], iii) evaluating the provider of

[35] http://www.w3.org/TR/2002/

[36] http://jena.sourceforge.net/

[37] http://pellet.owldl.com

[38] http://www.ibm.com/developerworks/library/specification/ws-bpel/

[39] This is relevant for the so called proof layer of the Semantic Web, which involves the actual deductive process as well as the representation of proofs in Web Languages and proof validation

the data. According to the goal, systems can exploit semantic representation of adaptation rules (using languages like RuleML[40], SWRL[41]) or use metadata about the reasoning strategies.

Another possible extension of the work is the intersection with new tendencies in web service field. In the last years, there has been an increasing interest in web service composition and related topic of choreography and orchestration of web service. Choreography in particular define a multi-parts conversation from a global perspective. This can be useful to enhance our dialogue model, which supports only bilateral interactions, since the definition of Communication Act in the model does not include more than two actors. Our model also does not support the synchronization of different dialogue games, due to the fact that while a system is playing a dialogue game with another system, it is not aware of other dialogue games occurring at the same time. Instead, in order to make the most appropriate decision of the next step to perform (for instance, to decide to stop the current dialogue and to start another one) it would be useful to be aware of the whole conversational context. To this aim, we are investigating the integration in the model of some emerging XML-based standards to model Web Service interaction as a conversation. Candidate languages are WSCL[42] (Web Service Conversation Language) and WCSI[43] (Web Service Choreography Interface). Both of them introduce an explicit representation of Web Service interaction process, defining the admissible sequence of messages in more elaborated protocols for communication in order to compose different services offered by different applications. Differently, our solution has the goal of improving the interaction with a single web service for the provision of a single service (the UM data exchange).

Acknowledgments

I would like to thank Luca Console and Lora Aroyo for their support, suggestions and fruitful discussions. A particular thank also to Cristina Gena for her help in the evaluation of the approach.

References

Antoniou, G., & van Harmelen, F (2004): A Semantic Web Primer. MIT Press

Antoniou, G., Baldoni, M. Baroglio, C., Baungartner, R., Bry, F., Eiter, T., Henze, N., Herzog, M., May, W., Patti, V., Schaffert, S., Schidlauer,S., & Tompits. H. (2004). Reasoning methods for personalization on the semantic web. Annal Math. Comput.Teleinformatics (AMCT) 2 (1), pp. 1-24.

Ardissono, L., Petrone., G., & Segnan, M. (2004). A conversational approach to the interaction with Web Services. In Computational Intelligence, Volume 20, Number 4

Aroyo, L., Dolog, P., Houben, G-J., Kravcik, M., Naeve, A., Nilsson, M. & Wild, F. (2006). Interoperability in Personalized Adaptive Learning. Educational Technology & Society, 9 (2), pp. 4-18

[40] http://www.ruleML.org

[41] http://www.w3.org/Submission/SWRL/

[42] WSCL focuses on modeling the sequencing of the interactions, exploiting a sequence diagram model that the service provider and the consumer should interpret to handle conversatio. http://www.w3.org/TR/wscl10/

[43] WSCI introduces the notion of interaction process, with the definition of timing constraints on the service invocation. However, it cannot support a fine-grained specification of the conversation turns. http://www.w3.org/TR/wsci

Aroyo, L., Denaux, R., Dimitrova, V., & Pye, M. (2005) Interactive Ontology-Based User Knowledge Acquisition: A Case Study. In Proceedings of UM2005 workshop

Balabanovic, M. and Y. Shoham, (1997) “Fab: Content-Based, Collaborative Recommendation,” Comm. ACM, vol. 40, no. 3, pp. 66-72

Bailin, S.C & Truszkowski, W. (2002) Ontology Negotiation: How Agents Can Really Get to Know Each Other. In Proceedings of the WRAC 02

Baldoni, M., Baroglio, C., Martelli, A., & Patti, V. (2004). Reasoning about interaction protocols for Web Service composition, in: Bravetti and Zavattaro pp. 21–36, Electronic Notes in Theoretical Computer Science, vol. 105.

Baldoni, M., Baroglio, C., &. Henze, N. (2005) Personalization for the Semantic Web. In Reasoning Web, First International Summer School, Springer, Berlin

Baker, M.(1994) A model for negotiation in teaching-learning dialogues. Journal of Artificial Intelligence in Education, 5 (2), pp. 199-254

Berkovsky, S. (2005) Ubiquitous User modeling in Recommender Systems. International Conference on User modeling. Edinburgh, Scotland

Berners-Lee, T., Hendler, J., & Lassila, O.(2001). The Semantic Web. In Scientific American, May 2001, pp. 28-37

Brusilovsky, P. (1996) Methods and techniques of adaptive hypermedia. In User modeling and User Adapted Interaction, v 6, n 2-3, pp. 87-129

Brusilovsky, P., Sosnovsky, S., & Yudelson, M. (2005). Ontology-based framework for User model interoperability in distributed learning environments. In: G. Richards (ed.) in Proceedings of World Conference on E-Learning, E-Learn 2005, Vancouver, Canada, AACE, pp. 2851-2855

Carmagnola, F., & Cena, F. (2006a). From Interoperable User model to Interoperable User modeling. In LNCS vol. 4018, ISSN: 0302-9743. ISBN: 3-540-34696-1, pp 409-413 “AH 2006, the 3rd Adaptive Hypermedia and Adaptive Web based Systems Conference” Dublin, Ireland, Jun. 20-23, 2006

Carmagnola, F., & Cena, F. (2006b). An approach for evaluating User model Data in an interoperability scenario. In Proceeding of “STAIRS 2006, the third European Starting AI Researcher Symposium”, IOS PRESS vol. 142, ISSN: 0922-6389. ISBN: 1-58603-645-9, 251-252, Riva del Garda (TN), Italy

Carmagnola, F., Cena, F., Console, C., Gena, C., Goy, A., Torre, I., Toso, A. & Vernero, F. (2008). ‘Tag-based User Models for Social Multi-Device Adaptive Guides’ in UMUAI User Modeling and User-Adapted Interaction, Special Issue on Personalizing Cutural Heritage Exploration, Vol. 18, Issue 3, May 2008, pp. 478-506

Carmagnola, F. & Cena, F. (2009). User Identification for Cross-System Personalization. Information Sciences, Volume 179, Issues 1-2, 2 January 2009, pp. 16-32

Cena, F., & Aroyo, L.M. (2007). A Semantics-based Dialogue for Interoperability of user-Adaptive Systems in a Ubiquitous Environment. In Proceedings of the Eleventh International Conference of User Modeling UM2007" , 25-29 June, Corfu, Greece, pp. 309-313

Cena, F. & Furnari, R. (2008). Discovering and Exchanging Information About Users in a SOA Environment, Communications of SIWN (Systemics and Informatics World Net) (formerly System and Information Sciences Notes) , ISSN 1757-4439 Volume 4, June 2008, pp 34-38

Cena, F., & Torre, I. (2006). Adapting the Interaction in a Call Centre. In Interacting with Computers, ISSN 0953-5438, ELSEVIER, Volume 18, Issue 3, pp 478-506, May 2006

Cena, F., Console, L., Gena, C., Goy, A., Levi, G., Modeo, S., & Torre, I (2006). Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. In AICommunication, The European Journal on Artificial Intelligence, IOS Press , 19(4) 301-400, pp. 369-384

Cena, F. (2007). The Role of Semantic Dialogue for Adaptation for Ubiquitous Environment. Ph.D dissertation. Department of Computer Science, University of Turin

Chen, D., B. Vallespir, and N. Daclin: 2008, ‘An Approach for Enterprise Interoperability Measurement’. In Proceeding of: MoDISE-EUS. pp. 1–12.

Chepegin, V. (2005). Broker-based Discovery Service for User models. In Proceedings of PerSWeb'05 workshop at User Modeling 2005 July 24 - 30, 2005, Edinburgh, UK

Cohen, P. (1985). Heuristic Reasoning about Uncertainly: An artificial Intelligence Approach. In Morgan Kaufmann Publishers, Inc., Los Altos, California

Chrobot, S. (2002). Modelling Communication in Distributed Systems. In: PARELEC.

De Bra, P., Aroyo, L., & Chepegin, V. (2004): The Next Big Thing: Adaptive Web-Based Systems. In Journal of Digital Information, Vol. 5 Issue 1 Article N. 247, pp. 55-60

Dimitrova, V. (2001). Interactive Open Learner Modelling, Ph.D. dissertation, Leeds University

Dolog, P. & Nejdl, W. (2003). “Challenges and Benefits of the Semantic Web for User Modeling”, in Proceedings of AH2003 workshop

Dolog, P. & Schäfer, M. (2005). A Framework for Browsing, Manipulating and Maintaining Interoperable Learner Profiles. User modeling 2005. pp. 397-401

Dolog, P., Henze, N., Nejdl, W., & Sintek, M.(2004). Personalization in Distributed e-Learning Environments. International World Wide Web Conference, Education Track, New York

Ehrig, M. and Y. Sure (2005). Ontology mapping - an integrated approach. In Proceedings of the European Semantic Web symposium (ESWS), pp. 76–91

Fink, J., Kobsa, A., & Nill, A. (1996). User-Oriented Adaptivity and Adaptability in the AVANTI Project. In Proceedings of the Conference “Designing for the Web: Empirical Studies”, Redmond, WA

Fukuda, M., Bic, L., Dillencourt, M. & Merchant, F. (1997). ‘Messages versus Messengers in Distributed Programming.’. In Proceedings of the 17th International Conference on Distributed Computing Systems - ICDCS’97. pp. 347–354

Gena, C. & S. Weibelzahl (2007). Usability Engineering for the Adaptive Web. In Brusilovsky et al., 2007, pp. 720–762.

Good, N., Schafer, B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., & Riedl J. (1999). Combining collaborative filtering with personal agents for better recommendations. In AAAI/IAAI, pp. 439–446

Hanson, J., Nandi, P., & Levine, D. (2003). Conversation-enabled Web Service for agents and e-business. In CAiSe, Klangenfurt, Springer-Verlag

Heckmann, D. (2005) : Ubiquitous User modeling. PhD thesis, Department of Computer Science, Saarland University, Germany

Henze, N., & Kriesell, M. (2004). Personalization functionality for the Semantic Web: Architectural Outline and First Sample Implementations. In First International Workshop on Engineering the Adaptive Web (EAW 2004), co-located with International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2004)

Herlocker, J. L., Konstan, J. A., Terveen, L. G. & Riedl, J. (2004). Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22(1), pp. 5–53.

Kay, J., Kummerfeld R. J., & Lauder, P. (2002). Personis: a server for user models. In De Bra, P, P Brusilovsky, R Conejo (eds). In Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems AH'2002, Springer, pp. 203 -212.

Kalfoglou, Y. & Schorlemmer M.(2003) Ontology mapping: the state of the art The Knowledge Engineering Review Volume 18 , Issue 1, pp. 1 - 31

Kobsa, A. (2007). Generic user modeling systems. The Adaptive Web: Methods and Strategies of Web Personalization, pp. 136-154

Kobsa, A., Koenemann, J., & Pohl, W. (2001). Personalized hypermedia presentation techniques for improving online customer relationship, The Knowledge Engineering Review

Laera, L., Tamma, V., Euzenat, J., Bench-Capon T., & Payne, T. (2006). Reaching Agreement over Ontology Alignments. I. Cruz et al. (Eds.): ISWC 2006, LNCS 4273, Springer-Verlag Berlin Heidelberg, pp. 371–384

Levin, J., & Moore, J. (1977). Dialogue games: meta-communication structures for natural language interaction. In Cognitive Science, 1 (4), pp. 395-420

Levinson, S. (2001): Pragmatics. In N.J. Smelser & P. Baltes (eds.), International Encyclopedia of Social and Behavioral Sciences. Vol. 17. Amsterdam/Oxford: Elsevier Science, pp. 11948-11954

Lorenz, A. (2005). Agent-Based Ubiquitous User modeling. in Proceedings of User Modeling 2005, pp. 512–514

Maedche, A. and S. Staab (2002), Measuring Similarity between Ontologies. In Proceeding of European Conference on Knowledge Acquisition and Management. Madrid, Spain.

Maximilien, E.M. & Singh, M.P. (2002). Conceptual Model of Web Service Reputation., ACM. *SIGMOD Record* 31(4), pp 36-41

McBurney, P., & Parsons, S. (2004). Games that agents play: A formal framework for dialogues between autonomous agents. In Journal of Logic, Language and Information, Volume 11, Number 3, Springer, pp. 315-334

McGinnis, J.P., Robertson, D., & Walton, C. (2003): Using Distributed Protocols as an Implementation of Dialogue Games. In EUMAS, the first European workshop on multi-agent systems

Musa, D., & Palazzo Moreira de Oliveira, J. (2004). Sharing Learner Information through a Web Services based Learning Architecture, in Proceedings of WISM Workshop co-located with the CAiSE Conference, Riga, Latvia

Montaner, M. & Lopez-B. de la Rosa J.L. (2003). A taxonomy of recommender agents on the internet. Artificial Intelligent Review, 19(4), pp. 285–330

Niederée, C., Stewart, A., & Mehta, B. (2004). A Multi-Dimensional., Unified User model for Cross-System Personalization. AVI'04, May 25, Gallipoli, Italy

Nurmi, P., Salden, A., Lun Lau, S., Suomela, J., Sutterer, M., Millerat, J., Martin, M., Lagerspetz, E., & Poortinga, R. (2006). A System for Context-Dependent User modeling in Proceedings of Workshop on Context Aware Mobile Systems (CAMS), pp. 1894-1903

Orgun, B., Dras, M., Cassidy, S. & Nayak, A. (2005). DASMAS - Dialogue based Automation of Semantic interoperability in Multi Agent Systems. In Procedings Australasian Ontology Workshop (AOW 2005), Sydney, Australia. CRPIT, 58. Meyer, T. and Orgun, M. A., Eds., ACS. pp. 75-82

Rich, E. (1983). Users are individuals: Individualizing user models. International Journal of Man-Machine Studies, 18(3):199-214

Salton, G., & McGill, M. (1984). Introduction to Modern Information Retrieval. McGraw-Hill, New York

Searle, J. (1972). What is a Speech Act. In Giglioli P. (ed.), Language and Social Context, Penguin Books Ltd., England

Shvaiko, P. & J. Euzenat (2005). A Survey of Schema-based Matching Approaches. Journal on Data Semantics (JoDS) IV, pp. 146–171

Sowa J.(1984) . Conceptual Structures: Information Processing in Mind and Machine. In Addison-Wesley, MA

Specht, M., Lorenz, A., & Zimmermann, A.(2005). Towards a Framework for Distributed User modelling for Ubiquitous Computing. In Proceedings of workshop on Decentralized, Agent Based and Social Approaches to User modelling (DASUM 2005) Edinburgh, UK July 25

Spooner, R. & Alistair, E. (1997). User modelling for Error Recovery: A Spelling Checker for Dyslexic Users. Vienna, New York, pp. 147–157

Stojanovic, N. (2005). On the role of a user's knowledge gap in an information retrieval process. In Proceeding of k-CAP'05, ACMPress

Sycara, K.P.(1998): Multiagent Systems. AI Magazine 19(2), pp.79-92.

Vassileva, J.(2001) Distributed User modelling for Universal Information Access. In Stephanidis C. (ed.), Universal Access in Human - Computer Interaction, Lawrence Erlbaum: Mahwah, N.J., pp. 122-126

Veltman, K. H. (2001). Syntactic and Semantic Interoperability: New Approaches to Knowledge and the Semantic Web. The New Review of Information Networking 7

Wooldridge, M. (1966): An introduction to Intelligent Agents. (Reprinted in August 2002 by John Wiley & Sons, LTD, England)

Zhang, F., Z. Song, & H. Zhang (2006). Web Service Based Architecture and Ontology Based User Model for Cross-System Personalization. In Proceeding of WI '06: the 2006 IEEE/WIC/ACM International Conference on Web Intelligence. Washington, DC, USA, pp. 849–852.

List of Figures

List of Tables