Advances in Internet of Things, **2012, 2, 91-105**
http://dx.doi.org/10.4236/ait.2012.24012 Published Online October 2012 (http://www.SciRP.org/journal/ait)

UMIS: A Service for User Model Interoperability

Federica Cena, Roberto Furnari
Department of Computer Science, University of Turin, Turin, Italy
Email: cena.furnari@di.unito.it

Received July 13, 2012; revised August 27, 2012; accepted September 10, 2012

ABSTRACT

In this paper we describe UMIS, a service architecture that enables user adaptive applications to exchange User Model data on the Web. UMIS provides a set of facilities that allow applications to interoperate with minimum changes in their internal logics and knowledge representation. The goal is to support the process of interoperability in three ways: providing an efficient centralized discovery service; offering a service for simple interaction for the exchange of UM value in a p2p way; and offering a negotiation mechanism to be used in case of communication hurdles (*i.e.* semantic ambiguities and missing response). We developed a proof-of-concept prototype of UMIS and we tested it with an existing user-adaptive application. According to our test results, our approach improves the communication with respect to standard solutions for interoperability regarding the quality of exchange, with a negligible impact on the communication costs and traffic generation.

Keywords: User Modeling; Interoperability; Dialogue Game; Semantic Web; Web Service

1. Introduction

In recent years, with the diffusion of Social Web applications people provide a lot of personal information to applications. That is particularly useful for *adaptive applications* [1], that need to collect data about users to provide adaptive services to them. In an adaptive system, the information on the users is explicitly stored in a specific data structure, the User Model (UM), which maintains *domain-independent* (e.g., age, gender, etc.) and *domain-dependent* user features, represented by values assigned to some domain concepts, indicating the user's level of preference for these concepts [2,3].

In this context, sharing information about users among applications (*UM interoperability* [4]) can be a practical way to have more complete and accurate user models. This enables systems to offer more accurate adapted services to the users, since the quality of adaptation mainly depends on the quantity and quality of user data in the user model [5,6]. Moreover this makes systems able not to bore the users asking them for the data which are already available in other systems. UM interoperability proved especially useful at the early stage of a user-system interaction to solve the so called *cold start problem* [7].

Despite these advantages, UM interoperability is not very often exploited by real systems. In fact, UM interoperability in an open and dynamic environment like the Web is a complex task and requires a very high level of alignment by applications, both on communication protocols and on syntax and semantics of data [8].

We aim at lowering some of the common barriers to interoperability, proposing a web service infrastructure for User Model Interoperability (UMIS), that adaptive web applications can exploit to exchange data about users. UMIS offers a set of facilities making applications able to interoperate in a flexible and dynamic way, limiting the changes in their internal logic and knowledge representation.

The process of UM data interoperability can be logically divided in three phases:

- Discovery of the system which stores the desired UM data;
- Exchange of the UM data;
- Interpretation of the data.

The main contribution of the work is to support the process of interoperability in such three phases:

- It *provides* an efficient centralized *service discovery tool*;
- It offers an *effective* simple basic *model of interaction* for the exchange of UM value in a peer to peer way;
- It offers a negotiation mechanism to support *interpretation* in case of semantic ambiguities or missing (or non satisfactory) responses.

UMIS can be exploited to exchange both domain-independent and domain-dependent information. In this paper we focus on the latter, considering the use case of applications exchanging the value of the *user interest* in some domain concepts. In order to evaluate our approach, we made iCITY [9], an existing adaptive application, interoperate exploiting UMIS services. Then, we evalu-

ated: 1) the performance of the exchange, with respect to the network traffic and the communication cost; 2) the quality of the exchange.

In this work, we implement the conversation model of [10], revised and modified in order to fit implementation requirements. With respect to such previous work we:

- Revised the original communication model and provided a new implementation of it;
- Provided a service architecture to support the model;
- Added an event-based information discovery mechanism.

The basic ideas of UMIS has been proposed in a preliminary version in [11].

The paper is structured as follows. Section 2 describes the interoperability problems and our approach to address them, while Section 3 focuses on the dialogue model. Section 4 describes the framework components and 5 presents the framework in action. Then Section 6 describes the evaluation of our solution. Section 7 analyzes some of the most relevant related work. Finally, Section 8 concludes the paper with some comments and remarks.

2. Interoperability Issues

As seen above, the UM interoperability process follows three steps: 1) discovery; 2) interaction; 3) interpretation. In our work we aim at solving some interoperability problems that can occur in each phases:

Discovery issue. Applications looking for user features should easily and quickly discover which applications offer the desired information and how they can be contacted. Thus, we aim at providing a sort of yellow page mechanism focused on the user model context.

Communication issue. The interaction between two applications can take place only if they agree on the communication protocols and on the syntactic structure of the exchanged information. Our purpose is to provide a framework which makes applications able to interoperate in a flexible and dynamic way, avoiding the necessity to a direct peer to peer preliminary agreement on the communication protocols.

Semantic interpretation issue. When the *knowledge model is not shared*, it is needed to ensure a correct understanding of the exchanged data; we aim at ensuring an interpretation mechanism to solve semantic ambiguities. In case of *missing* or *unreliable response*, our purpose is to provide a value derivation mechanism.

In order to address these problems, we propose the following solutions:

1) To enhance the *discovery issue*, we chose to enhanced a traditional UDDI (Universal Description Discovery and Integration) registry with a shared network space based on the publish/subscribe pattern.

2) To solve the *communication issue* we exploited Service Oriented Computing [12], since the loosely coupled structure and the well accepted stack of standards underlying Web Services technology represent an effective solution for communication on the Web.

3) To cope with the *semantic interpretation issue* we use resources from the Semantic Web (ontologies and specification languages) [13]. To preserve the autonomy of the applications, we do not impose to use a unique ontology[1], but we only require to refer to ontologies expressed with Semantic Web languages. In case of semantic ambiguities on the exchanged concepts, we chose to negotiate the meaning of the concepts by means of a *conversation*, adopting a revised version of a *Dialogue Game* model specifically conceived for UM interoperability [10] (Section 3). We also used the dialogue model to deal with the problem of *missing or unreliable response* in order to derive a response when the exact one is not available or is not satisfactory.

3. Conversation Model

As mentioned, UMIS extends the conversation model presented in [10] in order to solve the problems of semantic interpretation and of missing-unreliable responses. Such model adapts to UM interoperability context the diagnostic learning dialog model of [15], based on Dialogue Games [16] and Speech Acts [17] theories. The model is based on *Dialogue Game* (DG), a template describing the communication behavior the systems should follow to reach a particular goal. In the UM interoperability context, a dialogue involves a system looking for user data (*Requestor*), and a system providing this data (*Provider*). The basic dialogue primitive of a DG is the *Speech Act*, represented as a couple "*move*, *statement*" where:

- A *move* is a domain-independent verb expressing the system intention, such as to inquire for something, to inform about something, etc.
- A *statement* about the User Model is represented as a quad <*f*, *c*, *v*, *r*>, where:
  - *f is* a domain-dependent user *feature* in the UM;
  - *c* is the concept of the Domain Model the user feature refers to;
  - *v* is the *value* of the UM feature (a value from 0 to 1);
  - *r* is the *reliability* of the value (a value from 0 to 1).

An example of Speech Act is *inform*,<*Interest*, *Art*, 0.5, 0.2> which informs that the value of the UM feature *Interest* in the Domain Model concept *Art* is medium (0.5) with a low reliability (0.2).

More in details, the dialogue game components are:

- *Specifications*, the *pre-conditions* of the game, *i.e*. the specific situation triggering a dialogue game;
- *Parameters*, the *goals* of the game;

[1]To impose a common knowledge model is not always possible, since it is not easy to find a unique complete knowledge model with all the features of the different domains [14].

- *Components*, strategies determining the Communicative Acts to be generated during the game:
  - *Content tactic*: to gather the concepts to be discussed in the game (*focus space*).
  - *Scope strategies*: to extract from the focus space the concepts to use as statement of the CA.
  - *Dialogue rules*: to determine the sequence of Communicative Acts
  - *Post-conditions*: to determine the changes at the end of the dialogue.

We revise the original model to favor its implementation by applications.

First, we did not consider the *preconditions* and the *postcondition* of the game, leaving each system free to decide when to start and end a dialogue instance.

We also ruled out the *scope strategies*, relaxing the constraints in the order of concepts exchanged in a dialogue, leaving more autonomy to the systems.

Then, we selected only a subset from the original moves in [10] (*i.e.*, to inquire, *to agree*, *to disagree*, *to request*, *to inform*, *to deny*) sufficient for the goal of UM interoperability.

Finally, we revised the dialogue rules modeling the flow of Communicative Act to be exchanged during a game in terms of Conversation Protocol.

Moreover, we enriched the model introducing an explicit measure of similarity between concepts. In particular we consider a measure of Similarity derived from properties [20,21][2]. Similarity between objects can be expressed as a function of their common and distinctive features (properties). We derived this idea from the *Similarity Theorem* [18], that asserts that the similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B, and the information needed to fully describe what A and B are. Thus we define a similarity measure *sim* between two concepts A and B as:

$$sim(A,B) = \frac{comm(A,B)}{\text{description}(A,B)} \quad (1)$$

where *comm*(*A*, *B*) is the number of the properties shared by *A* and *B*, and description(*A*, *B*) is the total amount of properties of *A* and *B*.

As a result, in UMIS a Dialogue Game is defined by the combination of *Content Tactics* and a *Conversation Protocol*.

3.1. Content Tactics

UMIS implements the following tactics:

- *tc*1: *find.parents* (*c*), to gather the concepts directly related to the concept *c*, *i.e*. in an ontology, the direct superclasses. In **Figure 1**, *RedWine* for *c* = *Barolo*;
- *tc*2: *find.children* (*c*), to collect the descendants of *c*, *i.e*. in an ontology, the subclasses. In **Figure 1**, BaroloChinato and BaroloRiserva for Barolo.
- *tc*4: *find.property* (*c*), to collect all the properties of *c*; in **Figure 1**, *has_taste* and *has_smell* for *RedWine*.
- *tc*3: *find.siblings* (*c*), to gather the closest siblings concepts, *i.e* in an ontology, the concepts at the same level with the same parents and with a certain number of same properties. In the figure, among the children of RedWine, Barolo and Nebbiolo can be considered as the most similar since they share the property is_still.
- *tc*5: *find.ancestors* (*c*), to gather all the ancestors concepts of *c*, *i.e*. in an ontology, all superclasses upper than parents. In **Figure 1**, Wine and Beverage for Barolo.

3.2. Conversation Protocols

A conversation protocol is the set of ordered messages the two roles (Requestor and Provider) have to exchange to correctly implement the dialogue. In UMIS, we defined two main basic communication blocks, using the chosen subsets of moves:

***Request-pattern*:** a Requestor may ask to receive the value *v* (and its reliability *r*) of the UM feature *f* for the domain concept *c*: (*request* (*f*, *c*)) (e.g. *request* (*Interest*, *Barolo*)). As answer, a Provider can: 1) *inform* (*f*, *c*, *v*, *r*), to provide the value and its reliability, e.g. *inform* (*Interest*, *Barolo*, 0.5, 0.2); 2) *deny* (), to refuse to provide an answer.

***Inquiry-pattern*:** a Requestor may simply want to know if the model has the concept *c*: *inquire* (*c*) (e.g. *inquire* (*Barolo*)) or the property *p*: (*inquire* (*p*, *c*) (e.g. *inquire* (*is Sparkling*, *Barolo*)).

The Provider can then: 1) *agree* () (when it has *c* or *p*); 2) *disagree* () (when it does not have *c* or *p*).

In UMIS a communication protocol is represented by means of a UML Activity Diagram where each activity node is a *Communication Activity* representing the fact that an application *sends* (or *receives*) a message to (or from) another application. For instance, the activity (*send* (*inform*(*s*))) in the conversation protocol associated to the role of the Provider means that the application playing the role of Provider has to send the message *inform* with the statement *s* to the Requestor. Note that in the protocol no internal business logic and activities, as well as internal data flow, are represented. **Figures 2** and **3** represent the conversation protocols implemented in UMIS.

3.3. The Dialogue Games in UMIS

UMIS implements the two *Dialogue Games* presented in

[2]With *property* we refer to rdf:Property as defined in the specification: "RDF properties may be consider as *attribute* of resources and in this sense correspond to traditional attribute-value pairs. RDF properties also represent *relations* between resources." (http://www.w3. org/TR/ rdf-schema/).

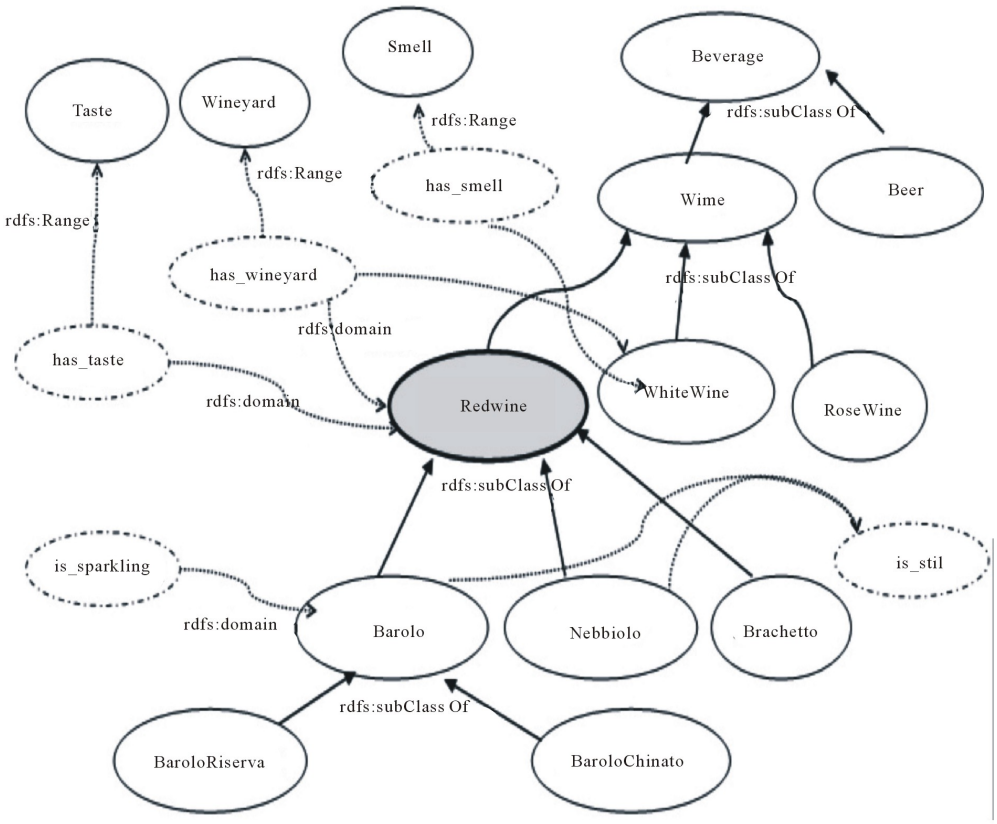

Figure 1. A portion of the domain ontology Food.

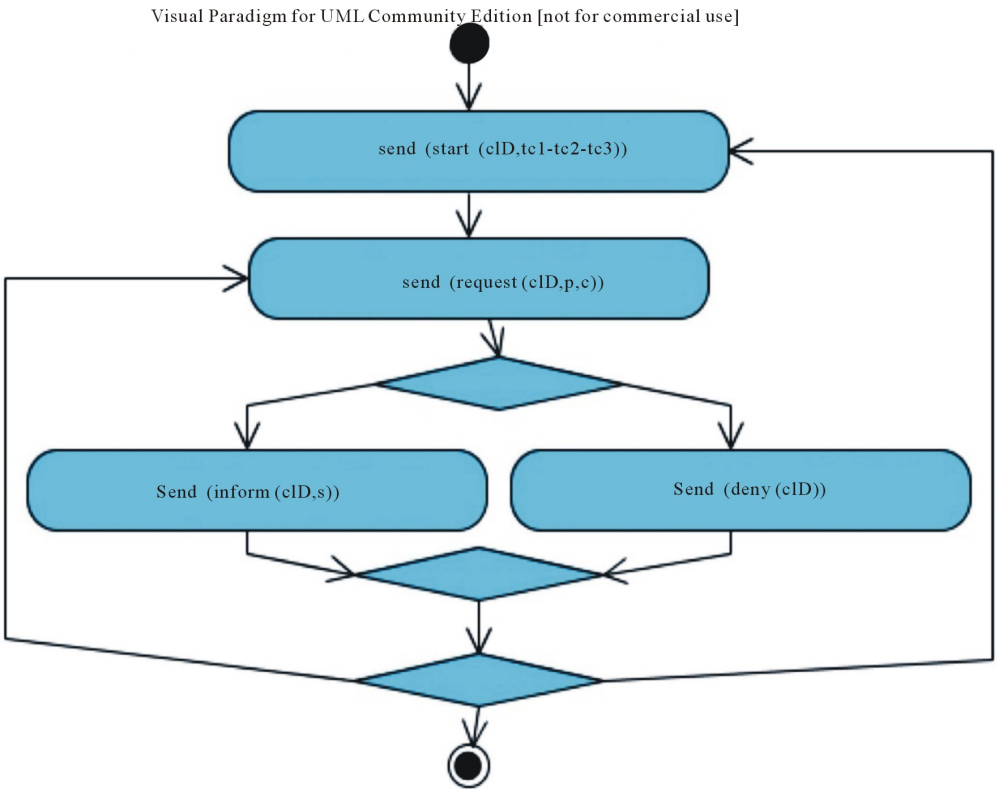

Figure 2. Conversation protocol for the Explorative Game from the Requestor point of view.

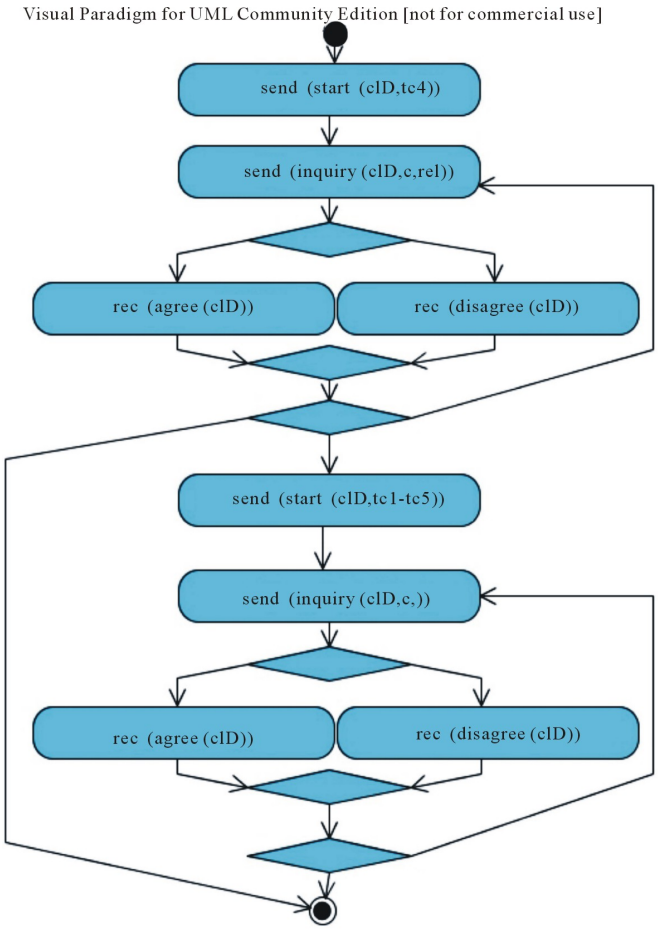

Figure 3. Conversation protocol for the clarification game from the provider point of view.

[10] according to the revised model:

Explorative Game, used for the derivation of the values for a concept of a shared ontology, when the Provider does not have the requested value or the reliability value is too low;

Clarification Game, used for the disambiguation of the domain concept *c* in the request, when *c* does not belong to a shared ontology.

In the following we describe in more details the two games, highlighting the difference with respect to the original model. An example of game execution in the framework is provided in Section 5.

3.4. Explorative Game

The goal of this dialogue game is to derive a value of a user feature for a domain concept starting from the values of such a feature for the *related* concepts. The rationale behind the game is that the attitude of a user on a concept can be assumed to be similar to the attitude on its related concepts (according to taxonomy approach [18]). For instance, if the user expresses an interest in a concept, this interest can be extended with high reliability to its upper and lower class concept, and to sibling concepts.

Content Tactics. The focus of this game is constituted by the concepts related to the starting one (parents, children, sibling concepts). The tactics *tc*1 (*find.parents*), *tc*2 (*find.children*), *tc*3 (*find.siblings*) can be used for this purpose.

Conversation Protocol. The conversation protocol of the game is based on the request-pattern and it is described in **Figure 2**. In the figure we give a compact representation of the protocol using the convention that *tc*1-*tc*2-*tc*3 indicates that the same block of actions can be repeated in sequence using tactics *tc*1, *tc*2, *tc*3.

A starting message *start* (*cID*, *tc*1) is sent by the Requestor to inform that it is starting to send messages applying the tactic *tc*1, and then it starts to send a message *request* (*cID*, *c*), where *request* is the move, and *c* is one of the concepts derived from the application of the *tc*1.

The Provider can *deny* (*cID*) the request or produce an *inform* (*cID*, *s*) message with the requested statement. In both cases, the Requestor can go on with the conversation or terminate it, according to its own internal strategy. If it decides to go on, it can produce a second execution of the block sending statements derived by the application of *tc*2

tactic. In this case the Requestor asks for the children concepts. The same can then be done using the tactics *tc*3. Notice that in the protocol we do not represent any condition associated to the *Choice* construct: the decision depends on the policies of each system.

Policies for the value derivation. Generally, IS-A relations do not represent a uniform similarity distance among concepts [19]. Thus, in the value derivation we consider a measure of similarity among classes connected with IS-A relations according to Formula (1). Thus we compute the derived value as an average of the values of related concepts, weighed on their similarity degree (Formula (2))

$$val(c) = \frac{\sum_{i=1\cdots n} val(c_i) * sim(c, c_i)}{\sum_{i=1\cdots n} sim(c, c_i)} \qquad (2)$$

where c_i are the n concepts related with c and $sim(c, c_i)$ is the similarity degree computed between c and c_i.

Reliability. The derivation of the value has a negative impact on its *reliability*; e.g., if the value of user *Interest* in *Barolo* is derived from the *Interest* in its parent *RedWine*, its reliability is lower than a value directly associated to the *Barolo*. In UMIS, reliability (r) of the derived value ($val(c)$) depends on the similarity of the involved concepts (Formula (3))

$$r_{val(c)} = \frac{\sum_{i=1,\cdots} r_i * sim(c, c_1)}{\sum_{i=1\cdots n} sim(c, c_1)} \qquad (3)$$

3.5. Clarification Game

The goal of the game is to disambiguate two concepts belonging to different ontologies, *i.e.* to discover if the two concepts are the same concept. The rationale of the game is that two concepts can be considered as the same if their similarity is very high. To measure similarity, we consider both the similarity derived from properties and from IS-A relations.

Content Tactics. The focus of the game is constituted by 1) the properties of the concept; 2) the parents of the concept. For example, the concept *Barolo* can refer to a typology of wine (ontology *Food*, **Figure 1**) or to an Italian city (ontology *GeoLocal*, **Figure 4**). To disambiguate it, a possible focus could be composed by:

- Properties (e.g *is still*, *has_taste*), according to *tc*4 (*find.properties*) tactic;
- Superclasses (e.g. *RedWine*, *Wine*, *Cuneo*, *Piedmont*) following *tc1* (*find.parents*) and *tc*5 (*find.ancestors*) tactics.

Conversation Protocol. The conversation protocol is based in the inquiry-pattern (**Figure 3**). The initiative of starting the game is undertaken by the Provider when it needs to disambiguate a request. The schema of the conversation is similar to the Explorative Game (**Figure 2**).

Policies for concept disambiguation. In UMIS, we consider two concepts as the same concept when a certain threshold is passed ($t = 0.55$). We use the following formula (Formula (4)):

$$sim'(c_1, c_2) + \sum_i 0.2 * sameSuper_i > t \qquad (4)$$

where $sim'(c_1, c_2)$ represents an approximation of the similarity between the two concepts c_1 and c_2 computed only on the exchanged properties (a minimal set of properties is considered). An additive factor of 0.2 is added for each superclass shared between the two concepts ($sameSuper_i = 1$ when the superclass is shared, 0 otherwise).

Reliability. With respect to the reliability, in the framework when the computed threshold is higher than 0.8, the reliability of the concept is considered not affected by the uncertainty deriving from the disambiguation procedure. In the other cases the reliability of the concept is weighed on the computed value.

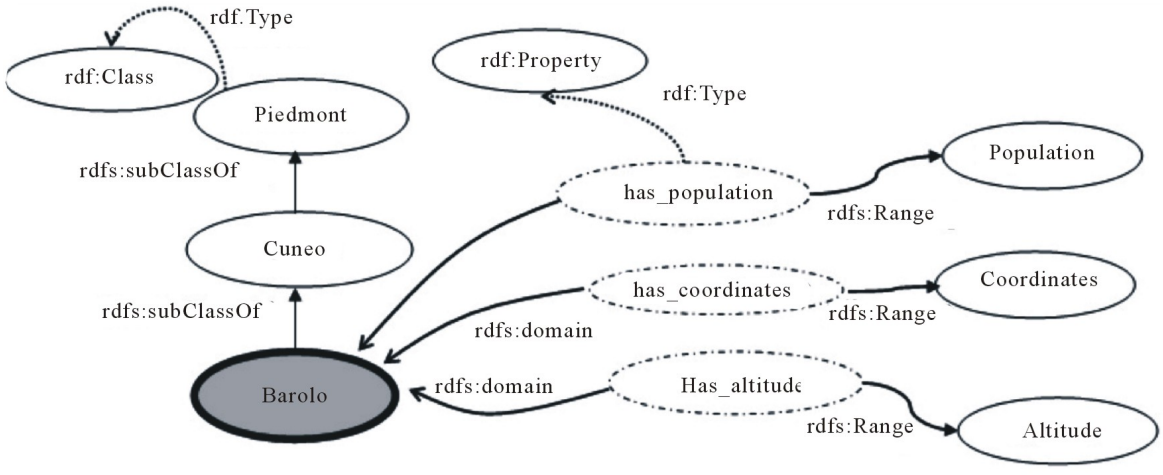

Figure 4. A portion of the domain ontology GeoLocal with the context of the concept Barolo.

4. A Service Architecture for User Model Interoperability

The UMIS service (**Figure 5**) is composed by a centralized set of tools **(**Enhanced User Model UDDI Registry (EUMUR**)** (Section 4.2) that UM-based Web Services (UM-WS) (Section 4.1) can exploit to exchange user data.

4.1. UM-Based Web Services

To join the framework, each UM-based web service (UM-WS) has to offer a *Web Service interface* and provide the basic WSDL operations required to interact with other services, and the interface to interact with the central registry (Section 4.2). We consider as mandatory at least the implementation of the WSDL operation *Statement get Value* (*user ID*, *feature*, *concept*) to inquiry for the value of the UM *feature* with respect to a domain *concept* for a certain user, where *Statement* is a complex type representing a Statement (Section 3). Then, a set of operations representing the moves specific for the games can be implemented, *i.e*. *inform*, *request*, *inquire*, *deny*, *agree*, *disagree*. Moreover the UM-WS has to refer to an ontology for the definition of the domain knowledge.

4.2. The Enhanced User Model UDDI Registry

The *Enhanced User Model UDDI Registry* (*EUMUR*) is a centralized shared component with the goal of offering a UDDI registry and a search service (Search Network Buffer).

4.2.1. UDDI Registry

It allows an application to know if another UM-WS shares the same ontology, which games it supports, and to start a conversation with it. It is composed by:

Communication Tools, the definitions of the Dialogue Games elements:

Conversation Protocol: the sequence of messages a UM-WS has to manage according to its role.

Content Tactics of the game.

Dialogue Game: the reference to the *Conversation Protocol* and to the *Content Tactics* of the game.

Service declaration, the list of the available UM-based Web Services, with the *Communication Tools* they support:

Service Name: the UM-WS's name.

Wsdl reference: the reference to the WSDL file with the operations it offers.

Ontologies: the domain ontologies it refers to.

Games: the Dialogue Games it is able to play.

4.2.2. Search Network Buffer

The Search Network Buffer (SNB) is a shared network space able to automatically match Requestors with Providers of specific UM features. The interaction model is managed according to the *publish*/*subscribe* pattern. All the UM-WSs *subscribe* to the SNB asking to be notified (as a Provider) when a certain kind of request arrives into the SNB. When a UM-WS (acting as a Requestor) looks for some user information, it *publishes* a request to the SNB with the desired feature. Then the SNB notifies all the services subscribed and the service having the request value can notify their availability to exchange this information always through the SNB (**Figure 6**).

In the SNB the user features *values* are not shared, since the interaction consists only in requests of User Model features, and in responses of availability. The ex-

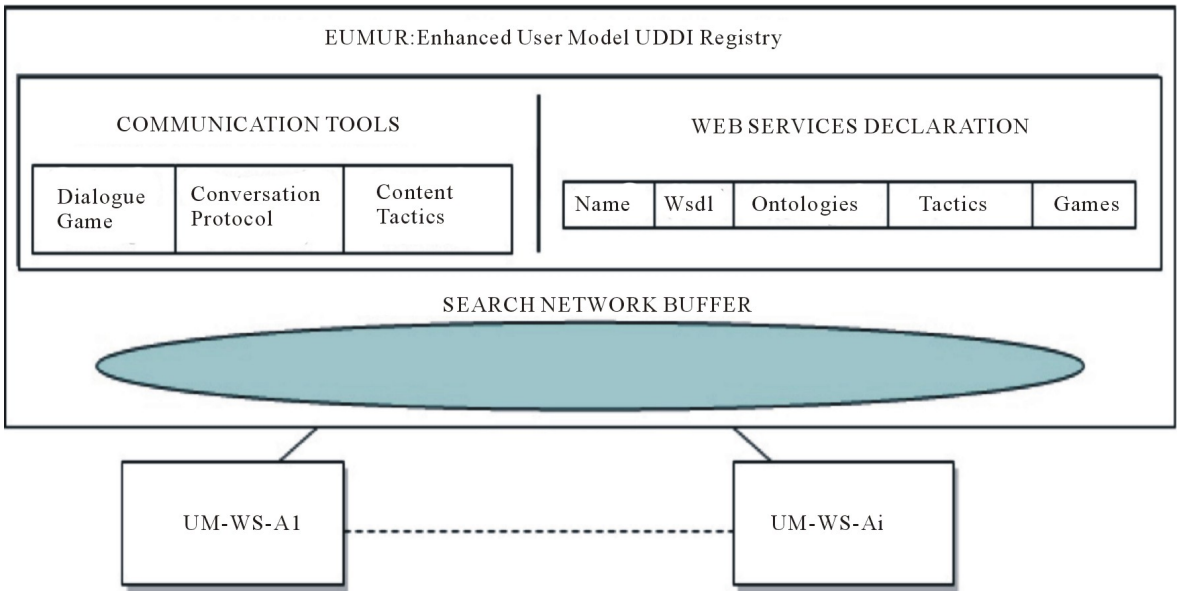

Figure 5. The framework architecture.

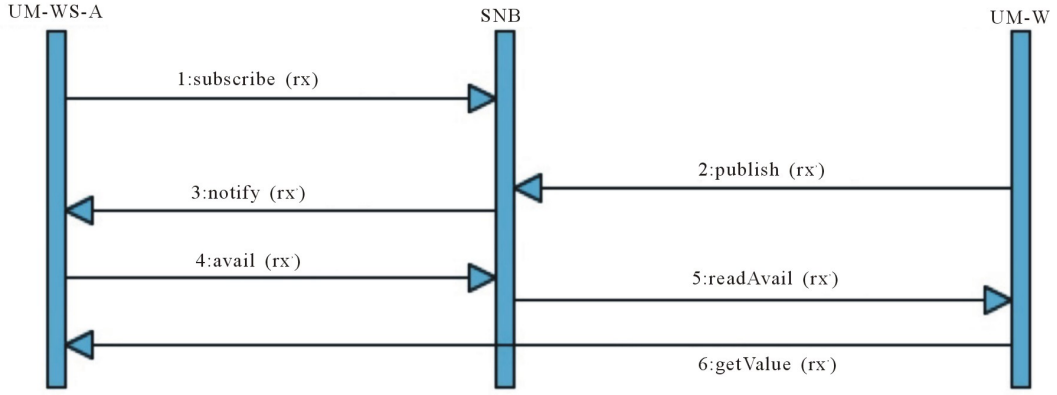

Figure 6. SNB sequence diagram.

change of the values will take place directly between the UM-WSs. In this way, the Requestor is free to select, according to its internal policies, which UM-WS to consult as Provider, and the Provider is free to apply its own privacy policies.

The general format of a request published on the SNB is: (*Sender Name*, *Inquiry*, *Interest*, *Type*, *Ontology*, *Object*, *User ID*) where in particular:

- *Type*: *is* the typology of the request: *byUri* (if the requested *object* refers to an ontology URI), and *byLabel* (if the *object* is expressed with a label);
- *Object*: if *Type* = *byUri*, it is the *URI* that identifies the requested concept; if *Type* = *byLabel*, it is the *Label* associated to the concept;

For instance if UM-WS-A needs the interest of the user *us44* in the concept *Barolo* of the ontology *Food*, it invokes on the SNB the operation: *publish* (*UM-WS-A*, *Inquiry*, *Interest*, *byUri*, *Food*, *null*, *Barolo*, *us44*). All the UM-WSs subscribed (by means of the operation: *subscribe* (*M-WS-X*, *Inquiry*, *Interest*, *byUri*, *Food*) will be notified by the SNB.

5. UMIS in Action

In this Sec. we describe an example of UMIS usage, presenting some technical details about the developed prototype and describing two use cases.

5.1. Prototype Implementation

To implement the *UDDI registry*, we chose to adopt *juddi*, an open source Java implementation of UDDI specification for Web Services. We then represent the additional information requested by our registry exploiting the mechanism offered by *tModel*, an abstraction for a technical specification of a service type offered by an UDDI registry.

To implement the *SNB*, we exploit the GigaSpaces middleware, which offers a robust implementation in of the JavaSpaces paradigm supporting the propagation of data items according to the Linda [23] tuple space model.

To implement the *Dialogue Game*, each application has to manage the status of the conversation and to select which action to execute, and to decide when the goal has been reached. These abilities are closely related to the business logic of each application and thus they can not be delegated to some external component. Thus, an *ad hoc* component managing the Dialogue Game has to be integrated in the application. For this purpose, we developed the *Dialogue Manager* module, which implements the logic of the Dialogue Games and maintains the status of each conversation instance. In order to execute the content tactics required by the games, it contains a special purpose java class with the needed general-purpose SPARQL queries over a OWL ontology, *i.e*. to extract the parents, the ancestors, the siblings, the children, the properties of a concept.

A goal of this prototype implementation is to make easier for existing applications to exploit UMIS. To minimize this effort, we developed a *UMIS Wrappers* component (**Figure 7**) composed by a set java-based libraries implementing a *Proxy IN Interface* (acting as a proxy between the systems and the framework), and a set of wrappers managing the interaction with the other components in the framework. Each wrapper offers to the Proxy IN Interface a set of APIs to interact with the component it manages, and invokes the APIs offered by the Proxy IN Interface to manage the requests coming from such components. The wrappers are the following ones:

WSwrapper: provides the implementation of the WSDL operations for communicating with the other services: it translates the SOAP messages into the format managed by the Proxy IN Interface. It also provides the Proxy IN Interface with the facilities to invoke the WSDL operations offered by the other services;

SNBwrapper: manages the communication with the SNB implementing the publish/subscribe pattern and

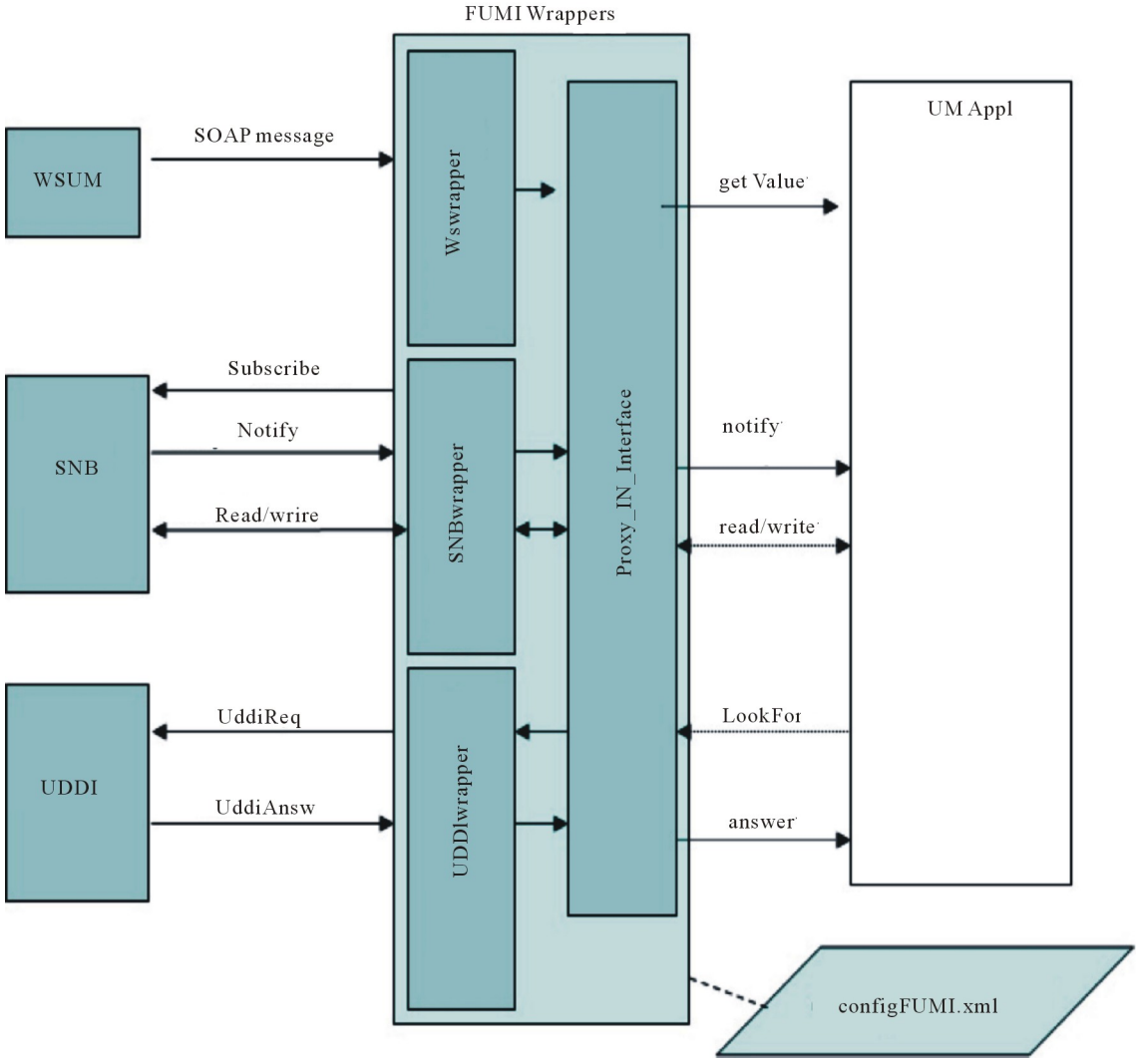

Figure 7. Service declaration in the use case.

mediating the interaction with the Proxy IN Interface;

UDDIwrapper: manages the communication with the UDDI registry, offering a set of APIs to query the registry.

We designed the Proxy IN Interface in order to decouple the defined wrappers from the technologies and the communication protocols used by the wrapped application. The independence of the wrapped application enables the integration of heterogeneous kind of adaptive applications, based on different environments and technologies. As showed in **Figure 7** a *configUMIS.xml* file is accessed by the UMIS Wrappers; it stores configuration information to be used at initialization time (e.g. the ontologies).

5.2. Use Case

In order to test our solution, we integrated an existing adaptive system, iCITY [9], a Turin events recommender[3] into UMIS. To this purpose we coded the ProxyINiCITY class, the implementation of the Proxy IN Interface that mediates between the existing iCITY REST interface and the UMIS Wrappers, and we filled the requested initialization information about iCITY. Then we made a change on the iCITY sw module for user data access, adding the invocation of the ProxyINiCITY to search for the missing data. We also added the DialogueManager component into iCITY, in order to start new games instance according to some simple rules: 1) a new Game is started each time is requested by an other application; 2) a new Clarification Game is started for each request that does not refer to a shared ontology; 3) a new Explorative Game is started when the reliability of the value is lower than 0.5 or when no one can give a response.

[3]iCITY is built on a servlet-based technology and implements some API REST in order to provide data about its users. It does not consider any invocation of external resources for retrieving the data it misses.

Then, we connected to UMIS two dummy applications, developed providing web service interfaces to existing internal repositories of user data: 1) WS-FOOD, in the food domain; and 2) WS-ART, in cultural events domain. In such an environment, we run our tests making such systems collaborate using UMIS. WS-FOOD and WS-ART share a subset of users with iCITY. **Figure 8** illustrates the entries in the *Web Services Declaration* component in the registry for the scenario.

As use case we consider WS-FOOD interacting with the user Roby. In particular, it needs his *Interest* in the concept *RedWine* in its domain ontology *Food* (**Figure 1**).

Discovery. WS-FOOD has to find another adaptive system, working in a similar domain and storing a model of Roby. Therefore, WS-FOOD publishes a request into the SNB implemented by the GigaSpace server: *publish* (*WS-FOOD*, *Inquiry*, *Interest*, *byUri*, *Food*, *#RedWine*, *Roby*). GigaSpace notifies the iCITY (its Gigawrapper component), which has been previously subscribed to this kind of request. iCITY has the requested value and publishes its availability on the SNB.

P2P request-response. WS-FOOD, reading such information from the SNB, directly contacts iCITY, by means of the provided WSDL interface (WSwrapper), asking for the value: *getValue* (*Roby*, *Interest*, *#RedWine*).

Dialogue-based negotiation for value derivation. The reliability of the value provided by iCITY is considered too low by WS-FOOD. In this situation, an *Explorative Game* can be played for finding other possible related concepts that can be used instead (Section 3.3.1). Looking at the registry WS-FOOD discovers that iCITY is able to play such a game. Following the conversation protocol of the game, WS-FOOD asks for Roby's interest in the parent concept of *RedWine*, in the children concepts, and then in the sibling concepts (tactic *tc*3).

Table 1 shows the obtained values and their reliability together with the similarity degree (column *sim*) with respect to the starting concept. Applying the value derivation defined in Section 3.2.1, the interest in *RedWine* is 0.74 with a reliability of 0.73.

Dialogue-based negotiation for concepts disambiguation. As second example, WS-ART has to suggest Roby a touristic tour for the weekend and it needs to know if he is interested in visiting the city of Barolo, represented as the concept *#Barolo* in its ontology *GeoLocal* (**Figure 4**). Since no system in the framework shares this ontology (no entry in the registry), it submits to the SNB the *byLabel* request: *publish* (*iCITY*, *inquiry*, *Interest*, *byLabel*, *Places*, *Barolo*, *Roby*). iCITY has the label *Barolo* associated to a concept in its ontology *Food* as wine and in its ontology *GeoLocal* as place. In order to figure out if one of the two concepts is the requested one, it exploits the *Clarification Game* (Section 3.3.2).

Since the concept *Barolo* in *GeoLocal* ontology (**Figure 4**) shares the property *has_coordinates* and the super-classes *Cuneo* and *Piedmont* with the concept in the *WS-ART* ontology, it concludes that they are the same concept.

6. Evaluation

In this section we discuss the evaluation of UMIS under different perspectives. First, we wanted to test the performance of UMIS communication process, *i.e*., the additional *network traffic* and the *communication cost* for the Requestor (Section 6.1). Second, we aimed at seeing how our solution affects the quality of the UM exchange, *i.e*., the effectiveness of the exchange (how many times the communication ends positively) (Section 6.2).

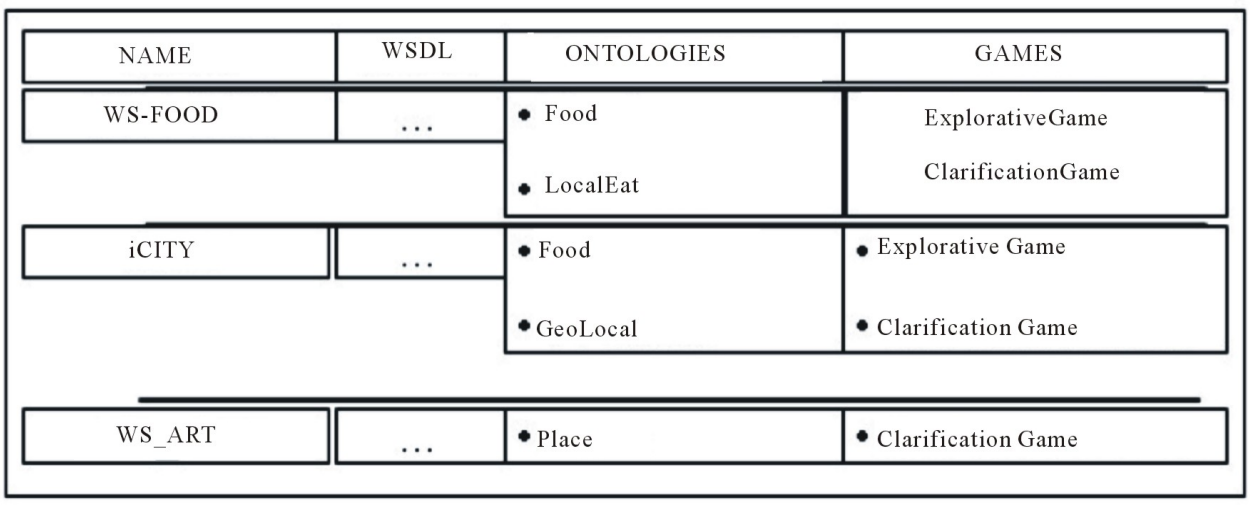

Figure 8. Service declaration in the use case.

Table 1. Value obtained for the Explorative Game.

Concept	Total features	Common features	*sim*	Value	Reliability
Wine	5	5	0.83	0.7	0.75
Barolo	16	7	0.60	0.9	0.7
WhiteWine	7	5	0.71	0.6	0.66

6.1. Performance Evaluation

We evaluated the UMIS communication process measuring: the network traffic and the communication cost.

Network traffic. We wanted to measure the additional *network tra*ffic generated by the introduction of UMIS in a decentralized collaboration. We compared in **Table 2** the number of data exchanged among iCITY and N Web Services (*WS*-1, *WS*-2, *WS-N*), when iCITY looks for some user value without using UMIS (row 1) and using UMIS (row 2). If UMIS is not used, iCITY has to directly contact each WS, hence it sends a maximum of N messages. Each WS-i can receive 1 message and the network traffic generated is of N messages in the worst case. Instead, if UMIS is used, iCITY always performs two interactions (publish the request and read the answer) with the SNB (2snb in the Table), plus one message exchanged with another WS (if some WS-i has the value). Each WS-i performs at most 2 interactions with the SNB (1 notification if subscribed for the request, and 1 writing of *avail*, only if it has the requested value) and only one WS-i can have 1 message exchanged with iCITY. The global network traffic generated is at most of 2N interactions with the SNB and 1 message between the services. We can conclude that even if the amount of exchanged data is growing using our approach, the difference strongly depends on the different cases[4]. However, the wider part of the traffic overhead is in charge of the SNB, which is specifically designed to scale on large numbers without loosing in performance.

Communication costs. We also examined the communication process in term of *communication cost* for the Requestor, with respect to a centralized/decentralized approach. To do so, we approximated the *cost* of each interaction with the SNB (*publish*, *notification*, *read*) as 1, and the cost of each SOAP message (*send* or *receive*) as 2 (due to their different complexity and dimension). **Table 3** summarizes the cost paid by Requestor, considering that, after a request to the SNB, M of the N systems in the framework keep the requested data. Row I shows the cost when for the Requestor the first obtained data is sufficient (*First*), Row II when the Requestor wants to look for all the available values (*All M*), and Row III when the requested data is not available at all (*Null*).

In a *centralized solution*, the cost is always 4 (2 for sending a request and 2 for receiving the value). In UMIS each system declares if it has the value, then in the *First* case the cost is always 6 (2 for the communication with the SNB (*publish* and *read*), and 4 for the communication with the service with the value), in the *All M* case is 1 for publishing, plus M for reading all the M answers and 4 M for interacting with each service. In the *Null* case is always 1 (request). In a pure decentralized solution the cost in the first case depends on when the services with the values are founded. In other cases each WS has to be contacted. In general, we can conclude that the convenience of our approach is proportional to the ratio N/M. It increases how much different data are scattered in the network: if each system keeps different UM fragments, the convenience is close to a centralized solution (and thus better than decentralized solutions); if all the systems have the same user data, it is worse than classical decentralized solutions.

6.2. Communication Effectiveness

As a second step, we wanted to verify how the use of UMIS impacts on the global effectiveness of communication for the exchange of UM, *i.e*. the requestor gains the desired value.

Testbed. As testing environment, we used the environment presented in Section 5 (iCITY, WS-FOOD, and WS-ART). The dataset is composed by data about 25 users shared by the systems. The UMs of the systems refer to three domain ontologies: *Food*, shared by iCITY and WS-FOOD (60 concepts), *Place* (258 concepts), used by WS-ART, and *GeoLocal* (180 concepts), used by iCITY.

Simulation parameters. In the test setting, iCITY needs the values of the user's interest in concepts of the shared ontology *Food* and of the non shared ontology *GeoLocal*.

Metrics. In each test, we measured the *quality of the exchange* (QoE), defined in [25] as the number of exchange that succeeded[5], $n(succ)$, out of the total number of exchange, $n(tot)$ (Formula (5)):

$$QoE = \frac{n(succ)}{n(tot)} \tag{5}$$

[4]For instance, when the searched value is not known by any applications, if iCITY does not use the framework, it will generate N messages without any result; instead, using the framework, apart from the publication on the SNB, no other message is exchanged.

[5]We define as "successful exchange" when the requestor gains the desired value.

Table 2. Network traffic.

#	*iCITY*	*WS*-1	*WS*-2	*WS-N*	*Network*
1	(Nmsg)	(1msg)	(1msg)	(1msg)	(N)
2	(2snb + 1msg)	(2snb + 1msg)	(2snb + 1msg)	(2snb + 1msg)	(2Nsnb + 1msg)

Table 3. Communication cost for the Requestor.

#	UMIS	centralized	decentralized
First	6	4	[1⋯4(N-M)]
All M	1 + M + 4*M	4	4N
null	1	4	4N

We performed two tests: one implementing the Explorative Game and one implementing Clarification Game.

Test 1: UMIS with Explorative Game. In the first test we aimed at verifying the usefulness of UMIS in 1) coping with missing responses; and 2) how it impacts on the effectiveness of exchanging UM data.

We measured the *quality of the exchange* (QoE) considering an exchange as successful when the derived value has *reliability* > 0.5. We simulated in iCITY the need of the values of user's *Interest* in 40 concepts in the shared ontology *Food*. First, iCITY asks the SNB, discovering that the data are available on other systems 24 times (60%). In the remaining 16 cases it start an Explorative Game, which ends with a derivation of values considered satisfactory by iCITY 8 times (50%). In this case, the quality of exchange is $QoE = n\,(sat)/n(tot) = 8/16 = 0.5$.

With respect to the effectiveness of communication, the *quality of exchange* (calculated over the initial 40 concepts) using UMIS is 0.8: $QoE = n\,(sat)/n\,(tot) = 32/40 = 0.8$. Instead, without the use of UMIS, it would be lower ($QoE = 24/40 = 0.6$) since the communication ends every time there are missing values.

Test 2: UMIS with Clarification game. The second test aimed at verifying if UMIS 1) reaches the disambiguation goal; and 2) how it affects the effectiveness of the global communication.

We measured again the *quality of exchange* resulted by the application of the clarification game, considering the data exchange as successful when the concept is correctly disambiguated. We simulated in iCITY the need of the value of the user *Interest* in 20 concepts of its not shared ontology *GeoLocal*. **Table 4** summarizes the results of the Clarification Game execution in these cases. In the table the column "answer" keeps the percentage of the time the game returns positive answers (same concept, 12 cases) or negative (different concept, 8 cases). The columns "correct" and "incorrect" represent for each row the correctness of the final disambiguation process. The *quality of exchange* in this case is: $QoE = n\,(correct)/n\,(tot) = 14/20 = 0.7$.

With respect to the *effectiveness of communication*, the UM communication is improved by using UMIS. In fact, without this game, the communication would be failed in all the 20 cases. To conclude, we can claim that, according to our test results, the UM communication will be improved by using UMIS implementing a clarification game with respect to the *quality of exchange*.

7. Related Work

Different research fields are involved in this work: interoperability architecture, semantic interpretation, mapping and communication.

Interoperability architecture. Starting from the beginning of the Nineties, two main approaches gain ground in the literature for UM interoperability depending on the way the User Model is stored and exchanged [8]. In *centralized* architectures [26], user data are collected in a central shared space and delivered to the applications through a client-server architecture (e.g. UMS [27]). This solution is widely adopted since it allows different applications to access the same user knowledge and mobile applications on devices with limited memory to exploit a UM. However, such centralized approach imposes a set of user features and a non-negotiable format of representation and protocols. Instead, in the *decentralized* approach [5], each system maintains a user model and interact with other systems in a peer-to-peer way (e.g. SUMI [28]). Advantages are flexibility in managing privacy and efficiency in obtaining only the necessary data. UMIS can be considered as a mixed approach: it is decentralized (each system has its UM and exchange data in a p2p way) and centralized (it offers a centralized discovery tool).

Semantic interpretation. We can distinguish two main approaches for data interpretation: *common representation* and *conversion*. In the first one, a uniform representation of the UM is required [22]. This is very restrictive

Table 4. Results of clarification game.

	Answer	Correct	Incorrect
Same concept	60% (12)	83% (10)	17% (2)
Different concepts	40% (8)	50% (4)	50% (4)

for applications, but it does not suffer from semantic ambiguities. The second approach instead lets free each application to represent UM as it prefers and then to use suitable techniques to ensure semantic interoperability (e.g. Mediator [29], IDIUMS [30]). UMIS follows this second approach, since we chose not to impose to use a unique ontology, and thus some form of concepts mapping is necessary. A comparison with most relevant related work with respect to this two features (interoperability architecture and semantic interpretation) is reported in **Table 5**.

The main contribution of this work is mainly related to mapping and communication modalities.

Mapping. In the literature, different methodologies for mapping have been proposed [31], from schema-based [32] to instance-based [33]. In our approach, each application can *negotiate* for reaching an agreement over not shared concepts using a negotiation protocol (Clarification Game) for concepts disambiguation, based on schema-based mapping and graph-based techniques [34]. With respect to standard mapping solutions, our approach, even if more suitable for small ontologies, it is ready-to-use, allowing to perform a basic form of ontology mapping at a run time, not requiring a pre-process task on ontologies or complex form of reasoning. Similarly to us, other approaches proposed to use negotiation through a dialogue [35] in order to enhance standard mapping approaches. For example, [36] solves semantic differences at run-time using the WordNet lexicon[6]; [37] presents an ontology negotiation protocol which enables agents to exchange parts of their ontologies by a process of interpretations, clarifications, and explanations; [38] presents a method for accepting a set of candidate mapping correspondences among concepts of different ontologies according to agents preferences and beliefs.

We use the dialogue model also to derive the value in the answers using related concepts (in Explorative Game). With respect to more complex forms of value propagation (e.g. Bayesian Networks) that requires a remarkable configuration efforts by applications, our solution can be more easily adopted by applications to guess missing values.

Communication. With respect to st andard request-response communication of Web service, the dialogue model is a mean for enriching the ineraction of Web Services with negotiation capabilities. The use of Speech-Act-based approach coming from multi-agents fields to specify the conversation flow between Web Services allows extending the interface in a direction of more flexibility and dynamism, managing more-than-two-turns interactions and defining different communication behaviors as response to messages [39,40].

One central aspect of our framework is an enhanced UDDI registry offering a discovery tool (SNB). The idea to enrich UDDI registry with semantic information is not new in Semantic Web community [41], but in general the goal is to describe the services in order to facilitate the discovery. Our registry works in a more bounded context, where all the services exchange UM knowledge, and thus here the need is limited to discover who maintains the desired user data and how to interact with it. Regarding our discovery mechanism, other works proposed a central repository which acts as a broker system, such as [42,43]. Other approaches propose a similar solution but in a totally decentralized perspective, such as [44].

8. Conclusions

The paper proposed UMIS, a set of services that support web-based user-adaptive applications in sharing user model data, in a loosely-coupled way and with little changes in their internal logics and knowledge representation. The main contribution of the work is to support the process of interoperability in three phases: 1) it provides an efficient centralized service discovery tool; 2) it offers an effective simple basic model of interaction for the exchange of UM value in a p2p way; 3) it offers a negotiation mechanism that can be used when the second step fails, in case of semantic ambiguities or missing (or non satisfactory) responses.

UMIS has been conceived to support interoperability among content-based adaptive applications representing their User Models by means of property-based linear parameters. However, a framework for interoperability becomes as more useful as it is able to increase the number of heterogeneous systems supported; in the UM interoperability context this means to support systems using different User Model representations. The basic principles of the framework logic makes it able to deal with different types of heterogeneity: different kinds of users information (both domain-dependent and independent), different user model representation (such as probabilistic model), different adaptive applications (both content-based and collaborative-filtering based) [24].

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

Table 5. A comparison with most relevant related work.

	Architecture	Data interpretation
UMIS	Mixed	Conversion
UMS (2006)	Centralized	Common representation
Mediator (2009)	Mixed	Conversion
SUMI (2009)	Mixed	Common representation
IDIUMS (2011)	Centralized	Conversion

As seen in Sections 4 and 5, UMIS architecture offers a certain level of flexibility. First, the UMIS Wrapper architecture provides a high level of decoupling between the components of the framework. The ProxyINInterface allows an easy integration of heterogeneous systems based on different technologies. Second, the SNB event-based model gives the possibility to add new components (or modify existing ones) in a seamless way. In order to enable applications to collaborate in this environment we provide them with a set of software tools to reduce their integration efforts. However, part of the integration remains still in charge of the applications that have to make their custom proxy implementation (Section 5). UMIS uses Dialogue Game model as a *negotiation technique* to solve semantic ambiguities. However, other modalities of ontology mapping could be easily included in the framework. For instance, the event-based model implemented by the SNB can be used as a blackboard system [45], where an OntologyMapper service can offer some ontology mapping capabilities to the UM-WSs in the framework.

Finally, we can mention the management of the possible novel issues the Social Semantic Web context [46,47] (e.g. privacy management, users reputation and systems proof).

REFERENCES

[1] P. Brusilovsky, "Methods and Techniques of Adaptive Hypermedia," *User Modeling and User-Adapted Interaction*, Vol. 6, No. 2-3, 1996, pp. 87-129. doi:10.1007/BF00143964

[2] M. Morita and Y. Shinoda, "Information Filtering Based on User Behavior Analysis and Best Match Text Retrieval," *SIGIR* '94: *Proceedings of Conference on Research and Development in Information Retrieval*, Springer-Verlag New York, Inc., New York, 1994, pp. 272-281.

[3] M. J. Pazzani, "A Framework for Collaborative, Content-Based and Demographic Filtering," *Artificial Intelligence Review*, Vol. 13, No. 5-6, 1999, pp. 393-408. doi:10.1023/A:1006544522159

[4] IEEE, "IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries," Institute of Electrical and Electronics Engineers, 1990.

[5] J. Vassileva, "Distributed User Modelling for Universal Information Access," *Universal Access in Human-Computer Interaction*, Vol. 3, 2001, pp. 122-126.

[6] S. Berkovsky, "Ubiquitous User Modeling in Recommender Systems," *Proceedings of the* 10*th International Conference on User Modeling*, Edinburgh, 24-29 July 2005, pp. 496-498.

[7] G. Salton and M. McGill, "Introduction to Modern Information Retrieval," McGraw-Hill Book Company, New York, 1984.

[8] F. Carmagnola, F. Cena and C. Gena, "User Model Interoperability: A Survey," *User Modeling and User-Adapted Interaction Journal*, Vol. 21, No. 3, 2001, pp. 285-331.

[9] F. Carmagnola, F. Cena, L. Console, O. Cortassa, C. Gena, I. Torre, A. Toso and F. Vernero, "Tag-Based User Modeling for Social Multi-Device Adaptive Guides," *User Modeling and User-Adapted Interaction*, Vol. 18, No. 3, 2008, pp. 478-506.

[10] F. Cena, "Integrating Web Service and Semantic Dialogue Model for User Models Interoperability on the Web," *Journal of Intelligent Information Systems*, Vol. 36, No. 2, 2011, pp. 131-166. doi:10.1007/s10844-010-0126-3

[11] F. Cena and R. Furnari, "A Model for Feature-Based User Model Interoperability on the Web," *Advances in Ubiquitous User Modeling*, Springer-Verlag, Berlin, 2009, pp. 37-54. doi:10.1007/978-3-642-05039-8_3

[12] G. Alonso, F. Casati, H. Kuno and V. Machiraju, "Web Services-Concepts, Architectures and Applications," Springer-Verlag, 2004.

[13] T. Berners-Lee, J. Hendler and O. Lassila, "The Semantic Web," Scientific American, 2001, pp. 28-37.

[14] B. Smith, "The Blackwell Guide to the Philosophy of Computing and Information Blackboard," 2004. http://www.blackboard. com/us/index.aspx

[15] V. Dimitrova, "Interactive Open Learner Modeling," Ph.D. Dissertation, Leeds University, Leeds, 2001.

[16] J. Levin and J. Moore, "Dialogue Games: Meta-Communication Structures for Natural Language Interaction," *Cognitive Science*, Vol. 1, No. 4, 1977, pp. 395-420. doi:10.1207/s15516709cog0104_2

[17] J. Searle, "What Is a Speech Act," *Language and Social Context*, Penguin Books, Baltimore, 1972.

[18] D. Lin, "An Information-Theoretic Definition of Similarity," *Proceedings of the International Conference on Machine Learning* (*ICML*), Morgan Kaufmann Publishers Inc., San Francisco, 1998, pp. 296-304.

[19] P. Resnik, "Using Information Content to Evaluate Semantic Similarity in a Taxonomy," *Proceedings of the International Joint Conference on Artificial Intelligence*, Montréal, 21 August 1995, pp. 448-453.

[20] R. Dieng and S. Hug, "Comparison of 'Personal Ontologies' Represented through Conceptual Graphs," *Proceedings of the European Conference on Artificial Intelligence*, 2005, pp. 341-345.

[21] M. Ehrig and Y. Sure, "Ontology Mapping—An Integrated Approach," *Proceedings of the European Semantic Web Symposium*, 2005, pp. 76-91.

[22] D. Heckmann, "Ubiquitous User Modelling," Ph.D. Dissertation, University of Saarlandes, Saarlandes, 2005.

[23] S. Ahuja, N. Carriero and D. Gelernter, "Linda and Friends," *Computer*, Vol. 19, No. 8, 1986, pp. 26-34. doi:10.1109/MC.1986.1663305

[24] J. B. Schafer, D. Frankowski, J. L. Herlocker and S. Sen, "Collaborative Filtering Recommender Systems," *The Adaptive Web*, Vol. 4321, 2007, pp. 291-324. doi:10.1007/978-3-540-72079-9_9

[25] D. Chen, B. Vallespir and N. Daclin, "An Approach for Enterprise Interoperability Measurement," MoDISE-EUS, 2008, pp. 1-12.

[26] A. Kobsa, "User Modeling in Dialog Systems: Potential and Hazards," *AI and Society*, Vol. 4, No. 3, 1990, pp. 214-240. doi:10.1007/BF01889941

[27] A. Kobsa and J. Fink, "An LDAP-Based User Modeling Server and Its Evaluation," *User Modeling and User-Adaptive Interaction*, Vol. 16, No. 2, 2006, pp. 129-169.

[28] D. Kyriacou, H. C. Davis and T. Tiropanis, "A (Multi-Domainsional) Scrutable User Modelling Infrastructure for Enriching Lifelong User Modelling," *Lifelong User Modelling Workshop*, 17*th International Conference*, Trento, 2009.

[29] S. Berkovsky, T. Kuflik and F. Ricci, "Mediation of User Models for Enhanced Personalization in Recommender Systems," *User Modeling and User-Adapted Interaction*, Vol. 18, No. 3, 2008, pp. 245-286. doi:10.1007/s11257-007-9042-9

[30] R. Prince and H. Davis, "IDIUMS: Sharing User Models through Application Attributes," *Proceedings of* 19*th International Conference*, Girona, 11-15 July 2011.

[31] P. Shvaiko and J. Euzenat, "A Survey of Schema-Based Matching Approaches," *Journal on Data Semantics* (*JoDS*), Vol. 4, 2005, pp. 146-171.

[32] Y. Kalfoglou and M. Schorlemmer, "Ontology Mapping: the State of the Art," *Knowledge Engineering Review Journal*, Vol. 18, No. 1, 2003, pp. 1-31. doi:10.1017/S0269888903000651

[33] F. Wiesman, N. Roos and V. P. Vogt, "Automatic Ontology Mapping for Agent Communication," *AAMAS*02, 2002.

[34] A. Maedche and S. Staab, "Measuring Similarity between Ontologies," *Proceedings of European Conference on Knowledge Acquisition and Management*, Madrid, 1-4 October 2002.

[35] L. Li, Y. Yang and B. Wu, "Agent-Based Ontology Mapping towards Ontology Interoperability," Springer, Berlin, 2005, pp. 843-846.

[36] B. Orgun, M. Dras, S. Cassidy and A. Nayak, "Reaching Agreement over Ontology Alignments," *Proceedings of Australasian Ontology Workshop*, Vol. 58, 2005, pp. 75-82.

[37] S. Bailin and W. Truszkowski, "Ontology Negotiation: How Agents Can Really Get to Know Each Other," *Proceedings of WRAC* 02, 2002.

[38] L. Laera, V. Tamma, J. Euzenat, T. Bench-Capon and T. Payne, "Reaching Agreement over Ontology Alignments," *Proceedings of ISWC* 2006, Springer, Berlin, 2006, pp. 371-384.

[39] J. E. Hanson, P. Nandi and D. W. Levine, "Conversation-Enabled Web Services for Agents and e-Business," *International Conference on Internet Computing*, Las Vegas, 24-27 June 2002, pp. 791-796.

[40] L. Ardissono, G. Petrone and M. Segnan, "A Coversational Approach to the Interaction with Web Services," *Computational Intelligence*, Vol. 20, No. 4, 2004, pp. 693-709. doi:10.1111/j.0824-7935.2004.00261.x

[41] M. Paolucci, Kawamura, T. R. Payne and K. Sycara, "Importing the Semantic Web in UDDI," *WES*, *LNCS*, Vol. 2512, Springer, Berlin, 2002, pp. 225-236.

[42] D. L. Musa and J. P. M. de Oliveira, "Sharing Learner Information through a Web Services-Based Learning Architecture," *Journal of Web Engineering*, Vol. 4, No. 3, 2005, pp. 263-278.

[43] V. I. Chepegin, L. Aroyo, and P. De Bra, "Discovery Service for User Models in a Multi-Application Context," *ICALT*, 5*th IEEE International Conference on Advanced Learning Technologies*, Kaohsiung, 5-8 July 2005, pp. 956-957. doi:10.1109/ICALT.2005.126

[44] M. Specht, A. Lorenz and A. Zimmermann, "Towards a Framework for Distributed User Modelling for Ubiquitous Computing," *DASUM* 2005 *at UM*05, 10*th International Conference of User Modeling*, Edinburgh, 16-20 July 2005.

[45] H. P. Nii, "Blackboard Systems," In: A. Barr, P. R. Cohen and E. A. Feigenbaum, Eds., *The Handbook of Artificial Intelligence*, Vol. 4, Addison-Wesley, Reading, 1989, pp. 1-82.

[46] A. Aroyo and G.-J. Houben, "User Modeling and Adaptive Semantic Web," *Semantic Web Journal*, Vol. 1, No. 1-2, 2010, pp. 105-110.

[47] F. Orlandi, J. Breslin and A. Passant, "Aggregated, Interoperable and Multi-Domain User Profiles for the Social Web," 8*th International Conference on Semantic Systems*, Graz, 5-7 September 2012.

