A semantic-based fully visual application for context-aware matchmaking and request refinement in ubiquitous computing

Michele Ruta, Tommaso Di Noia, Eugenio Di Sciascio, Floriano Scioscia

Politecnico di Bari
via Re David 200, I-70125
Bari, ITALY
[m.ruta,t.dinoia,disciascio,f.scioscia]@poliba.it

Abstract. This paper presents a mobile visual application aimed at fully exploiting semantics of request descriptions to enable advanced services. Distinguishing aspects of its underlying framework include semantic ranking of request results and logic-based explanation of matchmaking outcomes as well as the context identification and exploitation to select suitable resources. The GUI has been designed and implemented to be effective for handheld devices with reduced screen capabilities. It requires no knowledge of any logic principle to be fully used. The framework is general-purpose and could be easily adapted to different mobile scenarios. Here it is motivated and presented in a tourism case study.

1 Introduction and Motivation

In ubiquitous computing scenarios, information technology can assist users in discovering resources examining alternatives and analyzing choices, thus aiding people to retrieve information satisfying their needs and/or giving more elements to make rational decisions. Especially in some "evanescent" scenarios, to discover a service/resource could be a very difficult issue: finding items encountering user needs often requires too much effort and time. This is particularly true when a user has a vague idea of what she wants, having limited knowledge of a specific domain.

When an effective network infrastructure is lacking, the process of supporting the user searching goods or services is an even more challenging subject. Basically, several issues concerning traditional service discovery are exasperated in the case of ubiquitous and pervasive approaches because of the host mobility and limited capabilities of mobile devices. Small displays, uncomfortable input methods, reduced memory availability and computational capacities restrain the exploitation of such applications. Nevertheless the rising potentialities of wireless-enabled handheld devices provide the necessary basic requirements for implementing flexible discovery frameworks. The aim is to reduce the human effort in discovering resources and/or services, also granting an acceptable level of accuracy and coping with user mobility and heterogeneity of pervasive scenarios. We present a "pocket" application enabling a semantic-based discovery

in ubiquitous environments. The main goal we pursue is to allow users equipped with handheld devices to take advantage of semantic resource annotation and matchmaking as well as of logic-based ranking and explanation services, while hiding all technicalities from them and letting users interact with the system without requiring dependable wired infrastructures. The system we propose is a general-purpose mobile DSS (Decision Support System). It can be exploited by tourists performing a visit but also by a customer wanting to remotely retrieve goods or services getting them from a generic marketplace and so on. The knowledge domain is encapsulated within a specific ontology the user must select at the beginning of her interaction with the system.

A relevant feature of the proposed approach is the context awareness. It allows to perform a selective resource discovery based on proximity which is especially useful in the case of some specific mobile applications as the virtual guide presented here [1]. Recall that users equipped with PDAs or smartphones are dipped within a pervasive environment; hence they could be specifically interested to resources or services near them. Hence, during the discovery, resources/services close to the user should be ranked better than the ones far off (supposing an equivalent semantic distance from the request). In other words, the *semantic distance* between request and offered resource should be properly rectified taking into account the *physical distance* occurring between user and resource itself (if it has an environmental collocation). In the proposed virtual guide application, this feature has been obtained by means of the integration of a localization module within the tool. The application recognizes the user location and grades matchmaking outcomes according to vicinity criteria.

The retrieval process is accomplished across multiple steps. Request formulation is the first phase. It is critical especially in case of ontology-based systems as the one we present here. The query language has to be very simple but, at the same time, its expressiveness must allow to correctly interpret user requirements so retrieving only what the user is really looking for. In most cases users are unable to exploit logic formulas needed to use a formal ontology; they want a simple visual representation to manipulate the domain of interest. The overall system should be able to rapidly match resources to potential beneficiary interests and to present available resources in an appealing manner –compatible with restricted screens– facilitating examination and checking of their features.

We borrow techniques and ideas from the Semantic Web initiative and adapt them to volatile wireless scenarios. Semantic Web technologies applied to ubiquitous computing open extremely interesting new possibilities, including: formalization of annotated descriptions that become machine understandable so enabling interoperability; reasoning on descriptions and inference of new knowledge; validity of the Open World Assumption (OWA), overcoming limits of structured data models; possibility of going beyond physical boundaries of structured and fixed network infrastructures. By means of formal ontologies, modeled using OWL[1], knowledge about a specific domain can be modeled and exploited in

[1] OWL Web Ontology Language, W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-features/

order to derive new information from the one stated within metadata associated to each resource. Modeling the interest domain using an OWL ontology, the user is able to browse the related knowledge starting form "her vague idea" about the resource she wants to discover. By means of a preliminary selection of the reference ontology, the user fixes the scenario so conditioning the following interactions with the system. Different sessions in the application exploitation could refer to different ontologies and then could entail interaction with the system aimed at different purposes. For example a generic client could use the application as a pocket virtual guide for tourist purposes selecting a cultural heritage ontology and in a further phase after concluding her visit she can adopt it as a mobile shopping assistant to buy goods in a B2C m-marketplace: in that case she will select an e-commerce ontology. Once the request has been formalized w.r.t. the reference ontology, its formal relations are exploited in order to find resources able to satisfy user necessities. Based on the formal semantics of both the request and the returned resource/service descriptions, an explanation of the matchmaking outcome is then provided to the user to foster further interaction.

Main features of the proposed approach are: full exploitation of non-standard inferences to enable explanation services; semantic-based ranking of retrieved resources; fully graphical and usable interface with no prior knowledge of any logic principles; total absence of physical space-temporal bonds in system exploitation.

The framework we present here has been implemented within the Apulia project PE_082 "IC Technologies for tourist assistance through interactive consultancy of a virtual guide". It adopts the semantic-enhanced features described above and uses them to provide various user-oriented services, which benefit of a Knowledge Representation approach.

The remaining of the paper is structured as follows: in the next section, basics of matchmaking and exploited Description Logics inference services are briefly recalled. Section 3 outlines system features and architecture by means of a case study referred to a tourism scenario. In Section 4 we comment on related work. Finally conclusion and future work terminate the paper.

2 Matchmaking Principles

Main matchmaking issues emerge when the resource discovery is not limited to identity matches (which are probably too rare to be realistic in concrete applications), that is when the goal is to find resources suitable to grant also a not-complete satisfaction of the user request, in order to satisfy user needs "to the best possible extent". Semantic matchmaking is basically the process of finding best matches of a request among available resources, where both the request and the resource descriptions are semantically annotated w.r.t. a reference *ontology* (*i.e.*, a shared interpretation of the knowledge domain).

Knowledge Representation (KR) approaches to matchmaking usually exploit classical deductive services, namely *classification* and *consistency*. Basically, given a request/resource pair opportunely annotated, classification allows to check whether all request specifications are included within the resource de-

scription, whereas consistency verifies whether some specification in the request contradicts (some of) the ones within the resource annotation. In both cases, the response is then a binary *true/false* value. Although these inference services are very useful in the early phases of a discovery process, they are not sufficient to rank a set of resources with respect to a request. A positive classification corresponds to a resource completely satisfying the request, yet a) such a match might not occur during a discovery process; b) various resources could be "almost" classified with respect to a request. A matchmaker may then order resources according to their "distance" from the request and it may hypothesize a semantic based explanation for the causes of mismatch. In an analogous way, when an inconsistency occurs, a matchmaker may: a) order resources, putting before the ones "less conflicting" with the request; b) provide an explanation whether the request was completely unsatisfiable or only some part of it was in conflict with available resources.

Given R (for Request) and O (for Offer) both consistent with respect to an ontology $\mathcal{T}$, logic-based approaches to matchmaking proposed in the literature [2, 3] use classification and consistency to grade match results in five categories:
– **Exact**. All the features requested in R are exactly the same provided by O and vice versa –in formulae $\mathcal{T} \models R \Leftrightarrow O$.
– **Full-Subsumption**. All the features requested in R are contained in O –in formulae $\mathcal{T} \models O \Rightarrow R$.
– **Plug-In**. All the features offered in O are contained in R –in formulae $\mathcal{T} \models R \Rightarrow O$.
– **Potential-Intersection**. There is an intersection between the features offered in O and the ones requested in R –in formulae $\mathcal{T} \not\models \neg(R \wedge O)$.
– **Partial-Disjoint**. Some features requested in R are conflicting with some offered in O –in formulae $\mathcal{T} \models \neg(R \wedge O)$.

To better clarify differences among previous match types, let us suppose a tourist is performing a visit and she is interested in seeing "medieval palaces with courtyards" (this is the previously called **R**). If there is a resource $\mathbf{O_{exact}}$ annotated as "medieval palace with a courtyard", R and O coincide. From the matchmaking perspective, **Exact** match is obviously the best, because both R and O express the same preferences and, since all the resource characteristics requested in R are semantically implied by O (and vice versa), the user finds exactly what she is looking for. On the contrary, if we have $\mathbf{O_{full}}$ annotated as "medieval palace with a courtyard and frescoed roofs", all requirements in R are satisfied by O, but other non-conflicting characteristics are also specified in the returned resource. In a **Full** match all the interpretations for O are surely also interpretations for R and then O completely satisfies R. This means that all the resource characteristics requested in R are semantically implied by O but not, in general, vice versa. Then, in a full match, O may expose some unrequested characteristics. From the point of view of requester this is not a bad circumstance. In fact, characteristics she was looking for are already satisfied. If the provided resource is $\mathbf{O_{plug-in}}$ simply labeled as "medieval palace", all characteristics in O were required by R, but the requirement of a courtyard is not explicitly satisfied.

Plug-In match expresses the circumstance when O is more generic than R, and then it is possible that the latter can be satisfied by the former. Some characteristics in R are not specified, implicitly or explicitly, in O. This is surely more appealing for the provider than for the requester. As said, here we adopt the OWA. In case the returned resource is $\mathbf{O_{potential}}$ annotated as a "medieval palace with a frescoed roof", neither all elements in R are in O nor vice versa. R and O are still compatible, since an explicit conflict does not occur. With **Potential** match we can only say that there is some similarity between O and R, and then O might potentially satisfy R. Probably some features of O are underspecified in its description. So the requester has to contact the provider if she wants to know something more about them. Finally, supposing $\mathbf{O_{partial}}$ is a "medieval church with a courtyard", a requirement in R is explicitly violated by O, making the provided resource incompatible with the request. **Partial** match states that R and O are conflicting, yet notice that the disjointness between them might be due only to some –maybe negligible from the requester's standpoint– incompatible characteristics. Hence, after a revision of opposed features, an agreement can be reached.

As previously discussed, usually logic-based approaches only allow a categorization within match types. But while exact and full matches can be rare, a user may get several potential and partial matches. Then a useful logic-based matchmaker should provide a –logic– ordering of available resources w.r.t. the request, but what we get using classification and consistency is a boolean answer. Also partial matches might be just "near miss", *e.g.*, maybe just one requirement is in conflict, but a pure consistency check returns a hopeless *false* result, while it could be interesting to order "not so bad" ads according to their similarity to the request.

2.1 DL Inference Services for Matchmaking

In the proposed approach we exploit Description Logics (DLs) –whose formal semantics is at the basis of the Ontology Web Language OWL-DL. Furthermore, there is a strict correspondence between the OWL-DL syntax and DIG 1.1 [4], but DIG is less verbose and more compact. Here we formalize examples by adopting DL syntax instead of OWL for the sake of compactness. DLs are a family of logic formalisms for Knowledge Representation [5] whose basic syntax elements are *concept names*, *role names*, *individuals*. Intuitively, concepts stand for sets of objects in the domain, and roles link objects in different concepts. Individuals are used for special named elements belonging to concepts.

Concepts can be used in *inclusion assertions* $O \sqsubseteq D$, and *definitions* $O \equiv D$, which impose restrictions on possible interpretations according to the knowledge elicited for a specific domain. The semantics of inclusions and definitions is based on set containment: an interpretation $\mathcal{I}$ satisfies an inclusion $O \sqsubseteq D$ if $O^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, and it satisfies a definition $O \equiv D$ when $O^{\mathcal{I}} = D^{\mathcal{I}}$. A DL theory (a.k.a. TBox) is a set of inclusion assertions and definitions. A *model* of a TBox $\mathcal{T}$ is an interpretation satisfying all inclusions and definitions of $\mathcal{T}$.

DL-based systems usually provide at least two basic reasoning services:
– **Concept Satisfiability**: $\mathcal{T} \models R \not\sqsubseteq \bot$. Given a DL theory $\mathcal{T}$ and a concept R, does there exist at least one model of $\mathcal{T}$ assigning a non-empty extension to R?
– **Subsumption**: $\mathcal{T} \models R \sqsubseteq O$. Given a DL theory $\mathcal{T}$ and two concepts R and O, is R more general than O in any model of $\mathcal{T}$?
Nevertheless –given a DL– the matchmaking issues outlined above call for other, non-monotonic inference services that we briefly recall hereafter. Let us consider concepts R and O and an ontology $\mathcal{T}$. If a partial match occurs, *i.e.*, they are not compatible with each other with respect to $\mathcal{T}$, one may want to retract some specifications in R, G (for *Give up*), to obtain a concept K (for *Keep*) such that $K \sqcap O$ is satisfiable in $\mathcal{T}$.

In [6] the Concept Contraction problem was first defined as:
– **Concept Contraction**. Let $\mathcal{L}$ be a DL, R, O be two concepts in $\mathcal{L}$ and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$, where both R and O are satisfiable in $\mathcal{T}$. A *Concept Contraction Problem* (CCP), identified by $\langle \mathcal{L}, R, O, \mathcal{T} \rangle$, consists of finding a pair of concepts $\langle G, K \rangle \in \mathcal{L} \times \mathcal{L}$ such that $\mathcal{T} \models R \equiv G \sqcap K$, and $K \sqcap O$ is satisfiable in $\mathcal{T}$. Then K is a *contraction* of R according to O and $\mathcal{T}$.
If nothing can be kept in R during the contraction process, we get the worst solution –from a matchmaking point of view– $\langle G, K \rangle = \langle R, \top \rangle$, that is give up everything of R. Conversely, if $R \sqcap O$ is satisfiable in $\mathcal{T}$, that is a potential match occurs, nothing has to be given up and the solution is $\langle \top, R \rangle$, *i.e.*, give up nothing. Hence, a Concept Contraction problem amounts to an extension of a Concept Satisfiability one. Since usually one wants to give up as few things as possible, some minimality criteria in the contraction must be defined [7]. If the offered resource O is a potential match for R, it is necessary to assess what should be hypothesized H in O in order to completely satisfy R and then move to a full match. In [8] the Concept Abduction problem was first defined as:
– **Concept Abduction**. Let $\mathcal{L}$ be a DL, R, O be two concepts in $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$, where both O and R are satisfiable w.r.t. $\mathcal{T}$. A *Concept Abduction Problem* (CAP), identified by $\langle \mathcal{L}, R, O, \mathcal{T} \rangle$, is to find a concept $H \in \mathcal{L}$ such that $\mathcal{T} \models O \sqcap H \sqsubseteq R$, and moreover $O \sqcap H$ is satisfiable in $\mathcal{T}$. We call H a *hypothesis* about O according to R and $\mathcal{T}$.

Obviously the definition refers to R and O consistent. If $O \sqsubseteq R$ then we have $H = \top$ as a solution to the related CAP. Hence, Concept Abduction amounts to extending subsumption. On the other hand, if $O \equiv \top$ then $H \sqsubseteq R$.

Referring to the matchmaking example of Section 2, let us consider a TBox $\mathcal{T}$ for the tourism domain containing the axioms (i) $Palace \sqsubseteq \exists hasAge \sqcap \exists hasRoof$ and (ii) $Palace \sqsubseteq \neg Church$. Then both request and all offers could be modeled in DL formalism, *e.g.*,
- $\mathbf{R} \equiv Palace \sqcap \forall hasAge.MiddleAge \sqcap \exists hasCourtyard$
- $\mathbf{O_{partial}} \equiv Church \sqcap \forall hasAge.MiddleAge \sqcap \exists hasCourtyard$
- $\mathbf{O_{potential}} \equiv Palace \sqcap \forall hasAge.MiddleAge \sqcap \forall hasRoof.FrescoedRoof$
Having two concepts whose conjunction is unsatisfiable, in the solution $\langle G, K \rangle$ to the CCP $\langle \mathcal{L}, R, O, \mathcal{T} \rangle$, G represents "why" R and O are not compatible.

For instance, the CCP $\langle \mathcal{L}, \mathbf{R}, \mathbf{O_{partial}}, \mathcal{T} \rangle$ yields the pair $\langle G = Palace, K = \exists hasCourtyard \sqcap \forall hasAge.MiddleAge \rangle$. For Concept Abduction, having R and O such that $O \not\sqsubseteq R$, the solution H to the CAP $\langle \mathcal{L}, R, O, \mathcal{T} \rangle$ represents "why" the subsumption relation does not hold. H can be interpreted as *what is specified in R and not in O*. For example, the solution for the CAP $\langle \mathcal{L}, \mathbf{R}, \mathbf{O_{potential}}, \mathcal{T} \rangle$ is $H = \exists hasCourtyard$.

3 System Outline

Previous inference services have been exploited within a prototypical mobile client for semantic-based service/resource discovery. It is aimed at employing the semantic matchmaking approach outlined above, with a location-based resource filtering for mobile and ubiquitous applications.

3.1 Design and development guidelines

Design and development of the proposed application were driven by the following guidelines, taking the objective of maximizing efficiency, effectiveness and usability.
a. Limited computing resources of the target platform must be carefully taken into account. In particular for novel semantic-based mobile applications, from a performance viewpoint it is practically infeasible to reuse existing Semantic Web tools and libraries on current mobile devices. A compact and optimized implementation of the required features and technologies is thus needed.
b. Mobile computing platforms are much more heterogeneous than personal computers, with devices highly differentiating in form factor, computational and communication capabilities and operating systems. Cross-platform runtime environments have to be developed in order to overcome this fragmentation.
c. Graphical User Interface (GUI) design should endorse the peculiarities of mobile and pervasive computing. Unlike their desktop counterparts, mobile applications are characterized by a *bursting* usage pattern, *i.e.*, with frequent and short sessions. Hence, mobile GUIs must be designed so that users can satisfy their needs in a quick and straightforward way. A task-oriented and consistent look and feel is required, leveraging familiar metaphors which most users are accustomed to.
d. Finally, software design must be conscious of the inherent constraints of mobile ad-hoc networks. From an application viewpoint, the most important issues are unpredictable disconnections and low data rates. The former is mainly due to host mobility, higher transmission error rates of wireless links and limited battery duration; the latter is a typical concern of wireless networks w.r.t. wired ones and it is also due to energy saving requirements for small devices. Applications must be designed with built-in resilience against network failures and QoS degradation, so as to prevent unexpected behaviors.

For a greater compatibility with various mobile platforms, our tool was developed using Java Micro Edition (ME) technology, by far the most widespread

cross-platform mobile environment. The compliance with one of the Java ME *profiles* ensures the compatibility with a broad class of mobile computing devices. The Java Mobile Information Device Profile[2] (MIDP) was selected, which is currently available for the majority of mobile phones and connected PDAs. Our tool is fully compliant with Java MIDP 2.0 specifications and API. All UI elements are accessible through the keyboard/keypad of the mobile device; additionally, MIDP transparently adapts to pointer-based interaction (*e.g.*, via touchscreen) on platforms where it is available.

In order to allow location-based service/resource provisioning, the application exploits the Java Location API JSR-179[3] to determine the device's location. JSR-179 provides a unified API to interact with all *location providers –i.e.*, real-time positioning technologies– that are available on the device. These include an internal GPS (Global Positioning System) receiver, an external GPS device connected *e.g.*, via Bluetooth and the mobile phone network (cell-based positioning). Accuracy depends on the positioning method, being typically higher for GPS than for cell-based techniques. Our tool requests a high-accuracy location determination firstly; if the accuracy requirement cannot be satisfied by available location providers on the device, the constraint is relaxed.

The proposed tool supports a subset of the DIG 1.1 interface (for its compactness DIG has been preferred to OWL) extended for *MAMAS-tng*[4] reasoner. This interface allows interaction with the state-of-the-art of Knowledge Representation Systems (KRS) through a classical request/response mechanism.

A lightweight implementation of the client-side DIG interface has been developed in Java. A specialized library was designed for efficient manipulation of knowledge bases. In order to minimize runtime memory consumption, *kXML*[5] Java streaming XML parser was adopted, which implements the open standard XML Pull API[6]. Streaming parsers allow an application to closely control the parsing process and do not build an in-memory syntax tree model for the XML document (as DOM parsers do). This increases speed and reduces memory requirements, which is highly desirable in resource-constrained environments. Moreover, streaming parsers are best choice for processing XML data incoming from network connections, since parsing can be pipelined with input reception.

3.2 Case study: a quick tour in Bari

In order to better explain the approach, we consider the following case study: *Bob is in Bari for business. After his last meeting, he is near the old town centre with some spare time before reaching a colleague for dinner. He had never been in Bari before and he knows very little about the city. Being interested in medieval art and particularly in churches, he would like to visit interesting places*

[2] http://java.sun.com/products/midp/
[3] http://jcp.org/en/jsr/detail?id=179
[4] http://sisinfab.poliba.it/MAMAS-tng/
[5] http://kxml.sourceforge.net/
[6] http://xmlpull.org/

Fig. 1. Ontology selection screen

Fig. 2. Ontology browsing screen

near his current location. Let us see how we model the discovery issue to meet Bob's needs.

Use your smartphone. *Bob is looking for medieval churches in a 1 Km range. Under GPRS/UMTS or Wi-Fi coverage, his GPS-enabled smartphone can connect to a service/resource provider in order to search for interesting items in the area.* The service provider keeps track of semantically annotated descriptions of different types of touristic points of interest in Apulia region together with their position coordinates. It interacts with mobile clients replying to their requests: in particular, it is equipped with a DL reasoner. In our current implementation we adopt the previously cited *MAMAS-tng*. Designed user interface is task-oriented. The provided mobile tool assists the user in service/resource discovery through the following three main tasks.

Ontology management. *Firstly, Bob selects cultural heritage as the resource category he is interested in.* Different domain ontologies are used to describe general resource classes (*e.g.*, accommodation, cultural heritage, movie/theatre shows). When the application is started up, the ontology selection screen is shown, as portrayed in Figure 1. A list of managed ontologies is displayed in **(a)**. When the user highlights an ontology, more detailed information is shown in the lower portion **(b)** of the screen. Each ontology is labeled by a universally unique identifier (OUUID) which is required to perform the ontology agreement between client and resource provider before the matchmaking phase [9]; besides, ancillary information such as ontology domain name, vendor and an icon can be displayed. As soon as it is started, the virtual guide begins device discovery in

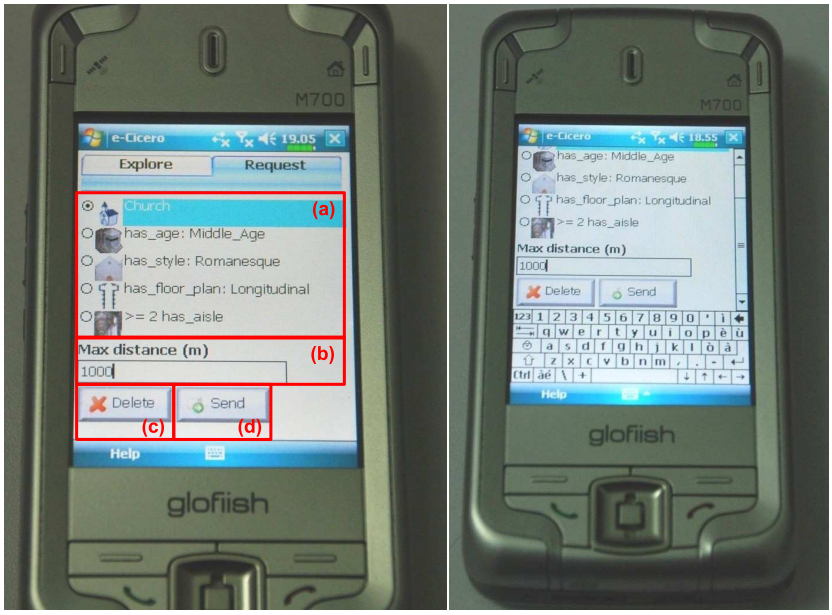

Fig. 3. Request confirmation screen

the background to find a suitable service provider.

Semantic request composition. *Bob composes his semantic-based request through a fully visual form. He browses resource features modeled in the domain ontology and selects desired characteristics, without actually seeing anything of the underlying DL-based formalism. Then he submits his request to the server.* Figure 2 shows the taxonomy browsing screen. A list-based browsing panel is displayed in **(b-c)**, adopting the same UI metaphors as file system navigation tools that are now familiar to PC users (*e.g.*, "My Computer" in Windows or "Nautilus" in GNOME). This avoids the complexity of typical tree or graph-based GUI representations of ontological knowledge, which can be cumbersome for exploring complex ontologies. In particular, the list **(c)** shows the current *focus* in the classification induced by terminological definitions and subsumptions in the reference ontology. The directional keys of the mobile device or the buttons in **(d)** are used to "surf" the ontology by expanding an item or going back one level. The panel focus changes accordingly. Navigation *storyboard* is displayed in **(b)**, so that the user can orient himself even in deeper ontologies. The user can select desired features (corresponding to ontology's concepts and role constructors) to build his request, through either the keypad, the appropriate button in **(d)** or commands in the menu **(e)**. Finally, the tabs in **(a)** allow the user to switch from the browsing screen to the request confirmation screen and vice versa.

The request screen is shown in Figure 3. Currently selected features are shown in the list **(a)**; if the user changes his mind, he can remove a previously

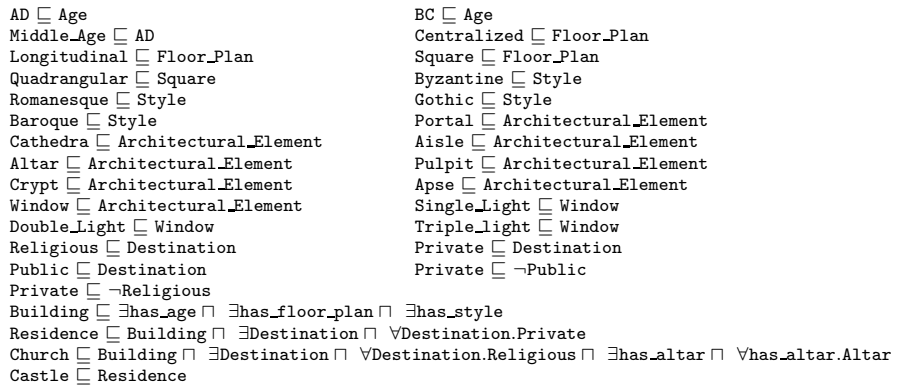

Fig. 4. Axioms in the toy cultural heritage ontology used in the case study

selected feature using the button **(c)**. Additionally, in **(b)** the user can specify a maximum distance D for resource retrieval. This range is submitted to the provider together with the current device coordinates –determined through the Location API. In this way, a circular area is identified, with its centre in the user's position: the service provider will return resources located in this area only. Eventually the user confirms his request and submits it to the resource provider through the appropriate command button **(d)**.

Bob would like to visit a Romanesque Middle Age church with longitudinal floor plan and two aisles. W.r.t. our cultural heritage reference ontology (reported in Figure 4 only partially for the sake of brevity), this can be expressed formally as

R: $Church \sqcap \forall\ has\_age.Middle\_Age \sqcap \forall\ has\_floor\_plan.Longitudinal \sqcap \geq 2\ has\_aisle \sqcap \forall\ has\_style.Romanesque$

in DL classical notation. As it can be seen, requests are formulated as DL conjunctive queries. Each conjunct represents a requested resource feature; it can be an atomic concept selected from the ontology, a universal quantifier or an unqualified number restriction on roles. If the domain ontology defines an atomic concept as range for a particular role, the application moves the browsing panel focus to that concept when the role is expanded. The user can then select that concept or one of its subclasses as filler for the universal quantifier constructor. Otherwise, if no range is specified for a role, a number restriction constructor is assumed and the user is allowed to specify the cardinality constraint type and value (as in the case of *has_aisle*). The communications module was designed as a finite state machine to precisely retain knowledge about the progress of client-server interaction. By doing so, only failed operations are actually repeated, thus improving efficiency from both time and energy standpoints. Retry command, along with other less frequently used functions, is located in the menu so as not to cram the main screen area.

Results review and query refinement. *The reasoner filters all available resources, discarding the ones not located in the user-specified area. Then it per-*

forms the matchmaking between user request and each resource description. Results are ranked according to their semantic distance from the request and best matching resources are returned to Bob, along with their location. For example, let us consider the following resources in the knowledge base of the provider:
S1: Basilica of St. Nicholas, Bari (distance from user: $d = 900$ m). A Romanesque Middle Age church, with longitudinal floor plant, two aisles, three portals and two towers. Other notable elements are its, crypt, altar, cathedra and Baroque ceiling. W.r.t. domain ontology, this is expressed as:
$Church \sqcap = 2\ has\_aisle \sqcap \forall\ has\_age.Middle\_Age \sqcap \forall\ has\_style.Romanesque \sqcap = 1\ has\_apse \sqcap = 3\ has\_portal \sqcap = 1\ has\_crypt \sqcap = 1\ has\_altar \sqcap = 2\ has\_tower \sqcap = 1\ has\_cathedra \sqcap \exists\ ceiling\_style \sqcap \forall\ ceiling\_style.Baroque \sqcap \forall\ has\_floor\_plan.Longitudinal$
S2: Norman-Hohenstaufen Castle, Bari ($d = 570$ m). It is described as a Middle Age castle, with Byzantine architectural style and a quadrangular plan with four towers.
$Castle \sqcap \forall\ has\_floor\_plan.Quadrangular \sqcap = 4\ has\_tower \sqcap \forall\ has\_style.Byzantine \sqcap \forall\ has\_age.Middle\_Age$
S3: Church of St. Scholastica ($d = 1100$ m). It is described as a Romanesque Middle Age church, with longitudinal floor plan, three aisles, an apse and a tower.
$Church \sqcap \forall\ has\_style.Romanesque \sqcap \forall\ has\_age.Middle\_Age \sqcap \forall\ has\_floor\_plan.Longitudinal \sqcap = 3\ has\_aisle \sqcap = 1\ has\_tower \sqcap = 1\ has\_apse$
S4: Church of St. Mark of the Venetians, Bari ($d = 430$ m). It is described as a Romanesque Middle Age church with two single-light windows and a tower.
$Church \sqcap \forall\ has\_style.Romanesque \sqcap \forall\ has\_window.Single\_Light \sqcap = 2\ has\_window \sqcap \forall\ has\_age.Middle\_Age \sqcap = 1\ has\_tower$

In order to make resource discovery really useful and effective, the user should receive some meaningful feedback about the outcome of his request. A simple list of results is then inadequate whereas a ranking score has more explanatory power. Moreover, looking at descriptions of retrieved resources the user might find other features he was not previously aware of, but he would be interested in. Table 1 reports results of the semantic-based matchmaking for the above example. **S3** is not processed since its distance from the user exceeds the limit, even though it would result in a full match. **S1** is a full match with the request, because it explicitly satisfies all user requirements. On the other hand, **S4** is described just as Romanesque Middle Age church, therefore due to Open World Assumption (what is not specified in a description has not to be interpreted as a constraint of absence) it is not specified whether it has a longitudinal floor plan with aisles or not: these characteristics become part of the *Hypothesis* computed through Concept Abduction. Finally, **S2** produces a partial match with user request, since it is a castle: this concept is incompatible with user request, so it forms the *Give Up* feature computed through Concept Contraction, while the *Keep* expression includes the remaining of the request. For each advertised resource O_i, its overall score S is then computed using the *utility function*:

$$S(O_i, d_i) = 100[1 - \frac{s\_match(R, O_i)}{max(s\_match)}(1 + \frac{d_i}{D})]$$

Fig. 5. Results screen

Fig. 6. Result details screen

where $s\_match(R, O_i)$ is the semantic distance from request to resource; $max(s\_match) \doteq s\_match(R, \top)$ is the maximum semantic distance from the request, which depends on axioms in the domain ontology; d_i is the physical distance of the resource from the user and D is the user-supplied distance limit. The purposes of the utility function are to weigh the result of semantic matching according to distance and to convert the score to a more user-friendly scale. Of course nearer resources are preferred, but in case of a full match $s\_match(R, O_i) = 0$ hence $S = 100$ regardless of distance, so that full matches will always be shown at the top of the result list.

The result screen is showed in Figure 5. Retrieved resources are listed –best matching first– along with their overall scores as pie charts. When the user selects a resource, a detail form screen is displayed, as depicted in Figure 6. A picture of the resource is shown in **(a)** together with its address and the distance from the user, while the semantically relevant resource properties contributing to the outcome are displayed in **(b)**.

If Bob is not satisfied with results, he can refine his semantic request and submit it again. The user can go back to the ontology browsing screen to modify his request. Furthermore, in case of partial match, he can select some elements of the *Give Up* list in the result details screen and they will be removed from the request automatically.

Supply	Match type	s_match	Outcome	Score
S1: Basilica of St. Nicholas	Full	0	**Hypothesis:** $H = \top$;	100
S4: Church of St. Mark	Potential	3	**Hypothesis:** $\forall\ has\_floor\_plan.Longitudinal \sqcap \geq 2\ has\_aisle$	92
S2: Norman-Hohenstaufen Castle	Partial	11	**Give up:** $Church$ **Keep:** $Building \sqcap \forall\ has\_age.Middle\_Age$ **Hypothesis:** $\forall\ has\_floor\_plan.Longitudinal \sqcap \geq 2\ has\_aisle \sqcap \forall\ has\_style.Romanesque$	68
S3: Cathedral of Trani	N.A.	N.A.	discarded due to distance	N.A.

Table 1. Matchmaking results

4 Related Work

Service/resource discovery in mobile and ubiquitous computing is a very active topic in both the research community and ICT industry. The main challenge is to provide effective and flexible discovery paradigms and techniques, yet intuitive enough to be of practical interest for a potentially wide user base. Ideas and technologies from the Semantic Web vision are attracting growing attention as an enabling element.

In [10] a prototypical mobile client is presented for semantic-based mobile service discovery, in which user preference specification plays a key role. An OWL ontology describes relevant properties of each service class. Ontology browsing is allowed through a graph-based representation, which is rendered adaptively. A pointing device and a large screen seem to be required in order to explore ontologies of moderate complexity with reasonable comfort. Moreover, preference specification requires a rather long interaction process, which can be impractical in mobile scenarios. Authors acknowledged this issue and introduced heuristic mechanisms to simplify interaction, *e.g.*, the exploitation of default values.

mSpace Mobile [11] is a location- and context-aware mobile Semantic Web client for tourism scenarios. Users can discover and combine services/resources in an area through an associative process, by browsing semantic relationships represented in an RDF knowledge base. Similarly to our ontology browsing, a "focus + context" paradigm is adopted in UI design. The desire to integrate multiple domains of information has led to a division of the user interface into many tiny sections, whose clarity and practical usability seem questionable. Moreover, knowledge is extracted from several independent sources to build a central RDF triple store, which can then be accessed by one or more servers in order to reply to client requests through the Internet. The proposed architecture is therefore hardly adaptable to MANET-based ubiquitous computing environments.

Fully decentralized semantic-based resource/service discovery infrastructures in ubiquitous computing call for peer-to-peer interaction paradigms. Hence, mobile hosts themselves should be endowed with reasoning and matchmaking capabilities. To the best of our knowledge, Pocket KRHyper [12] is currently the only

available reasoning engine for mobile devices. It provides satisfiability and subsumption inference services, which have been exploited by authors in a DL-based matchmaking scheme between user profiles and descriptions of resources/services [13]. Due to resource constraints, the proposed approach privileges simplicity over expressiveness and flexibility w.r.t. ours, and does not currently allow explicit explanation of outcomes. We conjecture that a native language optimized implementation can provide acceptable performance for more resource-intensive inference procedures, thus allowing to overcome those constraints at least partially.

5 Conclusion and Future Work

We have presented a fully visual mobile application for semantic-enabled resource discovery in ubiquitous computing. Non standard DL inference services are exploited to perform a matchmaking based on semantics of request/resource descriptions. The proposed tool is general purpose as it allows a mobile user to perform a discovery in multiple scenarios (from u-commerce [14] to u-tourism [1]) simply using her mobile phone. The final goal of our endeavor is to provide an m-DSS to retrieve resources/services through a fully dynamic wireless infrastructure, without relying on support facilities provided by more expensive information systems and wired networks. The approach has been motivated in a tourism case study. The proposed application aims to be really context-aware as it recognizes via GPS the user location and grades matchmaking outcomes according to vicinity criteria. Limits of the tool emerge because actually the time spent in performing the overall discovery and ranking procedure is still somewhat high to be definitively acceptable. From this point of view a significant bottleneck is represented by the matchmaker computation. The complexity of the matchmaker module claims for serious optimization and simplification of the reasoner features in sight of an explicit utilization for pervasive and ubiquitous applications. Future work will also focus on the enhancements of the user interface. A more immediate GUI also integrating a speech recognition engine is the most important effort we are now pursuing.

Acknowledgments

The authors are grateful to Saverio Ieva for useful implementations. They would also like to acknowledge the support of Apulia project PE_082 “IC Technologies for tourist assistance through interactive consultancy of a virtual guide”.

References

1. Watson, R., Akselsen, S., Monod, E., Pitt, L.: The Open Tourism Consortium: Laying The Foundations for the Future of Tourism. European Management Journal **22**(3) (2004) 315–326

2. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic Matching of Web Services Capabilities. In: The Semantic Web - ISWC 2002. Number 2342. (2002) 333–347
3. Li, L., Horrocks, I.: A Software Framework for Matchmaking Based on Semantic Web Technology. International Journal of Electronic Commerce **8**(4) (2004) 39–60
4. Bechhofer, S., Möller, R., Crowther, P.: The DIG Description Logic Interface. In: Proceedings of the 16th International Workshop on Description Logics (DL'03). Volume 81 of CEUR Workshop Proceedings. (September 2003)
5. Baader, F., Calvanese, D., Mc Guinness, D., Nardi, D., Patel-Schneider, P.: The Description Logic Handbook. Cambridge University Press (2002)
6. Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F.M., Mongiello, M.: Concept abduction and contraction in description logics. In: Proceedings of the 16th International Workshop on Description Logics (DL'03). Volume 81 of CEUR Workshop Proceedings. (Sept. 2003)
7. Gärdenfors, P.: Knowledge in Flux: Modeling the Dynamics of Epistemic States. Bradford Books, MIT Press, Cambridge, MA (1988)
8. Di Noia, T., Di Sciascio, E., Donini, F.M., Mongiello, M.: Abductive matchmaking using description logics. In: Proceedings of Eighteenth International Joint Conference on Artificial Intelligence IJCAI-03, Acapulco, Mexico, MK (Aug. 2003) 337–342
9. Ruta, M., Di Noia, T., Di Sciascio, E., Donini, F.: Semantic based collaborative p2p in ubiquitous computing. Web Intelligence and Agent Systems **5**(4) (2007) 375–391
10. Noppens, O., Luther, M., Liebig, T., Wagner, M., Paolucci, M.: Ontology-supported Preference Handling for Mobile Music Selection. In: Proceedings of the Multidisciplinary Workshop on Advances in Preference Handling, Riva del Garda, Italy (August 2006)
11. Wilson, M., Russell, A., Smith, D., Owens, A., Schraefel, M.: mSpace Mobile: A Mobile Application for the Semantic Web. In: Proceedings of the End User Semantic Web Workshop at ISWC 2005. (November 2005)
12. Sinner, A., Kleemann, T.: KRHyper - In Your Pocket. In: Proc. of 20th International Conference on Automated Deduction (CADE-20), Tallinn, Estonia (July 2005) 452–457
13. Kleemann, T., Sinner, A.: User Profiles and Matchmaking on Mobile Phones. In Bartenstein, O., ed.: Proc. of 16th International Conference on Applications of Declarative Programming and Knowledge Management INAP2005, Fukuoka. (2005)
14. Watson, R., Pitt, L., Berthon, P., Zinkhan, G.: U-Commerce: Expanding the Universe of Marketing. Journal of the Academy of Marketing Science **30**(4) (2002) 333–347