SEMANTIC-BASED BLUETOOTH-RFID INTERACTION FOR ADVANCED RESOURCE DISCOVERY IN PERVASIVE CONTEXTS[1]

Tommaso Di Noia, Eugenio Di Sciascio, Francesco M. Donini*, Michele Ruta, Floriano Scioscia, Eufemia Tinelli

SisInfLab -- Politecnico di Bari, Bari, Italy

{t.dinoia, disciascio, m.ruta, f.scioscia, e.tinelli}@poliba.it

*SisInfLab and Università della Tuscia, Viterbo, Italy

donini@unitus.it

Keywords

Knowledge Discovery, Semantic matchmaking, Ontologies, Mobile Technologies, Wireless Technologies, Knowledge Storage, Knowledge-Based Systems

ABSTRACT

We propose a novel object discovery framework integrating the application layer of Bluetooth and RFID standards. The approach is motivated and illustrated in an innovative u-commerce setting. Given a request, it allows an advanced discovery process, exploiting semantically annotated descriptions of goods available in the u-marketplace. The RFID data exchange protocol and the Bluetooth Service Discovery Protocol have been modified and enhanced, to enable support for such semantic annotation of products. Modifications to the standards have

---

[1] This is a draft version of the paper appearing in: International Journal on Semantic Web and Information Systems (IJSWIS), Volume 4, Number 1, page 50--74 - 2008

been conceived to be backward compatible, thus allowing the smooth coexistence of the legacy discovery and/or identification features. Also noteworthy is the introduction of a dedicated compression tool to reduce storage/transmission problems due to the verbosity of XML-based semantic languages.

INTRODUCTION AND MOTIVATION

Radio-Frequency IDentification (RFID) is an increasingly widespread and promising wireless technology interconnecting via radio a transponder carrying data (*tag*) located on an object, and an interrogator (*reader*) able to receive the transmitted data. Tags usually contain a unique identification code, which can be used by readers to identify the associated object. Since low-cost tags can be fastened to objects unobtrusively, preserving their common functions, RFID *de facto* increases the “pervasiveness” of a computing environment. Current RFID applications focus on retrieving relevant attributes of the object the tag is clung to, via a networked infrastructure from a fixed information server. This identification process involves the code associated to the transponder exploited as index key. Nowadays tags with larger memory capacity and on-board sensors enable new scenarios and further applications, not yet explored. We believe that, in the era of semantic technologies and mobile computing, there is room for more advanced and significant applications of RFIDs extended with structured descriptions, so that a good equipped with an RFID can semantically describe itself along its whole life-cycle. We therefore conceived a unified framework where a semantic-enhanced RFID-based infrastructure and an advanced Bluetooth service discovery –also endowed of semantic-based discovery features-- are virtually “interconnected” at the application layer permitting innovative services in u-environments. In our mobile framework, tagged objects expose to a reader not simply a string code but a semantically annotated description. Such objects may hence describe themselves in a variety of scenarios (*e.g.*, during supply chain management, shipment, storing, sale and post-sale), without depending on a centralized database. Exploiting these annotations calls for discovery/interaction protocols able to effectively deal with rich and articulated descriptions. Therefore a novel multi-protocol and interactive discovery mechanism has been designed. In this effort we borrowed from ideas and technologies devised for the Semantic Web initiative. To simply illustrate our proposal, we set our stage in a *u-marketplace* context [i], where objects endowed with RFID tags are dipped into an enhanced Bluetooth framework.

In particular, building on previous works that enhanced the basic discovery features of Bluetooth with semantic-based discovery capabilities (Ruta et al., 2006a), we propose an extension of EPCglobal specifications for RFID tag data standards, providing semantic-based value-added services. Coping with limited storage and computational capabilities of mobile and embedded devices, and with reduced bandwidth provided by wireless links, issues related to the verbosity of semantic annotation languages cannot be neglected. Compression techniques become essential to enable storage and transmission of semantically annotated information on mobile devices. We hence devised and exploited a novel efficient XML compression algorithm, specifically targeted for DIG 1.1 (Bechhofer et al., 2003) document instances. Benefits of compression apply to the whole ubiquitous computing environment, as decreasing data size means shorter communication delays, efficient usage of bandwidth and reduced battery drain for mobile devices in a Mobile Ad-hoc NETwork (MANET).

The remaining of the paper is structured as follows. In the next section relevant technological bricks of the proposed framework are surveyed. Section 3 outlines the framework, explaining the discovery process as well as proposed semantic-based enhancements to RFID standards. The compression algorithm for semantic annotations is outlined in Section 4. Section 5 exemplifies the approach in a u-commerce scenario. Results on key performance measures to assess the feasibility of the proposed approach, are provided in Section 6. Conclusions close the paper.

BASICS

In this section we survey relevant aspects of languages, technologies and protocols we use and adapt, concentrating on key features our proposal is based on. We assume the reader be familiar with at least basic elements of Semantic Web and ontologies (Berners-Lee et al., 2001; Shadbolt et al., 2006; Horrocks et al., 2001; McGuinness et al., 2002; Martin et al. 2002), of OWL (http://www.w3.org/TR/owl-features/) and related languages, such as Description Logics (DLs) (Borgida, 1995; Donini et al., 1996). We therefore move straightforwardly to analyze issues closely related to our proposal.

Exploiting semantically annotated descriptions

Given a domain ontology $\mathcal{T}$, DL-based systems usually provide at least two basic reasoning services: *Concept Satisfiability* and *Concept Subsumption*. Using subsumption it is possible to establish if a description C is more specific than a description D, $\mathcal{T} \models C \sqsubseteq D$. If the previous relation holds, then we may say that information C associated to a given resource completely satisfies what has been requested in D, *i.e.* a *full match* occurs. With Concept Satisfiability the discovery of incompatible resources with respect to a request can be performed. If $D \sqcap C$ is not satisfiable w.r.t. the ontology $\mathcal{T}$, then C is not compatible with the request. Obviously *full matches* cannot be deemed the only useful, as they will be probably rare in a variety of contexts.

Given a request and a set of resources, usually $C \not\sqsubseteq D$ and $D \sqcap C$ is satisfiable w.r.t. $\mathcal{T}$. That is, the resource does not completely satisfy the request but it is compatible with it. Hence, a metric is needed to establish "how much" the resource *C* is compatible with the request *D* or, equivalently, "how much" it is not specified in *C* to completely satisfy *D*, in order to make the subsumption relation $C \sqsubseteq D$ true. In (Di Noia et al., 2004) *rankPotential* algorithm was proposed to evaluate this measure. Given an $\mathcal{ALN}$ (Attributive Language with Number restrictions) ontology $\mathcal{T}$ and two $\mathcal{ALN}$ concepts *C* and *D* both satisfiable in $\mathcal{T}$, *rankPotential*(*C*, *D*, $\mathcal{T}$) computes a *semantic distance* of *C* from *D* with respect to the ontology $\mathcal{T}$.

If some requirements in the request *D* are in conflict with the resource *C*, *rankPotential* cannot be applied. Nevertheless, in looking for "not so much" unsatisfactory matches when recovering from an initial "no match", a partial match could still be useful. In (Di Noia et al., 2004) the *rankPartial* algorithm was proposed for ranking incoherent pairs of descriptions. Given an ontology $\mathcal{T}$ and two concept expressions *D* and *C*, both satisfiable with respect to $\mathcal{T}$, if *D* is not compatible with *C* *i.e.* their conjunction is not satisfiable with respect to $\mathcal{T}$, then *rankPartial* returns a score measuring the semantic incompatibility of *D* and *C*.

Semantic based Bluetooth Service Discovery

Usually, resource discovery protocols involve a requester, a lookup or directory server and finally a resource provider. As a MANET is a volatile environment, a flexible resource discovery paradigm is needed to overcome difficulties due to the host mobility. Nevertheless, existing protocols for mobile applications use a simple string-matching, which is largely inefficient in advanced scenarios (Ruta et al., 2006b). With specific reference to the Bluetooth service discovery protocol (SDP), it is based on a 128 bit Universally Unique Identifier (UUID) associated to single service classes. Resource matching in Bluetooth is hence strictly syntactic, and SDP manages only exact matches. In (Ruta et al., 2006a) a framework has been proposed that allows the management of both syntactic and semantic discovery of resources, by integrating a semantic layer within the OSI Bluetooth stack at application level. The Bluetooth standard has been enriched by new functionalities which permitted to maintain a backward compatibility (handheld device connectivity), adding the support to discovery of semantically annotated resources. Unused classes of 128 bit UUIDs in the original Bluetooth standard were exploited to mark each specific ontology thus calling this identifier *OUUID* (Ontology Universally Unique IDentifier). By means of the OUUID matching the context was identified and a preliminary selection of resource referring to the same request's ontology was performed. The fundamental assumption is that each resource is semantically annotated. A service provider stores annotations within resource records, labelled with unique 32-bit identifiers. Each record contains general information about a single semantic enabled resource and it entirely consists of a list of resource attributes. In addition to the OUUID attribute, there are a *ResourceName* (a human-readable name for the resource), a *ResourceDescription* (expressed using DIG syntax) and a variable number of *ResourceUtilityAttr i* attributes, *i.e.*, numerical values used according to specific applications. In (Ruta et al., 2006a), by adding four SDP Protocol Data Units (PDUs) *SDP_OntologySearch* (request and response) and *SDP_SemanticServiceSearch* (request and response) to the original standard (exploiting not used PDU ID), together with the original SDP capabilities, further semantic enabled discovery functionalities were introduced. The overall interaction was based on the original SDP in Bluetooth. No modifications were made to the original structure of transactions. In fact, semantic-based micro-layer has been built over the standard SDP recycling its basic parameters, data structures and functions, just differently using the basic framework.

RFID features

In our framework we refer to RFID transponders compliant with EPCglobal standard for Class 1-Generation 2 UHF tags (Traub et al., 2005). Tag memory is divided in four logical banks (EPCglobal Inc., 2005a): **(1)** ***Reserved***. It is optional; if present, it stores 32-bit kill and access passwords. **(2)** ***Electronic Product Code (EPC)***. It stores, starting from address 0: (i) 16 bits for a Cyclic Redundancy Check (CRC) code; (ii) a 16-bit Protocol Control (PC) field, composed of 5 bits for identification code length, 2 bits reserved for future use and 9 bits of numbering system identification; (iii) an EPC field for the identification code. **(3)** ***Tag identification (TID)***. It stores at least tag manufacturer and model identification codes. This bank may be enlarged to store other manufacturer or model-specific data (*e.g.* a tag serial number). **(4)** ***User***. An optional bank that stores data defined by the user application. Memory organization is user-defined. EPCglobal air interface protocol is an *Interrogator-Talks-First* (ITF) protocol: tags only reply to reader commands. Here we briefly outline basic protocol features.

An RFID reader can preselect a subset of the tag population currently in range, according to user-defined criteria, by means of a sequence of *Select* commands.

Select command sends a bit string to all tags in range. Each tag will compare it with the content of a memory area specified by the reader, then it will assert/deassert one of its status flags according to the comparison result (match/no-match). Command structure is shown in Table 1; parameters are as follows: (i) *Target* determines which tag status flag will be modified by the

Select command; (ii) *Action* tells how a tag is required to modify the flag (assert, deassert, do nothing) for either positive or negative match outcome (a three-bit field is thus required to encode the six cases); (iii) *MemBank* indicates what memory bank must be compared; (iv) *Pointer* is the address of the first bit of MemBank tag memory area that must be compared; (v) *Length* is the length of the bit string to be compared; (vi) *Mask* is the bit string to be compared with the content of the memory area selected by MemBank, Pointer and Length values; (vii) *Truncate* tells the tag to send only part of its EPC code in the following protocol step; (viii) *CRC,* used for command data integrity protection.

Opcode	**Target**	**Action**	**MemBank**	**Pointer**	**Length**	**Mask**	**Truncate**	**CRC**
1010_2	3 bits	3 bits	2 bits	bit vector	8 bits	1-255 bits	1 bit	16 bits

Table 1. Select command structure in RFID protocol

After this phase, the inventory loop begins. In each iteration the reader isolates one tag in range, reads its EPC code and can access its memory contents. Among available commands, only *Read* and *Write* are relevant for our purposes.

Read command allows to read from one of the four tag memory banks. Command structure is shown in Table 2; parameters are as follows: (i) *MemBank* indicates the bank data must be read from; (ii) *WordPtr* points to the first 16-bit memory word to be read; (iii) *WordCount* is the number of consecutive 16-bit memory words that must be read (if it is 0, then the tag will send data stored up to the end of the memory bank); (iv) *RN*, random number used as access transaction identifier between reader and tag; (v) *CRC*.

Write command allows a reader to write a 16-bit word to one of the four tag memory banks. Command structure is similar to *Read*, as shown in Table 3.

Opcode	**MemBank**	**WordPtr**	**WordCount**	**RN**	**CRC**
11000010_2	2 bits	bit vector	8 bits	16 bits	16 bits

Table 2. Read command structure in RFID protocol

Opcode	MemBank	WordPtr	Data	RN	CRC
11000011_2	2 bits	bit vector	16 bits	16 bits	16 bits

Table 3. Write command structure in RFID protocol

Together with tag data and air interface protocol, the EPCglobal standard defines a support infrastructure for RFID applications, where a key role is played by *Object Naming Service (ONS)* (EPCglobal Inc., 2005b). It is based on the Domain Name System adopted to solve symbolic Internet addresses. ONS allows to retrieve services related to a specific object using the EPC code stored within the tag as a URI. *EPCglobal Network Protocol Parameter Registry* is maintained by EPCglobal consortium and contains suffixes identifying all valid service types (*e.g., ws* for a Web Service, *html* for a Web Page of the manufacturer, *epcis* for a EPCglobal Information Service providing authoritative information about the object associated with an EPC code).

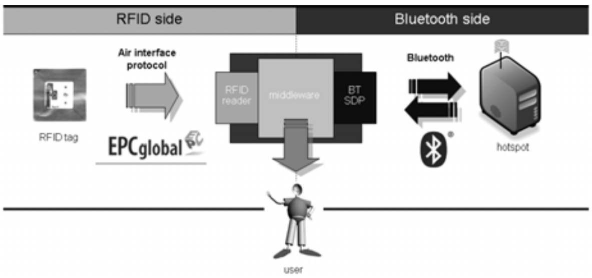


Figure 1. Infrastructure elements: semantic-enhanced RFID tags; air-interface EPCglobal RFID protocol; middleware stratum; Bluetooth SD protocol, hotspot enriched with semantic matchmaking capabilities.

FRAMEWORK AND APPROACH

We designed a unified semantic-aware framework, comprising modified RFID and Bluetooth based infrastructures that are virtually "interconnected" at the application layer permitting innovative services in u-environments. Our framework introduces a proposed extension of

EPCglobal standard, allowing a semantic-based object discovery. Protocols to read/write tags have been preserved maintaining original code-based access (so keeping a compatibility with legacy applications practically without modifications). A good can be easily and thoroughly described by means of a semantic annotated description stored within the tag it is associated with. Main elements of the proposed framework, see Figure 1, are: 1) goods equipped with semantic-enhanced RFID tags, 2) a middle tier component provided with an RFID reader and Bluetooth connectivity, 3) hotspot enriched with semantic matchmaking capabilities. Two identification/discovery paradigms are involved: EPCglobal air interface protocol for RFID tags and semantic-enhanced Bluetooth Service Discovery Protocol. Interaction can be triggered by the user by means of either an implicit or an explicit request. The simplest --though not trivial, as obviously requests may change over time and during the product life-cycle-- form of interaction is querying the tag (of the good) for some information, exploiting user's mobile handheld device. In implicit requests the framework can be used to recognize choices she performed so intercepting and interpreting them as a preliminary interaction aimed at discovery of goods similar or to be combined with the chosen one. In the first case the user can directly interact with the hotspot, issuing requests to it via the semantic-enhanced Bluetooth SDP and waiting for replies. In the latter one the user plays a more passive role as the "Environment" (in the sense of a pervasive and intelligent context, a marketplace in our example scenario) is able to perceive modifications w.r.t. an earlier situation. RFID tags are required for hosting product features and to set a link between the *real* and the *digital* world, whereas the middle tier is a double-faced component. It listens for descriptions directly coming from the objects (by reading the tag memory content), issues requests to the service provider and finally records and displays results to the user. The RFID reader, scanning characteristics of a selected product, enables the further discovery phase which is aimed at identifying resources similar to the chosen one or to be combined with it. Via the semantic based Bluetooth SDP and exploiting non standard inference services outlined above, best matching resources of the marketplace will be discovered and returned to the user. Hence the middleware integrates RFID and Bluetooth environments at the application layer: data coming from RFID tags are extracted, processed and reformatted. Furthermore they are arranged to enable the interaction with the service provider (*hotspot*) via the semantic-enhanced Bluetooth SDP. The hotspot keeps track of resources within the marketplace and replies to a submitted request with the best matching products for similarity and association. To this aim, it is equipped with a DL reasoner able to provide previously introduced services. Such an approach may provide several benefits. Information about a product is structured and complete; it accurately follows the product history within the supply chain, being progressively built or updated during the good life cycle. This improves traceability of production and distribution, facilitates sales and post-sale services thanks to an advanced and selective discovery infrastructure.

Semantic-enhanced EPCglobal RFID standard

In this subsection we outline the proposed backward-compatible extensions to EPCglobal RFID standards enabling the framework described above. It is noteworthy that our semantic enabled descriptions are expressed in DIG formalism (Bechhofer et al., 2003), a more compact syntactic variant of OWL.

Two reserved bits in the EPC area within each tag memory are exploited. The first one – at 15_h (10101_2) address – is exploited to indicate if the tag has a user memory (bit set) or not (bit cleared). The next one – at 16_h address – is asserted to mark semantic enabled tags. In this manner, by means of a *Select* command (see Table 4), a reader can easily distinguish semantic based tags. In particular Target and Action parameters have the effect to assert the SL tag status flag only for semantic-enabled tags and deassert it for remaining ones. The following inventory step will skip tags having SL flag deasserted, thus allowing a reader to identify only semantic-enabled tags (protocol commands belonging to the inventory step have not been described, because they are used in the standard fashion).

Parameter	Target	Action	MemBank	Pointer	Length	Mask
Value	100_2	000_2	01_2	00010101_2	00000010_2	11_2
Description	SL flag	assert in case of match, deassert otherwise	EPC memory bank	initial address	number of bits to compare	bit mask

Table 4. Select command parameters to detect semantic enabled tags

The EPC standard for UHF-Class 1 tags impose the content of TID memory up to $1F_h$ bit is fixed. As said above, optional information could be stored in additional TID memory. We use the TID memory area starting from 100000_2 address. There we store the identifier of the ontology (OUUID) w.r.t. the description contained within the tag is expressed. In order to make RFID systems compliant with the ontology support system proposed in (Ruta et al., 2006a), we define a bidirectional correspondence of OUUIDs stored in RFID transponders with those managed by Bluetooth devices. To retrieve the OUUID value stored within a tag, a reader will exploit a *Read* command with parameters as in Table 5:

Parameter	MemBank	WordPtr	WordCount
Value	10_2	000000010_2	00001000_2
Description	TID memory bank	initial address	read up to 8 words (128 bits)

Table 5. Read command parameters to extract OUUID from TID memory bank

Within the user memory bank together with the semantically annotated description of the good the tag is clung to (opportunely compressed) will be stored also contextual parameters (whose meaning depends on the specific application).

The extraction or the storing of a description within a tag can be performed by a reader through one or more Read or Write commands, respectively. Both commands are used in compliance with the standard air interface protocol. In Table 6, parameters of the Read command for extracting a compressed description are reported.

Parameter	MemBank	WordPtr	WordCount
Value	11_2	000000000_2	00000000_2
Description	User memory bank	initial address	read up to the end

Table 6. Read command parameters to extract semantic annotations from the User bank

In our approach the ONS mechanism is considered as a supplementary system able to grant the ontology support. In case the reader does not manage the ontology the description within the tag refers to, it may need an Internet connection in order to retrieve the related DIG file, which will then remain stored for further usage on other goods of the same category. For this purpose we use the ONS service and we hypothesize to register within the *EPCglobal Network Protocol Parameter Registry* a new service suffix, the *dig* one, that will contain the URL of the DIG file ontology. Of course the same can be done for OWL.

In case of EPC code families derived from the GS1 standard (formerly EAN.UCC) for barcode product identification, we assume that the pair of fields used for ONS requests – which refer to the manufacturer and to the merchandise class of the good – will correspond to a specific ontology. In fact that pair exactly identifies the product category. Two goods with the same value for that field parameter will be surely homogeneous or even equal. Note that the vice versa is not verified, but this is not a concern for our purposes because ONS searches proceed only from the EPC code toward the ontology. Hence we can surely have an unambiguous correspondence.

Deploying the approach

In our case study framework, we hypothesize a “smart shopping cart” is equipped with a sensor and a tablet computer, which integrates an RFID reader and Bluetooth connectivity. When a customer picks up a product, the system assists her in discovering additional items, either similar or to be combined with the selected one. To this aim, a two-step discovery is performed, exploiting two different but related ontologies. In the first step, *rankPotential* algorithm is exploited to retrieve correspondences with the request. Resources analogous to the one selected by the user are identified, but – at the same time – semantically incompatible goods are recognized. Their descriptions are submitted to the second matchmaking step. It exploits *rankPartial* over a differently modeled ontology so allowing to discover products to be associated with the chosen one. The hotspot will return two different lists of resource records respectively for objects in a potential correspondence with the request and in a partial one.

In advanced mobile scenarios, usually the match between a request and a provided resource involves not only the description of the resource itself but also data-oriented contextual properties. In fact, it would be quite strange to have a mobile commerce application without taking into account for example price or delivery time, among others. Hence, the overall match value should depend not only on the semantic distance between the description of the demand and of the resource, but also on those subsidiary values. An overall *utility function* has to combine them with semantic matchmaking results, in order to give a concrete match measure (Ruta et al., 2006b). In the proposed case study – referred to a u-commerce electronic product store – the utility function adopts three contextual parameters: price (in US dollars), estimated delivery time (in days), and product category, as shown in Table 7. They are exploited in a post-processing phase following the semantic-based matchmaking and aimed to better agree discovery results with user needs.

The proposed utility function (whose formulation derives from common sense considerations) has two expressions, for potential and partial matches respectively:

$$f_{POT}(\cdot) = \frac{pot\_match}{2} + \frac{\tanh(\frac{t_R - t_O}{\beta})}{3} u(t_R - t_O) + (\frac{p_O}{p_R} - 0.5)\frac{(1+\alpha)p_R - p_O}{3(1+\alpha)p_R}$$

$$f_{PAR}(\cdot) = \frac{par\_match}{2} + \frac{\tanh(\frac{t_R - t_O}{\beta})}{6} u(t_R - t_O) + \frac{1 - \gamma|c_R - c_O|}{3(2 + |c_R - c_O|)}$$

where *pot_match* and *par_match* are the potential and partial match values, *p* is price, *t* is delivery time and *c* is product category. The index *R* is referred to the request whereas the *O* one is referred to the supply and $u(\cdot)$ is Heaviside step function. Parameters α, β, γ can be used to fine-tune the utility function. Values we experimentally experienced with good results are $\alpha = 0.1$, $\beta = 10$, $\gamma = 0.2$. They have been determined by means of empirical tests through the comparison of system results with human users judgement. The higher the utility value the better the obtained match. In both formulas the leading term is represented by the semantic match.

Product category	phones	computers	photo	audio/video	hobbies
Value	1	2	3	4	5

Table 7. Product category contextual parameter

The second term depends on the estimated delivery time and it is differently weighted in proposed formulas. In the first one (discovery of goods similar to the request) a late delivery is more penalized. On the other hand, partial matches refer to items that can be used together with the selected one (such as accessories or complements), therefore a delay is less of a concern. The last term is different in the two formulas. For potential matches, it is related to product price. The price imposed by the requester is increased with a factor α on the assumption that, usually, the demander is willing to pay up to some more than what she originally specified, on condition that she finds the requested item or something very similar. Supplies with a much lower price than request (less than 50%) are penalized since they likely represent items in a different market segment. In the formula for partial matches, the last addend considers product category. Products in the same category are favored, because they are presumably more suitable to be used together with the one selected by the user.

COPING WITH VERBOSE DESCRIPTIONS

Languages at the heart of Semantic Web are based upon XML, whose known drawback is verbosity. Usually this is not a concern for Internet based applications (because link bandwidth and host storage capacity are enough for most practical purposes), but surely reduces efficiency of data storage and communication in mobile environments. Adapting ideas and techniques from the Semantic Web vision to ubiquitous scenarios requires to cope with the limited storage and computational capabilities of mobile and embedded devices and with reduced bandwidth provided by wireless links. Here we provide details about a novel efficient XML compression algorithm devised for the purposes of the framework presented in this paper. It is specifically oriented to the packing of standard DIG 1.1 syntax. The XML Schema for DIG format contains 40 tags at most. A DIG document is an XML document exposing specific characteristics. That is, no value is set for any tag; the value of tag attributes is within a well defined finite set of values. A basic distinction among various encoding techniques is in *fixed length* and *variable length* algorithms (Hamming, 1986). In the first case, having a specified alphabet, a fixed bit number is used to encode each symbol: in particular we need $n = \log_2 k$ bits, with k alphabet symbols. A DIG file is encoded by means of `ISO 8859-1` or `UTF-8` encoding. In particular each allowed character can be associated to 1 byte (special characters needing more than 1 byte in `UTF-8` do not belong to the symbol set of DIG). Hence, in order to obtain a good compression rate, we must recur to a variable length coding algorithm: in this case the most efficient algorithm is the *Huffman* one (Huffman, 1952; Cover and Thomas, 1991). It requires to have a *dictionary* containing the correspondences between each symbol and the bits sequence encoding it. This dictionary obviously varies according to the document. Although Huffman algorithm could seem a good choice to compress an ontology in DIG syntax, it does not work well with short semantically annotated DIG descriptions as the ones referred to resource metadata annotation. A resource description is usually few hundred bytes long, so the Huffman compression is sometimes inadequate because a description could be smaller than the dictionary itself. We propose a different DIG compression solution, particularly suitable for pervasive applications, whose structure is shown in Figure 2. We exploit the peculiarity of the DIG format having few, well defined and limited tag elements and being mostly composed of empty XML elements. Three fundamental phases can be identified: **(1)** *data-structures packing*; **(2)** *attribute-values packing*; **(3)** *zlib packing*.

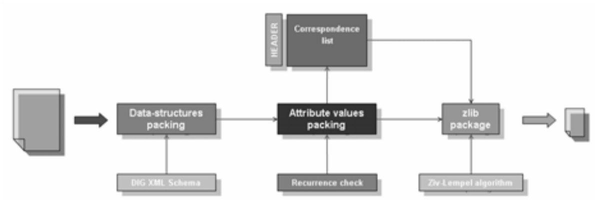


Figure 2. Structure of the proposed DIG compression tool.

(1) *Data-structures packing*. The proposed compression algorithm is based on two fundamental principles. First of all, pure data have to be divided from data-structures; furthermore data and data-structures have to be separately encoded in order to obtain a more effective compression rate. Data-structures are basically XML elements with possible related attributes, whereas data simply are attribute values. Recall that data-structures in DIG syntax are fixed and well defined by means of the DIG XML Schema, whereas data are different from document to document. XML elements are encoded by associating an unambiguous 8-bit code to each structure in a static fashion. Consider that DIG files adopt an encoding which exploits one byte for each character: so an early size saving is performed. Note that the association between XML structures and corresponding code is fixed and invariable. This is a further benefit because it is unnecessary to integrate within the compressed file a header containing the decoding table.

(2) *Attribute-values packing*. In order to pack the attribute values, in the proposed approach a further phase is introduced. Most recurrent words are identified in the previously distinguished data section. They will be encoded with a 16-bit sequence. The second compression stage allows to obtain a further size saving especially in ontologies with recurrent concepts and roles. The second packing phase needs to build and maintain a header of the compressed file containing correspondences between each text string and the related 16-bit code. It is dynamically created and exclusively belongs to a specific DIG document instance. The provided header will be exploited in the decompression steps. Notice that assigned codes differ for the second byte, because the first octet is adopted as padding in order to distinguish the attribute value coding from the ASCII one. The use of this header could compromise compression performances for short files: recall that the size consumption for the header reduces saving obtained with compression. Hence the encoding of all the string values of a DIG file without any a-priori distinction must be avoided. Care has to be paid in the choice of attribute-value strings to encode. A correct compression procedure should properly take into account both the length of an attribute string and its number of occurrences within the file. The minimum length of strings to encode can be trivially established by comparing the size consumption needed to store correspondences *string–code* and the saving obtained with the encoding: in the proposed approach only text attributes with a length of at least three characters will be processed. Furthermore, in order to establish what attribute values (among remaining ones) have to be encoded, we must evaluate the number of occurrences of each attribute *i* (from now on *nr_occurences_i*). We fix a minimum optimum value *nr_occurences_min* and we will encode only *i* attribute values where *nr_occurences_i* > *nr_occurences_min*. We have performed statistical evaluations trying the compression of 72 sample ontologies and evaluating obtained compression rates varying *nr_occurences_min*. Results show the best compression rates are produced by *nr_occurences_min* values within the range [2–8] with an average of 4.03 and a standard deviation in the range [0–0.3]. In the proposed approach we set *nr_occurences_min* = 4, so we will encode only attribute strings with at least three characters recurring at least four times.

(3) *Zlib packing*. The third and final compression step exploits *zlib* library. Although the *zlib* algorithm does not work well when it has to compress a partially encoded input (it is difficult to find more occurrences of the same sequence), the use of *zlib* in our approach resulted however useful especially for large files, where it produces the compression of words excluded by the previous compression steps and of the file header.

PROTOTYPE FRAMEWORK

U-commerce was chosen as reference scenario for evaluating the effectiveness and feasibility of our object discovery framework and architecture.

A central role is played by the user interface component equipped with an integrated RFID reader as well as Bluetooth connectivity. The above logical framework can be adapted to different real scenarios with various physical devices involved. It will be now clarified and motivated in a consumer electronics store case study, where a “smart shopping cart” equipped with a tablet touchscreen, RFID reader and Bluetooth transceiver interacts with the store hotspot at SDP level. UML sequence diagram in Figure 3 shows the role played by these logical elements in a basic use case. We hypothesize that hotspot maintains semantic annotations and context values. Annotations of products in the marketplace refer to a consumer electronics ontology, marked with a specific identifier we indicate $OUUID_E$. Interaction is triggered by inserting an item into the shopping cart, which is detected by a pressure sensor (also simulated in our current environment) and identified by the integrated RFID reader.

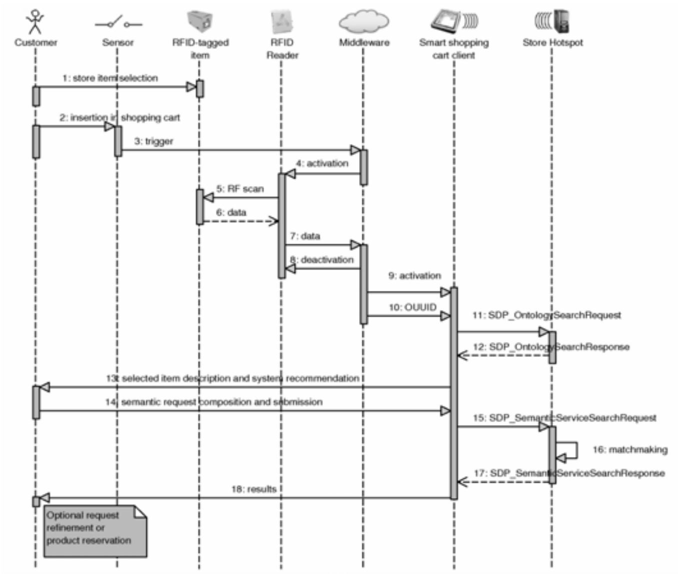


Figure 3. Sequence diagram of a basic use case in our reference scenario.

In the proposed object discovery framework, a session starts after submission from client to server of the ontology identifier $OUUID_E$, in order to agree on the resource category to be adopted in upcoming requests. Semantic annotation describing the selected item will be exploited as basic user request to be adapted or updated for discovering further resources. Feature selection is performed by an intensional navigation of the reference ontology, represented as hierarchy of elements. A tabbed panel allows easy navigation even in large ontologies. User can concentrate on her current focus and at the same time freely change the entry point through the upper tabs which record navigation history (Colucci et al., 2006). Pop-up menus and drag-and-drop are supported to further simplify user interaction.

Simulation test bed

A prototypical scenario was developed to validate the theoretical framework and to evaluate the feasibility and effectiveness of the proposed solution. IBM WebSphere RFID Tracking Kit (Chamberlain et al., 2006) was adopted as development and simulation platform. It is a message-based service-oriented middleware for the integration of RFID systems and other mobile and embedded technologies in enterprise applications. It is based on OSGi (Open Service Gateway Initiative) Alliance open standard for platform-neutral, network-managed SOAs (Service Oriented Architectures) (OSGi Alliance, 2005).

OSGi basic building block is the *bundle*, a self-contained software module whose lifecycle (install/start/update/restart/stop/uninstall) can be managed dynamically through a network. Bundle implementing EPCglobal RFID tag standards has been extended with support to semantic-based product descriptions: RFID tags and readers are software-simulated in our test bed.

Upon this infrastructure, WebSphere RFID Tracking Kit provides MBAF (MicroBroker Application Framework), a framework for event-based notification among software components. It adopts WebSphere MQTT (Message Queue Telemetry Transport), a lightweight publish-subscribe protocol for asynchronous message exchange. Components developed within MBAF are called *agents*. MBAF agents comply to OSGi bundle specifications and an agent behaves like a black box: it subscribes to messages representing events of interest. Upon message receipt, the agent processes its content and publishes zero or more new messages as a result. Overall application behavior is determined by the message flow among agents.

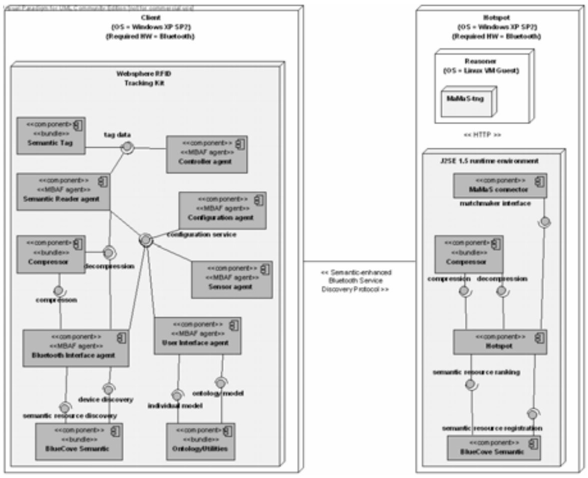


Figure 4. Test bed. White: hardware nodes; yellow: software nodes; blue: components.

This service and message oriented design fits well into ubiquitous computer paradigms, as it is aimed at maximizing modularity, flexibility and scalability. Such properties are essential in volatile and resource-constrained mobile environments, and benefits were noticeable in prototype testing and evolution. Simulation sessions with our prototype platform evidenced that semantic-enhanced object discovery services can coexist with traditional RF identification and tracking applications. Test bed deployment and system components are represented in the UML diagram depicted in Figure 4 and the main elements are highlighted here. Both client and hotspot were deployed on notebook PCs running Microsoft Windows XP. WebSphere RFID Tracking Kit runs on client machine, integrating a Bluetooth communication interface with simulated RFID reader, environmental sensor and user interface by means of dedicated MBAF agents. Controller and Configuration agents are added for supervising the application execution and managing system operating parameters, respectively. Hotspot hosts a J2SE runtime environment for running the service provider and a Linux Virtual Machine for running MAMAS-tng reasoner. Compressor library implementing the algorithm described in the above section is included in both hosts as well as BlueCove[ii], an open source Java Bluetooth library which was extended with support for semantic-based resource discovery. Both these components are deployed as standard Java archives in the hotspot, whereas they are encapsulated in OSGi bundles on the client node.

Example scenario

Let us suppose Mary is looking for a new laptop computer. She notices a quite cheap notebook model, bundled with an office productivity suite. She puts it into the smart shopping cart. Sensor detects the event and the RFID reader is triggered. It reads data stored within the tag attached to the laptop package, then it is deactivated again. Extracted tag data consists of product EPC, ontology identifier $OUUID_E$, semantic-annotated description (stored as a compressed DIG expression) and contextual parameters. Let us suppose that tagged description corresponds to a notebook with Intel Centrino Core Duo CPU, 1 GB RAM, 80 GB hard disk drive, DVD writer and wireless LAN connectivity; it includes Microsoft Windows XP Home Edition OS and an office software suite. Price is $550, delivery time is 0 days and product category is 2. The equivalent expression in DL formalism w.r.t. $OUUID_E$ reference ontology is:

R: notebook ⊓ ∀has_CPU.Intel_centrino_core_duo ⊓ ∀has_HDD.hard_disk_80_GB ⊓ ∀has_disc_recorder.DVD_rec_16X_6X ⊓ ∀has_ram.ram_1_GB ⊓ ∀has_cards.wireless_802_11 card ⊓ ∀has_OS.Windows_XP_Home_edition ⊓ ∀has_software.suite_office

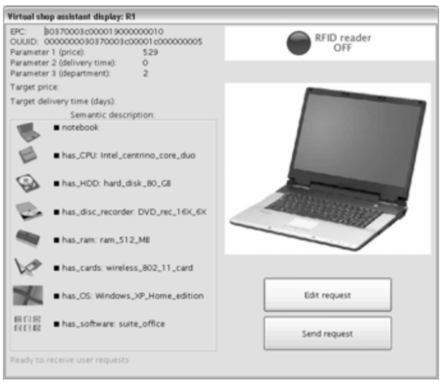


Figure 5. Product details are read via RFID and shown to the user.

As reported in Figure 5, the tablet touchscreen shows the received product details for building further semantic based requests. Let us suppose Mary likes her choice. Now she would like to find some basic accessories.

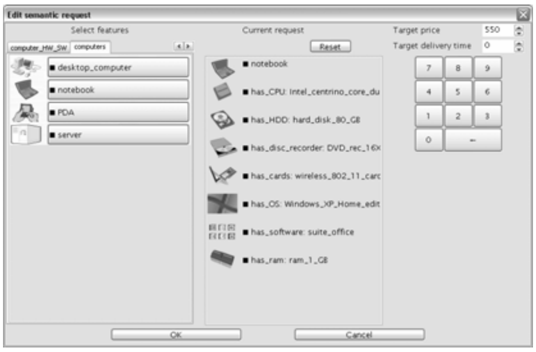


Figure 6. Graphical User Interface for semantic request composition.

She confirms the system-recommended request (shown in Figure 6), which is submitted via the semantic-enhanced Bluetooth SDP from the smart shopping cart to the hotspot.

Let us suppose the following products are available in the store knowledge base:

S1: notebook with AMD Athlon XP-M CPU, 1 GB RAM, 80 GB hard disk drive, DVD writer and wireless LAN connectivity. It is bundled with Windows XP Professional and antivirus software. Price is $599; delivery time is 0 days; product category is 2:

notebook ⊓ ∀has_CPU.AMD_Athlon_XP_M ⊓ ∀has_HDD.hard_disk_80_GB ⊓ ∀has_disc_recorder.DVD_rec_16X_6X ⊓ ∀has_cards.wireless_802_11_card ⊓ ∀has_ram.ram_1_GB ⊓ ∀has_OS.Windows_XP_Professional ⊓ ∀has_software.antivirus

S2: notebook with Intel Centrino Core Duo CPU, 1 GB RAM, 80 GB hard disk drive, DVD writer and wireless LAN connectivity. It is bundled with Linux and an office suite. Price is $529; estimated delivery time is 1 day; product category is 2:

notebook ⊓ ∀has_CPU.Intel_centrino_core_duo ⊓ ∀has_HDD.hard_disk_80_GB ⊓ ∀has_disc_recorder.DVD_rec_16X_6X ⊓ ∀has_cards.wireless_802_11_card ⊓ ∀has_ram.ram_1_GB ⊓ ∀has_OS.Linux ⊓ ∀has_software.suite_office

S3: a desktop computer with Intel Pentium 4 CPU, 1 GB RAM, 250 GB hard disk drive, DVD writer, wireless LAN connectivity and an LCD display. It is bundled with Windows XP Home Edition and an office suite. Price is $499; delivery time is 0 days; product category is 2:

desktop_computer ⊓ ∀has_CPU.Intel_Pentium_4 ⊓ ∀has_HDD.hard_disk_250_GB ⊓ ∀has_display.LCD_display ⊓ ∀has_disc_recorder.DVD_rec_16X_6X ⊓ ∀has_ram.ram_1_GB ⊓ ∀has_cards.wireless_802_11_card ⊓ ∀has_OS.Windows_XP_Home_edition ⊓ ∀has_software.suite_office

S4: a blue notebook bag. Price is $19; delivery time is 0 days; product category is 2:

notebook_bag ⊓ ∀has_color.blue

S5: a silver-colored UMTS mobile phone with dual display and miniSD memory card support. Price is $169; delivery time is 0 days; product category is 1:

mobile phone ⊓ ∀has_connectivity.UMTS ⊓ =2 has_display ⊓ ∀has_display.LCD_display ⊓ ∀has_memory_card.mini_sd

Supply	Compatibility (Y/N)	*rankPotential* score	*rankPartial* score	*f(·)*
S1: notebook with antivirus	Y	6		0.001
S2: notebook with office suite	Y	3		0.236
S3: desktop computer	N		79	0.166
S4: notebook bag	N		26	0.502
S5: UMTS phone	N		23	0.443

Table 8. Matchmaking results.

Hotspot performs the discovery and matchmaking processes as described in Section 3 and returns results via Bluetooth SDP. Matchmaking results for this example are presented in Table 8. The second column shows whether each retrieved resource is compatible with request **R**. If so, the *rankPotential* computed result is shown, otherwise the *rankPartial* computed result is presented. In the last column results of the overall utility function are reported. Note that **S2** is ranked as the best supply for similarity match, despite a longer delivery time than **S1**. This is due to a better *rankPotential* outcome. Among candidate resources for combination, category affinity favors **S4** over **S5**, while **S3** has a clearly poorer match. For each retrieved resource a picture is returned along with matchmaking score, price and description, as displayed in Figure 7.

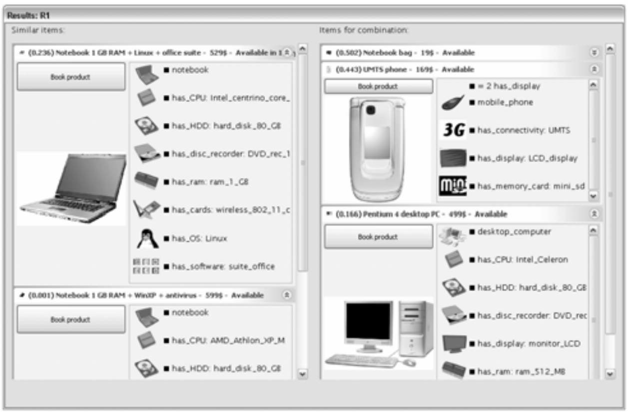


Figure 7. Retrieved results are shown to the user.

EXPERIMENTAL RESULTS

A thorough and significant experimental evaluation of all aspects of system performance requires a complete implementation of our framework into a testbed with real semantic-enabled RFID devices. That would only be possible through partnership agreements with device manufacturers/integrators. Therefore analysis with our current PC-based simulation testbed focused on four groups of performance measures, that can provide valuable and reliable information about practical feasibility and efficiency of the proposed approach: 1) performance of the compression algorithm as a stand-alone tool; 2) impact of compression over Bluetooth system performance; 3) preliminary evaluation of access time of compressed semantic annotations on RFID tags w.r.t. tag scanning performance of EPCglobal RFID systems; 4) estimation of semantic matchmaking processing time.

The following subsections cover methods, results and discussion for each of the four analyses.

Compression algorithm

Performance evaluation of the proposed algorithm has been carried out estimating three fundamental parameters: **(1)** compression rate; **(2)** turnaround time; **(3)** memory usage.

Two stand-alone tools were developed in C language implementing our compression and decompression algorithms, named *DIG Compressor* and *DIG Decompressor*, respectively. Currently, Windows and Linux platforms are supported, leveraging the freely available *zlib* compression library. Tests for compression rate and running time were performed using a PC equipped with an Intel Pentium 4 CPU (3.06 GHz clock frequency), 512 MB RAM at 266 MHz and Windows XP operating system. Tests for memory usage were performed on a PC running

Gentoo GNU/Linux with 2.6.19 kernel version and *Valgrind* profiling toolkit (Nethercote and Seward, 2007). This second PC was equipped with a Pentium M CPU (2.00 GHz clock frequency) and 1 GB RAM at 533 MHz.

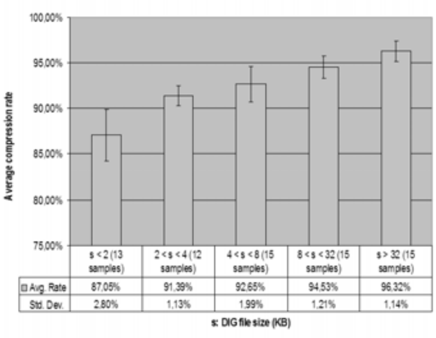
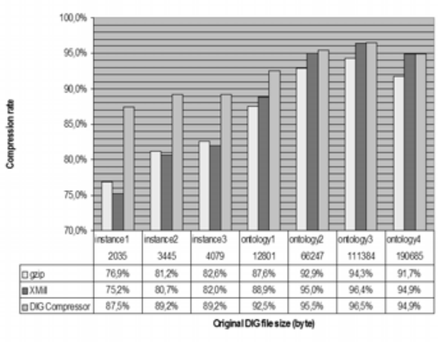


Figure 8. Compression rates obtained by the proposed algorithm.

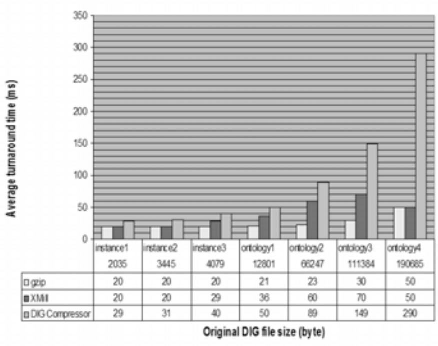
(1) The compression rates achieved by the proposed algorithm was tested with 70 DIG documents of various size. Our aim was to evaluate them for both smaller instance descriptions and larger ontologies. Figure 8 shows average compression rates and standard deviations for different size ranges of DIG input data. Overall average compression rate is 92.58±3.58%. As expected, higher compression rates were achieved for larger documents. Even for very short DIG files (less than 2 KB), however, average compression rate is 87.05±2.80, which is surely satisfactory for our purposes. Comparative evaluation was carried out using *XMill* general purpose XML compressor (Liefke, Suciu, 2000) and *gzip* generic compressor[iii] as benchmarks. Testing the compression rate, the proposed tool allowed to obtain smallest resulting files, as shown in Figure 9. It should be noticed that our algorithm performed significantly better for small DIG documents. This is a very encouraging result, since mobile scenarios usually deal with small XML annotations of available resources.



	instance1 2035	instance2 3445	instance3 4079	ontology1 12801	ontology2 66247	ontology3 111384	ontology4 190685
gzip	76,9%	81,2%	82,6%	87,6%	92,9%	94,3%	91,7%
XMill	75,2%	80,7%	82,0%	88,9%	95,0%	96,4%	94,9%
DIG Compressor	87,5%	89,2%	89,2%	92,5%	95,5%	96,5%	94,9%

Figure 9. Compression rates in gzip, XMill and DIG Compressor

(2) In order to evaluate turnaround time, each test was run 10 times consecutively, and the average of the last 8 runs was taken. Results are presented in Figure 10. It can be noticed that processing times are comparable for documents up to 80 kB. For larger documents DIG Compressor has higher turnaround times than other tools, though absolute values are still quite acceptable. Such an outcome suggests further work should be put into optimizing the implementation for execution speed.



	instance1 2035	instance2 3445	instance3 4079	ontology1 12801	ontology2 66247	ontology3 111384	ontology4 190685
gzip	20	20	20	21	23	30	50
XMill	20	20	29	36	60	70	50
DIG Compressor	29	31	40	50	89	149	290

Figure 10. Turnaround time in gzip, XMill and DIG Compressor.

(3) Finally, memory usage analysis was performed using *Massif* tool of *Valgrind* debugging/profiling toolkit. Massif measures stack and heap memory profile throughout the life of a process. For our comparison, only the memory occupancy peak was considered. Results are reported in Table 9.

DIG document	Original size (B)	gzip	XMill	DigCompressor
Playstation_2_Slim.dig	2035	220	2700	290
Kodak_P880_camera.dig	3445	200	4500	250
Asus_A3FP_Notebook.dig	4079	200	6500	250
Toy_ontology.dig	12801	200	4000	240
Rent_ontology.dig	66247	200	6500	250
clothing_ontology.dig	111384	202	4500	250
electronic_products_ontology.dig	190685	210	4000	260

Table 9 Memory usage peak (kB) in gzip, XMill and DIG Compressor.

DIG Compressor memory usage is only slightly higher than the one of gzip, with high correlation (r = 0.96) between the two value sets. This result could be expected, since our algorithm relies on Ziv-Lempel compression in its last phase. On the contrary, XMill showed a more erratic behavior. Outcomes can be reputed as good because memory-efficient implementations of zlib library are currently available for all major mobile platforms.

Impact of compression over Bluetooth performance

Like other wireless networking technologies, throughput of data transfer between two Bluetooth nodes is influenced mostly by: (1) fading due to obstacles and physical distance between nodes (Zanella et al., 2002); (2) interference from other electromagnetic sources in range (Haartsen and Zürbes, 1999). In particular, Bluetooth operates in the unlicensed 2.45 GHz band also used by IEEE 802.11 wireless LANs, which are widespread in home and business environments. It is therefore important to take the above two factors into proper account when investigating the impact of data compression on Bluetooth application performance in ubiquitous computing scenarios. Since in our approach a domain ontology is typically two or three orders of magnitude larger than individual resource annotations, ontology transfer from hotspot to a client was chosen as performance measure. The consumer electronics store ontology developed for the case study was used: original document size is 187 kB, whereas compressed size is 9.5 kB. Bluetooth transfer time of the uncompressed ontology was compared to the sum of (i) hotspot compression, (ii) transfer and (iii) client decompression time for the same resource. Tests were repeated at 3 client/hotspot distances (1 m, 5 m and 10 m) and both with and without a 802.11b/g WLAN (composed of an access point and a terminal) actively operating in the same room, for a total of six different environmental conditions. Each ontology transfer test was run 10 times consecutively; average values and standard deviations were then calculated.

Figure 11 summarizes results. Reported transfer time when compression is enabled, comprises the time spent in encoding and decoding of ontological data. Compression produces a significant speedup in all cases. At the same time, when compression is enabled, the system is much more resilient to performance degradation due to longer communication distance and interference by an active Wi-Fi network.

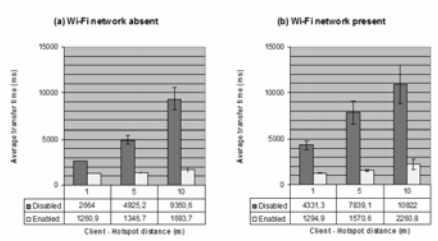


Figure 11. Ontology transfer time via Bluetooth (data compression disabled and enabled).

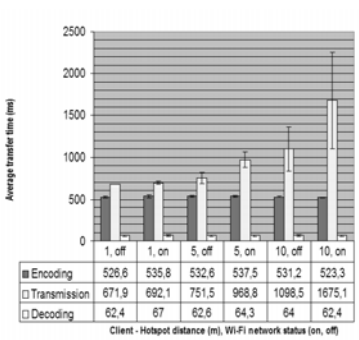


Figure 12. Hotspot encoding, Bluetooth transmission and client decoding times for a compressed ontology in various environmental conditions.

In Figure 12 the overall transfer time is dissected into its three components. Note that data compression occupies a significant share of total time (23.1% to 41.8%), while decompression is almost negligible. As expected, transmission time accounts for much of total variability and is affected by environmental conditions, while compression and decompression times are substantially constant.

Access time of simulated semantic-enhanced RFID tag

This evaluation has been performed with the aim of providing a preliminary judgment of the impact that our approach may have on RFID system performance. Compressed semantic annotations of 40 different marketplace items created for the above case study were used. Their mean size is 266±104 B (range 91-440 B). Simulated RFID data access from each tagged item was repeated 100 times, recording the sum of reading and decompression time. For each item the mean value was then considered.

Results are reported in Figure 13. Average access time is 2.02±0.36 ms, corresponding to a theoretical tag read rate of approximately 500 tags/s. Since tests were run on a software-simulated RFID platform, exact numerical values are not as significant as their order of magnitude. The latter can be sensibly compared to performance of RFID systems compliant with EPCglobal standards for Class 1 Generation 2 UHF RFID systems.

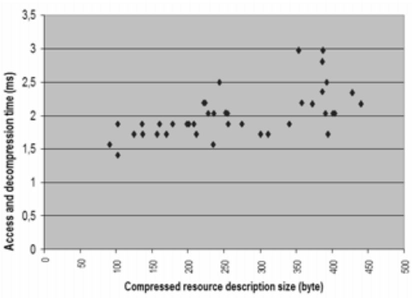


Figure 13. RFID tag reading and decompression time for 40 resource descriptions.

RFID performance in the field highly depends on the application, environment conditions (electromagnetic noise, RFID reader density) and local regulations affecting the available bandwidth. Alien Technology RFID equipment manufacturer claimed maximum tag read rates of 1000 tags/s in environments with good insulation from electromagnetic noise and 50-100 tags/s in noisy environments (2005). Early simulations and tests by universities and independent laboratories estimated read rates ranging from 7 to approximately 100 tags/s in typical application conditions (Ramakrishnan and Deavours, 2006; Kawakita and Mistugi, 2006). Our simulation results are fully compatible with such data, thus providing a very preliminary evidence that adoption of compressed semantic resource annotations on RFID tags does not impair performance of semantic-based RFID applications in the field w.r.t. to traditional ones. The latter, in turn, will not suffer any direct performance degradation from the newly introduced features, as they will read the EPC only. Finally, access time showed a moderate positive correlation (r = 0.60) with annotation size. This may suggest that structure of a DIG annotation has also an impact over the decompression.

Semantic matchmaking processing computation

A semantic enhanced Bluetooth simulated test bed embedded in *ns-2 Network Simulator*[iv] has been used to assess semantic matchmaking processing times. Three ontologies of different level of complexity were examined, and five different demands for each one were submitted to MAMAS-tng reasoning engine. Average response times were recorded. Figure 14 shows the average response time, the number of concepts and the ratio of these two values for each ontology.

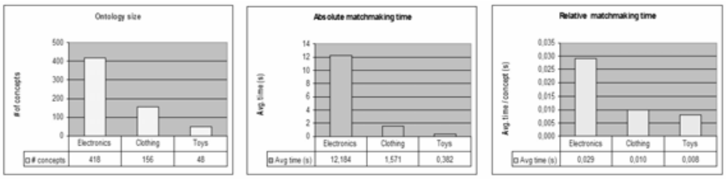


Figure 14. Performance evaluation of semantic matchmaking.

It can be noticed that, for the most complex ontologies, semantic matchmaking time dominates the other system performance measures shown in previous subsections. This problem was also pointed out by Ben Mokhtar et al. (2006), who devised optimizations to reduce online reasoning time in a semantic-based mobile service discovery protocol. The main proposed optimizations were offline pre-classification of ontology concepts and concept encoding: both solutions, however, are viable in matchmaking schemes based on pure subsumption (and therefore able to provide only binary yes/no answers), but are not directly applicable to our matchmaking scheme. Also note that, due to the large amount of time required by the matchmaker computation, RFID reading times have a relative importance within the overall performance evaluation. From this point of view the differences between tag reading time interval in a practical deployment of the system and in the simulated one are uninfluential in an extensive evaluation of the approach.

RELATED WORK

Smart identification technologies and techniques have been discussed in many recent research proposals in the field of ubiquitous computing. In (De et al., 2004) a pervasive architecture for tracking mobile objects in real-time is presented, aimed at supply chain and B2B transaction management. A global and persistent IT infrastructure is necessary in order to interface RFID system within partner organizations through the Internet. These requirements make the approach less suitable for B2C and C2C scenarios especially in MANET contexts. An XML formalism named Physical Markup Language (PML) is used to describe objects and processes. However, it does not exploit any semantics of resource descriptions and only allows string-matching discovery.

Römer et al. (2004) present two frameworks (respectively based on Jini and UDDI service discovery protocols) for ubiquitous computing applications using smart identification technologies. Core design abstractions such as object location, neighborhood, composition, history and context make them flexible. Nevertheless, as admitted by the authors, scalability issues are present. A further limitation is that semantics of object properties and capabilities is not explicit, but it is encapsulated in either Java classes or Web Services.

A key usability issue of mobile and ubiquitous computing solutions is to assist the user in timely and unobtrusive ways, without either being inappropriate or altering her habits. In our framework, interaction is started implicitly, *i.e.* by user actions in the real world. Schmidt et al. (2000) focus on implicit *Human-Computer Interaction* (HCI) in pervasive computing. The authors introduce a wearable RFID solution enabling operations on an information system simply by picking up or using an operation-related tagged object. The proposed system has been also integrated with SAP R/3 in a case study. Since no semantic information is associated to tags, however, RFID is used merely as a bridge towards the centralized information infrastructure. Interaction patterns are quite unnatural in some cases, because real-world objects are used to start even those tasks that need explicit HCI (e.g. editing a document in a word processor).

In (Siegemund and Florkemeier, 2003), interaction patterns between users endowed with GSM phones and everyday objects are investigated. Exploited objects are augmented through active RFID transponders equipped with on-board sensors, modest computing capabilities and Bluetooth connectivity. An infrastructure enabling a hybrid implicit-explicit HCI model is implemented. In order to minimize user involvement, an “invisible” pre-selection based on contextual conditions is performed. Elected objects send *interaction stubs* to the GSM terminal of the user. Basically stubs are SMS templates to issue commands to objects or to ask their status. Authors claim that proposed interaction patterns are perceived as natural, nevertheless sending SMS messages to special objects requires too much user attention so altering normal relationships between people and everyday things. The need for a costly communication link such as GSM is an open issue.

In (Kawakita et al., 2004) a support system aimed at enhancing information exchange within a conference room is presented. RFID-enabled badges are given to the meeting attendants everyone having a remotely stored profile. Each room has an RFID reader. A location and time aware middleware tracks participants while entering or exiting meetings. Upon this basic infrastructure, location-based instant messaging and file sharing services are provided. This is a good example of implicit HCI in ubiquitous computing, even though the applicability is somewhat limited by the preliminary explicit profiling of both users and conference events.

Data compression of XML based ontological languages is another major problem tackled in our proposal. Gzip (along with its library version *zlib*) is perhaps the most popular universal compression tool. It is based on a variant of the LZ77 algorithm (Ziv and Lempel, 1977). Among general purpose compression algorithms and tools, the *PAQ* family (Mahoney, 2005) shows the best compression rates. It is based on two fundamental ideas which evolve upon classic Huffman encoding algorithm: *context mixing* and *arithmetic coding*. A major drawback of PAQ algorithms is their huge processing and memory requirements, which currently are far beyond the capabilities of mobile computing devices.

By exploiting structural peculiarities of XML better compression rates can be achieved than most general purpose tools. *XMill* (Liefke and Suciu, 2000) is an efficient XML compressor. Its approach is based upon the separation of XML content into different *containers*, which are stored sequentially in the output file. Each container is compressed by a specialized module.

XMill performances are better than generic compressors only for medium and large XML documents.

CONCLUSION AND FUTURE WORK

In this paper we proposed a fully unified framework integrating RFID technologies with enhanced Bluetooth SDP supporting formal semantics. Objects tagged with RFID transponders carry a semantically annotated description so permitting to implement an advanced object discovery. Thanks to the semantic-enhanced SDP in Bluetooth it is possible to exploit reasoning services from everywhere in the marketplace also in case of lack of dependable and stable network links. The proposed approach aims to avoid the need for stable Internet connections in order to make the framework really “mobile”, as a Bluetooth infrastructure is deeply different from a fixed one in terms of resource consumption and required support, therefore more suitable for giving a *non-invasive* structure to fully decentralized volatile environments. This is a good feature in sight of a future work in the mobility direction aimed to make the reasoner resident on mobile devices. Some slight modifications to the EPCglobal standards have allowed to support ontology-based data as well as non standard inference services, while keeping total compatibility with legacy applications. The framework includes a compression tool based on an efficient algorithm specifically aiming at size reduction of document instances expressed in various ontological languages. The complete framework has been implemented within a message-oriented commercial middleware in order to test the feasibility and the usability of the proposed solutions.

Current limits of the proposed approach and tool emerge because the time spent in performing the overall discovery and ranking procedure is still somewhat high to be definitively acceptable (even if it has to be traded-off with a far higher quality of the discovery w.r.t. traditional approaches). Future work is aimed to an optimization of the reasoner features for a dedicated utilization in pervasive and ubiquitous applications.

A real world practical application of the proposed framework has to face concrete difficulties of the modification of a closed standard as the EPCglobal currently is, and furthermore exhaustive preliminary studies have to be performed in order to test the feasibility of the approach in case of multiple readings (in those cases the collision phenomenon must be taken into account, although it is not expected to be a real problem). Finally privacy and security issues have to be faced in order to make the proposed approach ready for a real-world commercial exploitation.

References

Alien Technology (2005). *EPCglobal Class 1 Gen 2 RFID Specification*. Whitepaper. Morgan Hill, CA: Alien Technology Corporation.

Baader, F., Horrocks, I., & Sattler, U. (2003) Description Logics as Ontology Languages for the Semantic Web. *Lecture Notes in Artificial Intelligence*, Springer.

Bechhofer, S., Möller, R., & Crowther, P. (2003): The DIG Description Logic Interface, *Proceedings of the 16th International Workshop on Description Logics (DL'03)*.

Ben Mokhtar, S., Kaul, A., Georgantas, N., & Issarny, V. (2006): Efficient Semantic Service Discovery in Pervasive Computing Environments. *Proceedings of the ACM/IFIP/USENIX 7th International Middleware Conference, Middleware '06* (December 2006).

Berners-Lee, T., Hendler, J., & Lassila, O. (2001): The Semantic Web. *Scientific American* (vol. 248, num. 4, pp. 34-43).

Borgida, A. (1995). Description Logics in Data Management. *IEEE Transactions on Knowledge and Data Engineering* (vol. 7, num. 5, pp. 671-682).

Brachman, R., & Levesque, H. (1984). The Tractability of Subsumption in Frame-based Description Languages. *4th National Conference on Artificial Intelligence – AAAI-84* (pp. 34-37).

Chamberlain, J., Blanchard, C., Burlingame, S., Chandramohan, S., Forestier, E., Griffith, G. et al. (2006). *IBM WebSphere RFID Handbook: A Solution Guide* (1st ed.). IBM International Technical Support Organization.

Colucci, S., Di Noia, T., Di Sciascio, E., Donini, F. M., Ragone, A., & Rizzi, R. (2006). A semantic-based fully visual application for matchmaking and query refinement in b2c e-marketplaces. *8th International conference on Electronic Commerce, ICEC 06* (pp. 174-184).

Cover, T. M., & Thomas, J. A. (1991). *Elements of Information Theory*. John Wiley and Sons Inc.

De, P., Basu, K., & Das, S. (2004). An ubiquitous architectural framework and protocol for object tracking using RFID tags. *The First Annual International Conference on Mobile and Ubiquitous Systems, Networking and Services, MOBIQUITOUS 2004* (pp. 174–182).

Di Noia, T., Di Sciascio, E., Donini, F. M., & Mongiello, M. (2004). A system for principled matchmaking in an electronic marketplace. *International Journal of Electronic Commerce* (vol. 8, num. 4, pp. 9-37).

Donini, F. M., Lenzerini, M., Nardi, D., & Schaerf, A. (1996). Reasoning in Description Logics. *Principles of Knowledge Representation: Studies in Logic, Language and Information*, CSLI Publications, pp. 191-236, Eds: G. Brewka.

EPCglobal Inc. (2005a): *EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz-960 MHz*. January 2005.

EPCglobal Inc. (2005b). *Object Naming Service (ONS - ver. 1.0)*, October 2005.

Haartsen, J. C., & Zürbes, S. (1999). *Bluetooth Voice and Data Performance in 802.11 DS WLAN Environment*. Ericsson Sig Publication.

Hamming, R. W. (1986). *Coding and Information Theory*. Prentice Hall.

Horrocks, I., van Harmelen, F., Patel-Schneider, P., Berners-Lee, T., Brickley., D. (2001). *DAML+OIL Specifications*. Retrieved May 01, 2007 from http://www.daml.org/2001/03/daml+oil-index.html

Huffman, D. A. (1952). A method for the Construction of Minimum Redundancy Codes. *IRE* (pp. 1098-1101).

Kawakita, Y., Wakayama, S., Hada, H., Nakamura, O., & Murai, J. (2004). Rendezvous enhancement for conference support system based on RFID. *International Symposium on Applications and the Internet Workshops, SAINT2004* (pp. 280–286).

Kawakita, Y., & Mistugi, J. (2006). Anti-collision performance of Gen2 Air Protocol in Random Error Communication Link. *International Symposium on Applications and the Internet Workshops, SAINTW'06* (pp. 68-71).

Liefke, H., & Suciu D. (2000). XMill: an efficient compressor for XML data. *SIGMOD Rec.* (vol. 29, num. 2, pp. 153-164).

Mahoney, M.V. (2005). *Adaptive Weighing of Context Models for Lossless Data Compression*. Florida Tech. Technical Report, CS-2005-16.

Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D. (n.d.). *OWL-S: Semantic Markup for Web Services*. Retrieved May 01, 2007 from http://www.daml.org/services/owl-s/1.1/overview/

McGuinness, D., Fikes, R., Hendler, J., & Stein, L. (2002). DAML+OIL: An Ontology Language for the Semantic Web. *IEEE Intelligent Systems* (vol. 17, num. 5, pp. 72-80)

Nethercote, N., & Seward J. (2007). Valgrind: A Framework for Heavyweight Dynamic Binary Instrumentation. *Conference on Programming Language Design and Implementation - PLDI 07*. ACM SIGPLAN.

OSGi Alliance (2005). *About the OSGi Service Platform* (rev. 4.1). San Ramon, CA: OSGi Alliance.

Ramakrishnan, K. M., & Deavours, D. D. (2006). Performance benchmarks for passive UHF RFID tags. *Proceedings of the 13th GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communication Systems* (Nurenberg, Germany; March 27-29, 2006).

Römer, K., Schoch, T., Mattern, F., & Dübendorfer, T. (2004). Smart Identification Frameworks for Ubiquitous Computing Applications. *Wireless Networks* (vol. 10, num. 6, pp. 689–700).

Ruta, M., Di Noia, T., Di Sciascio, E., & Donini, F. M. (2006a). Semantic-Enhanced Bluetooth Discovery Protocol for M-Commerce Applications. *International Journal of Web and Grid Services* (vol. 2, num. 4, pp. 424-452)

Ruta, M., Di Noia, T., Di Sciascio, E., Donini, F. M., & Piscitelli, G. (2006b). Advanced Resource Discovery Protocol for Semantic-enabled M-commerce. *Encyclopedia of Mobile Computing and Commerce (EMCC)*. Idea Group, USA.

Schmidt, A., Gellersen, H., & Merz, C. (2000). Enabling implicit human computer interaction: a wearable RFID tag reader. *The Fourth International Symposium on Wearable Computers* (pp. 193–194).

F. Siegemund & C. Florkemeier (2003). Interaction in Pervasive Computing Settings using Bluetooth-Enabled Active Tags and Passive RFID Technology together with Mobile Phones. *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, PerCom 2003* (pp. 378–387; March 2003).

Shadbolt, N., Hall, W., & Berners-Lee, T. (2006). The Semantic Web revisited, *IEEE Intelligent Systems* (vol. 21, num. 3, pp. 96-101)

Traub, K., Allgair, G., Barthel, H., Bustein, L., Garrett, J. et al. (2005). *EPCglobal Architecture Framework*. Technical report, EPCglobal Inc., July 2005.

Zanella, A., Tonello, A., & Pupolin S. (2002). On the impact of fading and inter-piconet interference on Bluetooth performance. *The 5th International Symposium on Wireless Personal Multimedia Communications* (vol. 1, pp. 218-222).

Ziv, J., & Lempel, A. (1977). A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory* (vol. 23, num. 3, pp. 337-343).

[i] *i.e.,* an ubiquitous environment where mobile peer users - both buyers and sellers - can submit their advertisements, browse through available ads and be assisted in finding the best available counterparts to meet their needs so beginning a commercial transaction.

[ii] http://sourceforge.net/projects/bluecove/

[iii] http://www.gzip.org/

[iv] http://www.isi.edu/nsnam/ns/