Comput Syst Sci & Eng (2010) 3: 235–249

International Journal of
Computer Systems
Science & Engineering

A hybrid ZigBee/Bluetooth approach to mobile semantic grids

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

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

Mobile semantic grids are characterized by high heterogeneity and volatility of their component nodes. Hence, discovery procedures more flexible than the ones borrowed from wired approaches are desirable. We present a hybrid ZigBee/Bluetooth grid infrastructure where Knowledge Representation techniques and technologies have been exploited to perform a capillary data dissemination, to interconnect the user with interface nodes and to perform an advanced resource discovery. A case study is reported along with experimental results on a prototype implementation.

1. INTRODUCTION

In the last few years, distributed collaborative systems, namely *Computational Grids*, are assuming a more and more relevant role in the field of distributed computing. Such infrastructures integrate heterogeneous resources as computers, storage systems, network apparatus exploiting common interfaces and general purpose communication protocols.

The *Semantic Grids* refer to a Grid computing approach where component resources, internal data and available services are annotated following a semantic model. In particular, Knowledge Representation formalisms allow to use metadata to describe all the relevant processes and actors of such architectures in a machine understandable format. This facilitates automatic discovery and integration of resources as well as advanced tasks involving different nodes: computational resources are brought together to create virtual organizations. Hence, the Semantic Grids can be seen as an evolution of current Grids so that "information and services are given well-defined meaning, better enabling computers and people to work in cooperation" [1].

Mobile Semantic Grids are the latest evolutionary stage of computational Grids. They are highly dynamic environments: whatever mobile device in radio range could potentially belong to the Grid, but in the same way it could exit unpredictably. The reduced availability of both storage and computational capabilities poses complex issues, but the absence of spatial constraints fosters interesting applications. These requirements call for a flexible infrastructure whose resource discovery and sharing mechanisms allow to cope with the intrinsic unforeseeable nature of such a context. Differently from the traditional "static" approach, an efficient discovery paradigm for mobile semantic Grids should cope with mobility of nodes and with Grid variable consistence. Hence, more flexible discovery procedures are desirable than the ones borrowed from wired approaches. They are based on trivial syntactic comparisons between the user request and Grid resources, providing only binary *match/no-match* outcomes. Unfortunately, perfect request/resource overlap is a case too uncommon to be realistic, particularly in mobile environments. In a more efficient way, it should also take into account partial correspondences, possibly providing a measure of the semantic similarity degree between a request of the user and the available resources.

In this paper we draw approach and formalisms from the Semantic Web initiative and adapt them to Mobile Semantic Grids. The adopted reference communication protocol is ZigBee [2], an emerging wireless standard for low range transmissions. ZigBee adapters are characterized by a compact size and by a thrifty power consumption; hence they are particularly suitable for mobile embedded applications [3]. Such a protocol allows a great network scalability and can interconnect a large number of nodes in a simple fashion. Nevertheless, ZigBee presents other serious drawbacks. First of all, it is too recent to be largely integrated in

off the shelf devices. Furthermore, the reduced battery drain is paid in term of bareness of allowed application infrastructures. The protocol supports only low data rates w.r.t. other widespread wireless standards such as Bluetooth [4] and the application layer is somewhat simplistic for a significant exploitation in more advanced contexts.

Leveraging the basic likeness between ZigBee and Bluetooth standards, in our approach we integrate the semantic-based Bluetooth Service Discovery Protocol (SDP) presented in [5] and the ZigBee application layer, to enable a semantic Grid architecture. The Bluetooth standard is adopted to interface the user with the actor nodes within the Grid, i.e., nodes able to retrieve information from lower levels and to perform advanced services. On the contrary, inter-node communication is managed via ZigBee.

The remaining of the paper is structured as follows: in the next Section we present the state of the art in the mobile semantic Grid field. Section 3 introduces basic notions on the ZigBee protocol and outlines the proposed semantic-based enhancements. In Section 4 the proposed approach is presented: with reference to notions introduced in Section 3, our data dissemination and discovery framework is outlined. The added value of the proposal with respect to currently adopted solution presented in Section 2 emerges with the aid of a case study which is presented in Section 5, where an illustrative example clarifies and motivates the proposed framework. Section 6 presents some experimental results corroborating the approach and related to previously outlined system details. Section 7 closes the paper.

2. MOBILE SEMANTIC GRID: STATE OF THE ART

The Computational Grid notion was born in the mid-1990s to denote "a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high-end computational capabilities" [6], like the electrical power grid grants access to electricity. First-generation Grids were created within scientific communities or industrial consortia, interconnecting large supercomputing centers. Those early experiences allowed to better understand the technological challenges imposed by the Grid vision: a "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations" [7] was the basic purpose of a Grid. A Virtual Organization (VO) can be defined as a community (of individuals and/or institutions) with specific rules and policies allowing a controlled resource sharing. Main requirements for a distributed computing system to be qualified as a Grid [8] are:

1. coordination of resources not subject to centralized control;
2. use of standard, open, general-purpose protocols and interfaces;
3. delivery of various kinds of non-trivial Quality of Service (QoS).

In today's second-generation Grids, a Grid middleware allows the cooperation of virtual organizations on a global scale and grants the needed interoperability among heterogeneous distributed systems, each providing resources managed by pre-existing local policies. The infrastructure targets technical challenges in communication, information, scheduling, data access, security, and fault detection areas. Grid middleware architecture was designed as a stack of protocols and services, including: standard protocols (such as HTTP, LDAP and PKI) for network communication and authentication; a set of basic services for resource discovery, management, access and coordination; Application Programming Interfaces (APIs) and Software Development Kits (SDKs) to enable the development of applications that leverage the Grid infrastructure. The Globus Toolkit[1] and Legion[2] are perhaps the most representative Grid infrastructures.

Current indications in effectively using the large amounts of data involved in Grid operations to produce useful information and usable knowledge are the main driver for next-generation Grid systems [9]. Extensive research, carried out in latest years following the Semantic Web initiative [10], provided strong evidence that knowledge representation techniques can be fruitfully exploited to describe resources in geographically distributed heterogeneous systems. The Semantic Web endeavor aims to provide Web information contents with unambiguous meaning by means of semantic-based annotations (metadata) expressed in languages such as RDF[3] (Resource Description Framework) and OWL[4] (Ontology Web Language).

Similarities between the Web and Grid environments (see the first and second requirement above) have suggested that a knowledge-based approach could be also applied to describe and manage resources, services and components within a Grid. Knowledge representation and reasoning techniques could be exploited to enhance and automate Grid key processes, such as resource/service discovery, composition, negotiation and information elicitation from data sources. This, in turn, would enable a new generation of modular Grid applications that are both simpler to build and more flexible. Consequently, the notion of *Semantic Grid* was proposed. The Semantic Grid vision integrates relevant aspects from three technology areas: Grid Computing, Semantic Web and Web Services. A Semantic Grid can be seen as a Service Oriented Architecture (SOA), where *agents* (acting on behalf of users) exchange services through *contracts* established by *negotiation* within a *marketplace* [11]. Each service is characterized by one or more providers (either individuals or institutions), which set terms and conditions for service access. The Open Grid Services Architecture[5] (OGSA) is the evolution of Globus, embracing Web Service technologies. It is useful to point out that, in a Grid environment, the meaning of "service" includes access to resources shared in a VO: as stated above, resource sharing requires a preliminary agreement between provider and user upon several non-trivial concerns (e.g., resource properties, authorization, QoS, accounting), according to a contract (i.e., a Service Level Agreement, in SOA words).

It is natural to describe a SOA in terms of *service life-cycle*. It consists of three stages [11]:

1. *Creation*: the act of making a service available for fruition. In this step the service provider may need to advertise the resource in order to let potential users know about its ex-

[1] Globus Toolkit, Globus Alliance: http://www.globus.org/toolkit/

[2] Legion - A Worldwide Virtual Computer: http://legion.virginia.edu/

[3] RDF Primer - W3C Recommendation, 10 February 2004: http://www.w3.org/TR/rdf-primer/

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

[5] OGSA, Globus Alliance: http://www.globus.org/ogsa/

istence and characteristics. Depending on the underlying computing infrastructure, advertisement can be performed in either centralized or decentralized way, through *publishing* into a service directory or information *dissemination* respectively.

2. *Procurement*: the negotiation stage between a service provider and a (potential) user, where a contract is established.

3. *Enactment*: the stage where the service is actually brought to fruition.

Within the Grid, several marketplaces can be created by different communities having specific interests and needs. The entity that runs a marketplace can grant access either to all agents in the Grid or to qualified entities in a controlled fashion. The entire Grid system can thus be viewed as a large collection of services located in various marketplaces. Services from different providers can be chosen and combined in flexible (even unimagined) ways in order to build full-fledged applications customized to user's needs.

Traditional Grid infrastructures have been concerned with sharing of fixed resources among fixed users. On the contrary, the ever-growing trend toward mobile and ubiquitous computing is posing new challenging requirements. The main aspects of ubiquitous computing are: mobility of users and devices; device pervasiveness into the environment [12]. Hence, devices must be capable of obtaining incremental information from the environment and adapt their behavior, while moving. Current research projects on *Mobile Grids*, such as Akogrimo [13] and Mobile OGSI .NET [14], are studying approaches and techniques to integrate mobile devices within Grid environments [15].

The so-called Mobile Dynamic Virtual Organizations (MDVO) are "virtual organizations whose members are able to change locations while provided or consumed services remain available even after temporary loss of reachability, and while running or yet to be initiated workflows adapt to changed conditions" [13]. For being an MDVO, not all members of a VO have to be mobile. Nevertheless, user and service mobility highly increases the dynamic nature of the Grid. The vision of a *Mobile Semantic Grid*, adding mobility and ubiquity to the Semantic Grid SOA, implies greater complexity of Grid operations which must be adapted to the limited processing and storage resources of mobile Grid nodes.

To the best of our knowledge, the literature does not propose fully comprehensive approaches specifically devised for Mobile Semantic Grids. ZigBee technology is mainly adopted in wireless sensors network (WSN) designs for several scenarios, such as home/office automation, agriculture and continuous health monitoring. Wireless Semantic Sensor Networks (WSSNs) [16] are nowadays very actively investigated and they share basic technologies with Mobile Semantic Grids. Nevertheless, architecture scale and applications are different, so requiring specialized paradigms and design choices. Evidence is provided by Gridstix [17], a ZigBee-based WSN for flood monitoring, which also embed a mobile Grid middleware. A simple set of ad-hoc primitives, lacking explicit semantics, was adopted for resource description and discovery. As a result, flexibility of the system was limited and subsequent evolutions [18] focused on the WSN rather than the Grid perspective. Furthermore, acknowledging range and data rate limitations of ZigBee, authors adopted IEEE 802.11 or GPRS cellular protocol for inter-cluster communications. Similarly, several investigations compared ZigBee and Bluetooth technologies (e.g., [19, 20]) and suggested that hybrid infrastructures can combine their respective strenghts [21].

3. SEMANTIC-ENHANCED ZIGBEE PROTOCOL

The ZigBee stack protocol [2] is based on the *IEEE standard 802.15.4*, which defines the physical and MAC levels. Upper layers and specifically the application one have been formalized by the ZigBee Alliance.

3.1 Basics

ZigBee is a wireless standard providing data rates up to 250 kbit/sec in the 2.4 GHz ISM band [2]. Network nodes are unambiguously identified by 64 bit MAC addresses whereas the medium access is regulated via the CSMA-CA algorithm which aims to avoid collisions thanks to the RTS/CTS mechanism. In the definition of the network layer of the stack (NWK) two different device categories are identified:

– *Full Function Devices* (or FFD): i.e., devices with good storage and computation capabilities able to support all the standard features. FFDs can play the role of network coordinators managing routing functionalities among Personal Area Network (PAN) members.

– *Reduced Function Devices* (or RFD): i.e., apparatus with restricted capabilities which can be used only as network end points.

Two different network topologies are enabled:

– *star*: end devices communicate only with a coordinator node. All the network PDUs (Protocol Data Units) transit through it;

– *peer to peer*: devices can communicate with each other (in particular the RFDs can communicate only with the related FFD). In a p2p ZigBee-based network there is only one coordinator node.

Exploiting previous network topologies, it is possible to build more complex infrastructures where an FFD acts as *CLuster Head* (CLH) and more clusters are interconnected by means of their CLHs to build a *Cluster Tree*. Also in a cluster tree topology, there is a single PAN coordinator chosen from the Heads of component clusters.

Finally, from the device standpoint, two fundamental notions have to be defined: the *Cluster* and *Profile* ones. A cluster is a data flow in input/output from/toward a ZigBee device; it is identified by means of a unique *cluster identifier*. A profile is a collection of device descriptions, which participate to build an application. For instance, a thermostat on a node communicates with a furnace on another node. Together, they cooperatively work to define a heating application profile. Each ZigBee profile is labeled by an unambiguous identifier assigned by the ZigBee alliance.

In what follows we will focus on the Application layer of the ZigBee protocol stack and we will give a closer look to functions and data structures useful for the further explanation of the proposed approach.

3.2 ZigBee Application layer

The ZigBee Application level is composed by three different sub-layers:

- APplication support Sublayer (APS)
- Application Framework (AF)
- ZigBee Device Object (ZDO)

The ZDO and AF levels completely refer to the APplication Layer (APL), whereas APS is an interface level between APL itself and the underlying network layer. Features of single sub-layers are explained in detail hereafter.

APS. The application support sub-layer is basically an interface level between the NWK and APL layers. It can be further divided into: (i) *APS Data Entity* (APSDE) which provides the data transmission service for the application PDUs (APDUs) exchanged by two or more devices located on the same network; (ii) *APS Management Entity* (APSME) providing services for discovery and binding of devices and maintaining a database of managed objects, the APS Information Base (AIB).

AF. ZigBee specification defines *Application Objects* as components located on the top portion of the application layer customized by the manufacturer implementing an application [2]. Hence, the AF can be considered as the environment where application objects are hosted on devices. Each implemented application object is associated to a specific component *endpoint*. The standard allows to define up to 240 different application objects, each interfacing an endpoint indexed from 1 to 240. Furthermore, endpoint 0 is reserved for the data interchange with the ZDO and endpoint 255 is reserved for the broadcast addressing toward all the application objects. Endpoints 241-254 are reserved for future use. By exploiting the previous framework (managed by the APSDE Service Access Point, APSDE-SAP), the AF provides a Key Value Pair (KVP) service and a generic message (MSG) service.

ZDO. The ZigBee Device Object sub-layer manages functionalities which provide an interface between the application objects, the device profile and the APS. The ZDO is located between the application framework and the application support sub-layer. The ZDO allows to: (i) initialize both the APS and NWK levels; (ii) assemble configuration information from the end applications to both implement discovery and security and allow communication and binding among devices. The ZDO exposes some public interfaces for the application objects in the AF layer to enable both device control and network functions. Furthermore, the ZDO provides an interface for lower portions of the stack exploiting the endpoint 0, through the APSDE-SAP for data messages, and through the APSME-SAP for control messages. Among features managed by the ZDO, specific relevance assumes the *binding management* and the *device/service discovery*. The binding happens associating an input cluster of a ZigBee device with an output cluster of another one. This can be made only if both devices belong to the same ZigBee profile. Made associations are stored in binding tables maintained by each FFD node. By exploiting this simple mechanism the PAN addressing is built. It follows three different modes:

- direct: both device address and endpoint where the destination application object is located have to be specified;
- indirect: only the sender address is specified. It is exploited by RFD nodes which do not have memory enough to host a binding table. A PDU forwarding is performed by the PAN Coordinator or the Cluster Head. They exploit the binding table to determine the address of the receiver;
- broadcast: the message is sent to all device endpoints.

The application objects provide *device discovery* and *service discovery* features. When queried, device discovery allows to return the IEEE address of a specified device (end device), along with the addresses of all associated devices (in case of coordinators or routers). The service discovery allows to detect what services are offered on each endpoint of a device by respective application objects. A device can perform the discovery of active endpoints either on a single node or on all nodes. Furthermore a device can perform the discovery of specific services matching given criteria (profile and cluster identifiers). In order to enable the detection of each service by the other PAN members, it has to be associated with a descriptor expressed in an XML compressed format. The ZigBee Alliance define the following service descriptors:

- node: which contains information about the node features;
- node power: which provides information about the battery level of a node;
- simplex: which contains general information about the device resident on an endpoint;
- complex: which contains detailed information about the device resident on an endpoint;
- user: a user-defined descriptor.

3.3 Semantic enhancements

In the proposed approach, the ZigBee protocol stack has been readjusted to support Knowledge Representation techniques and technologies introducing semantics of exchanged information. Modifications have been designed with the aim of granting the reuse of standard discovery procedures without variations. Modifications safeguard *Profile* and *Cluster* notions [2]. A public profile was defined with `profileId=0x00FF` namely *Semantic Profile*, featuring ZigBee nodes supporting semantics. When receiving semantic-based information, a node not managing that profile has to simply discard it.

A set of AF PDUs were introduced, characterizing the semantic enhancements of the Application Framework (*semantic-enabled AF*) which enable service discovery within the Semantic Profile. The overall packet specification is reported in Table 1.

To send/receive semantic-based PDUs, we introduce the following functions in the APSDE interface described above:

- *APSDE_Semantic_req* which enables the transfer of a semantic frame from the *semantic-enabled AF* layer to the APS one (output flow).
- *APSDE_Semantic_indication* which allows to perform the opposite transfer, that is from the APS level to the *semantic-enabled AF* one (input flow).

Table 1 Assigned values for the Frame Type packet field

FRAME TYPE	FRAME NAME
0000	Reserved
0001	KVP
0010	MSG
0011	**AdvertisementMSG**
0100	**ServiceRequestMSG**
0101	**CacheEntryMSG**
0110	**ServiceDescriptionRequestMSG**
0111	**ServiceDescriptionMSG**
1000	**ReasonerReplyMSG**
1001-1111	Reserved

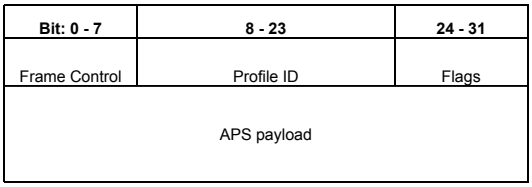

Figure 1 Structure of a semantic APDU

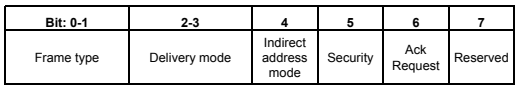

Figure 2 Frame Control field

Fig. 1 and Fig. 2 report the structure of a semantic APDU with related `Frame Control` field.

Among values of the *Frame Control* field, the *Frame type* one has a specific relevance. It is used to label the semantic-based PDUs among others (see Table 2).

As said, in case of semantic-based frames the *Profile Id* takes the 0x00FF value. Furthermore, one bit of the *Flags* field we call *Continuation State* indicates (if set) that another complementary frame is arriving. Finally, the APS payload contains one of the protocol frames reported in Table 1.

The *Type Identifiers* are reported in Table 3, which shows the ZigBee protocol standard data types along with two new types. The first one refers to Ontology Universally Unique IDentifier (OUUID) unambiguously marking ontologies [5]. The second one refers to the format for semantic annotations, DIG (Description Logics Implementation Group) language [22]. They have been introduced exploiting two reserved identifiers (`0101` and `0110`) with the aim of supporting the semantics in data exchange.

4. SEMANTIC-BASED ADVANCED MATCH SUPPORT FOR MOBILE GRIDS

A hybrid ZigBee/Bluetooth architecture is proposed in order to provide the needed connectivity and to support discovery services. The ZigBee protocol stack has upgradeability, low energy consumption and topology self-configuration that are required to build the basic infrastructure of dynamic mobile semantic Grid environments. ZigBee wireless networks can scale out to thousands of nodes and each node has an operating space of about 50 m, thus fixed and mobile nodes can provide radio coverage

Table 2 Allowed values for the Frame Type field

Frame type	Frame Type Name
00	Data
01	Command
10	Acknowledgement
11	**Semantic**

Table 3 Type Identifier allowed values

Data type identifier	Data type	Length (byte)
0000	No data	0
0001	Unsigned 8-bit integer	1
0010	Signed 8-bit integer	1
0011	Unsigned 16-bit integer	2
0100	Signed 16-bit integer	2
0101	**OUUID**	**2**
0110	**DIG description**	**Defined in first octet**
0111 - 1010	Reserved	/
1011	Semi-precision	2
1100	Absolute time	4
1101	Relative time	4
1110	Character string	Defined in first octet
1111	Octet string	Defined in first octet

in local and metropolitan areas. Furthermore, users can join, move and leave dynamically without any human intervention for network reconfiguration.

On the other hand, ZigBee is a low data rate radio technology, designed for low-throughput applications. Bluetooth standard [4], which has a similar protocol architecture, allows higher data rates (2 Mbps vs 250 kbps) and has a widespread diffusion in common handheld devices. So, in the Grid framework we propose, it was preferred for the interaction between users and the Grid itself.

Both protocols were extended with semantic-based service discovery capabilities. A middleware allowing the intercommunication between stacks at the application layer was also devised, enabling devices equipped with both wireless interfaces to act as bidirectional gateways interconnecting Bluetooth piconet and ZigBee PAN. Consequently, Bluetooth-only devices can be fully integrated within the mobile semantic Grid, with the ability to request and/or provide services to other Grid nodes. In other words, the mobile Semantic Grid architecture proposed here is based on a basic ZigBee infrastructure which provides the right pervasiveness and versatility to enable the underlying data propagation. Nevertheless low provided data rates as well as the scarce diffusion in common handheld devices are serious obstacles to the exploitation of that stack protocol for semantic-based interaction with users and make Bluetooth a viable alternative.

The proposed infrastructure supports the following key tasks:

- *Device discovery*: it is carried out by standard ZigBee protocol.
- *Data dissemination*: the periodic exchange of advertisements allows to distribute information about available services within the Grid. This process corresponds to the service creation step in the classic SOA model. Furthermore, the advertisement caching allows easier addressing of a specific provider for the service enactment step.
- *Service request*: this is the only task where the user is actively involved. The request is composed by an annotated description of desired service characteristics –expressed in DIG– and a list of contextual data-oriented attributes.

– *Service discovery*: in our Grid infrastructure, service procurement is performed through a fully automated, collaborative and dynamic discovery of service instances best matching the submitted request. The service discovery process involves:

  - the requester, which may be either a Bluetooth node or a ZigBee FFD (the former being more likely due to the larger diffusion of Bluetooth in current end user devices);
  - the Grid node that received the request (front end), which acts as a bidirectional gateway between the Bluetooth piconet originating the request and the ZigBee network servicing it;
  - service providers, which are other FFDs in the ZigBee network;
  - a reasoning engine, which may be located in the requester node itself, but in the most general case it is a separate entity.

Finally, the service enactment can take place, in a way depending on the particular service class. Each service class is modeled by a reference DIG ontology, formalizing domain knowledge by means of inter-related concepts and properties. Service instances can therefore be described by means of semantic-based annotations in DIG language, exploiting the domain vocabulary provided by the ontology. As said, it is identified by an OUUID code, which is used in the proposed semantic-based extension of both Bluetooth [5] and ZigBee to associate each annotated resource to the ontology it refers to.

Details on the developed protocol and packet structure for data dissemination and service discovery are provided hereafter. In the following subsection data dissemination protocol is described which supports the subsequent resource discovery whose details are provided in subsection 4.2. Finally subsection 4.3 outlines Bluetooth/ZigBee protocol interaction the above dissemination and discovery frameworks enable.

4.1 Data dissemination supporting service/ resource discovery

Resource advertisement. Each node of the Grid advertises managed resources periodically sending advertisements to the other network nodes. The structure of an advertisement frame is reported in Fig. 3.

In what follows the meaning of each packet field is explained:

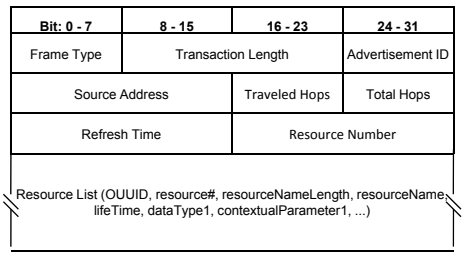

Figure 3 Advertisement PDU structure

– *Frame Type*: identifies the packet type.

– *Transaction Length*: the size (in byte) of the packet.

– *Advertisement ID*: a value which is increased when a node forwards an advertisement.

– *Source Address*: contains the `NWK_address` of the sender.

– *Traveled Hops*: the hops number a PDU (Protocol Data Unit) has gone across.

– *Total Hops*: the maximum hops number the the packet has to travel.

– *Resource Number*: the number of resources advertised with the packet.

For each advertised resource, the PDU contains the following payload data:

– *OUUID*: a numeric identifier of the reference ontology;

– *resource#*: resource identifier referred to both the node and the specified ontology;

– *resourceNameLength*: length of the textual label describing the resource;

– *resourceName*: resource label;

– *lifeTime*: resource expiration.

The protocol allows to associate up to five contextual parameters to each resource. They are used to provide supplementary information about the context where the resource is placed or about the resource itself. The *i-th* parameter is expressed by means the $< dataType_i, contextualParameter_i >$ pair.

Each node which "exposes" resources within the ad-hoc network has to periodically send advertisements to all the nodes in its radio range. They will retransmit the PDUs to their neighbors exploiting a selective mechanism. In particular, each node checks the *Source Address* and the *Advertisement ID* fields to verify whether it has already received a similar frame: if so, the frame is discarded. After an adjustment phase, the data dissemination process will reach a steady-state regime where all the nodes are aware of the "Grid content".

***Cache table* management**. Grid nodes store advertised information in order to perform the following discovery tasks. The prototype of a *Cache Table* entry is shown in the following Table 4, reporting information coming from the advertisement packet (in the first fields) related to a specified resource as well as complementary data which are stored in the other fields. In what follows we briefly sketch their meaning:

– *local*: if asserted, this flag indicates the resource is managed by the node;

– *lifetime*: remaining time interval to consider a resource as valid. This value is progressively decreased up to zero. In this case the associated cache entry is removed;

– *timestamp*: it corresponds either to the cache entry creation or to the entry reference time (read or update);

– *Traveled Hops*: traversed hops up to the current node;

– *digPointer*: it is a pointer exploited only for local resources which contains the memory address where the resource annotation is stored.

Notice that nodes with low or no storage availability (i.e., RFD) are not able to manage a *Cache Table*, hence they participate to the dissemination process in a passive fashion simply forwarding the advertisement PDUs.

Table 4 *Cache table* fields

FIELD NAME	SIZE (byte)
Advertisement ID	1
Source Address	2
OUUID	2
resource#	2
resourceNameLength	2
resourceName	variable
dataType_i	1
contextualParameter_i	variable
local	1
lifeTime	4
timestamp	4
Traveled Hops	1
digPointer	4

When an FFD node receives an advertisement, it scans the cache to verify if the advertised information is new or more recent. Notice that we can unambiguously identify a single resource by means of the triple <Source Address, OUUID, resource#>. Hence by checking for that triple we can find the presence of a resource annotation within the cache and, in case, we will update the related entry either if the advertisement PDU has made a shorter travel or the packet is more recent (i.e., it has a higher advertisement ID). In the latter case, the node updates *timestamp* and *lifeTime* values whereas, in the former case, in addition to the previous ones also the *Traveled Hops* field is updated.

The *lifeTime* value is imposed by the resource provider according to their mobility features. Its value is decreased by each node receiving an advertisement following the formula:

$$\text{lifetime} = \text{lifetime} \cdot [1 - s \cdot H(s)]$$

where $H(s)$ is the *Heaviside step function* and s is obtained by means of the relation:

$$s = \frac{\text{totalEntries} - \text{availableEntries}}{\text{totalEntries}} - 0.9$$

where $totalEntries$ is the number of cache entries whereas $availableEntries$ counts the available ones. In this way, the remaining time-to-live of a resource is decreased proportionally to the cache availability, so assigning a better storage possibility to more recent resources.

4.2 Semantic-based service/resource discovery

Request. In order to ask for a resource, a node has to submit a broadcast request to the Grid. Hence it builds and sends a packet

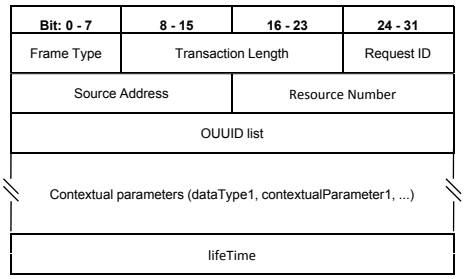

Figure 4 Request PDU structure

structured as depicted in Fig. 4.

The *Frame Type* value is fixed to `0x04`, the *Transaction Length* field contains the packet size (in byte), the *Request ID* unambiguously identifies each request submitted by a node, the *OUUID list* reports the OUUIDs referred to ontologies the requester is able to manage. Finally, the *Contextual parameters* field contains up to five $< key, value >$ pairs specifying some numeric constraints for the resource (in our implementation, for example, we use the geographic location of the requester) and the *lifeTime* value is the deadline for receiving a reply.

Reply. Each FFD in radio range with the requester is able to start a resource discovery phase so it acts as a *front-end* for the client node. When a front-end receives a resource request packet, first of all it searches for at least one OUUID contained in the request within its *Cache Table*. Furthermore, among the selected entries, it will extract only the ones satisfying the contextual parameters constraints. In particular, we can assume a resource is valid only if it is available within the desired time interval and if the distance from the provider is less than the one fixed by the user. Finally, the front-end node replies to the requester with a cache entry PDU as the one pictured in Fig. 5.

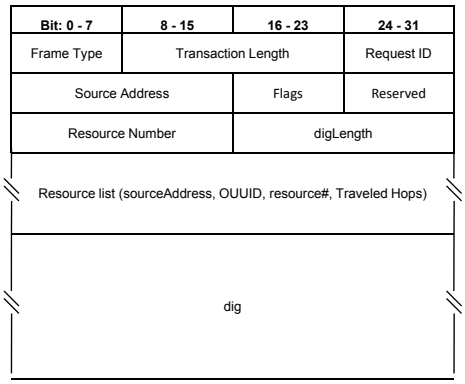

Figure 5 Cache entry PDU structure

In the field *Request ID* the identifier extracted by the request packet is pasted. Among remaining fields, the *Flags* octet contains two bits (so called `REQ` and `FWD`), respectively indicating if the packet transports the DIG description of the request and if the packet is not originating by the requester which has simply forwarded it (as explained later on). The field *Resource Number* enumerates the resources reported in the *Resource list* where, for each of them, the following information is stored: < sourceAddress, OUUID, resource# >, which is the triple identifying a Grid resource, and *Traveled Hops*.

The *dig* and *digLength* fields respectively contain the DIG re-

quest description and its size (in byte). They will be exploited in the following matchmaking phase. When the requester receives a cache entry PDU, it first extracts possible reasoning services (labeled by means of a specified OUUID) from the resource list, also taking the address of the respective provider. This node (from now on *reasoner node*) will receive in unicast the cache entry packet, properly enriched by the request DIG description as well as asserting the FRW and REQ flags. In this way it can retrieve the DIG description of each advertised resource as well as the semantic annotation of the request, in order to perform the matchmaking among them.

Semantic description request and reply. To obtain the semantic descriptions of the resources contained in the cache entry PDU, the reasoner node sends in unicast a packet as the one sketched in Fig. 6.

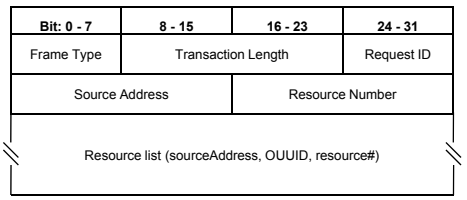

Figure 6 Structure of the PDU to require the semantic description of a specified resource

Source Address field contains the network address of the reasoner, whereas *Resource Number* indicates the number of resources whose description is required and each of them is labeled by means of the triple $<$ sourceAddress, OUUID, resource# $>$. The resource provider will reply with a packet as the one reported in Fig. 7, whose fields have the ordinary meaning.

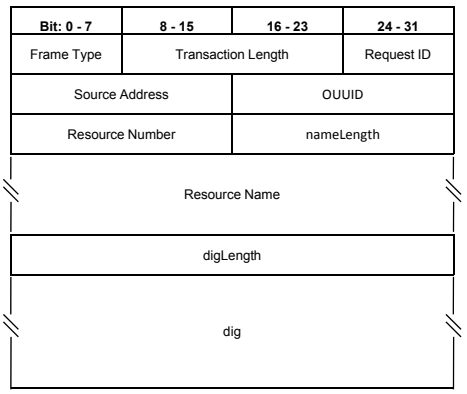

Figure 7 Structure of the PDU to provide the semantic description of a specified resource

When the reasoner node receives all the resource descriptions needed to perform the matchmaking, it verifies the compatibility between the request and each resource annotation. For compatible resources it will calculate the semantic distance by means of the *rankpotential* score [23]. This is a measure of the semantic distance between request and resource description. Finally, in order to take into account also the influence of contextual parameters, the *rankpotential* value will be weighted with a factor w which depends on the physical distance between requester and resource provider:

$$score = \frac{rankpotential}{w}$$

Notice that a fully compatible resource (*rankpotential*) is not influenced by the distance between requester and provider. The reasoner node selects the best result and replies to the requester by means of a PDU as the one pictured in Fig. 8. *rankpotential* field contains the $score$ value; the PDU also carries the semantic annotation within the *Resource Name* space.

4.3 Bluetooth/ZigBee protocol interaction

The user interacts with the Grid via the semantic-enhanced Bluetooth [5] exploiting her mobile device. She establishes a communication toward specific *gateway* nodes whose characteristics will be given in greater detail in what follows.

As obvious, gateways have to be "visible" to both user and Grid ZigBee nodes in a bi-directional communication. Basically, a gateway will be seen as a front-end by the "ZigBee side" of the Grid and as a piconet member by the other Bluetooth hosts. In a few words, in a gateway the Bluetooth and ZigBee application layers coexist; in particular a cross-protocol middleware running on the gateway provides the following functionalities:

- format conversion;
- cross-reference between resource databases;
- reasoner interface;
- integration between dissemination and discovery protocols.

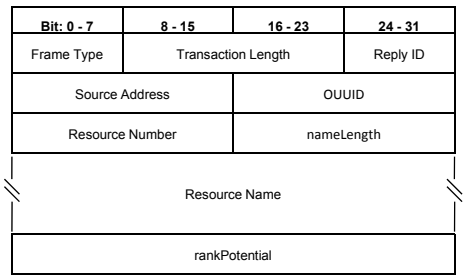

Figure 8 Structure of the reasoner reply PDU

Format conversion. The semantic enhancements of both Bluetooth and ZigBee adopt different data representations at the application layer. Hence the gateway middleware has to convert each $<$ key, value $>$ pair of the Bluetooth type dictionary in the corresponding ZigBee one and vice versa. For signed and unsigned integers as well as for string values, the format conversion is straightforward because it is only needed to exchange the input type with the output one, without intervention on the data field. Notice that some types are not defined in both protocols. In those cases, the conversion does not happen. Data are simply processed as raw strings, letting to the middleware the correct interpretation given the input type. The following Table 5 and Table 6 report the format conversion from Bluetooth to ZigBee and vice versa.

Cross-reference between resource databases. Databases containing resource descriptions managed by either ZigBee or Bluetooth nodes (namely *Cache table* and *Service Discovery Database* respectively) are kept distinct because they have a different structure. Hence, in gateways a direct mapping between

Table 5 Format conversion table from Bluetooth to ZigBee data types

Bluetooth type	ZigBee type
nil	nil
(un)signed integer (8/16 bit)	(un)signed integer (8/16 bit)
(un)signed integer (32/64/128 bit)	string
UUID	string
string	string
boolean	unsigned integer (8 bit)
Data Element Sequence	string
Data Element Alternative	string
URL	string
OUUID	OUUID
DIG string	DIG string

Table 6 Format conversion table from ZigBee to Bluetooth data types

ZigBee type	Bluetooth type
nil	nil
(un)signed integer (8/16 bit)	(un)signed integer (8/16 bit)
floating point (semi-precision)	string
absolute time	string
relative time	string
text string	string
byte string	string
OUUID	OUUID
DIG string	DIG string

them is required in order to properly manage references within the *Cache table* from the Bluetooth perspective and vice versa. Obviously, the general consistency and uniformity of information has to be guaranteed. As shown in Fig. 9, each entry in the former table must correspond to at most one entry in the latter and vice versa. In this way, a discovery procedure issued by a Grid node will be able to individuate resources belonging to Bluetooth nodes and vice versa.

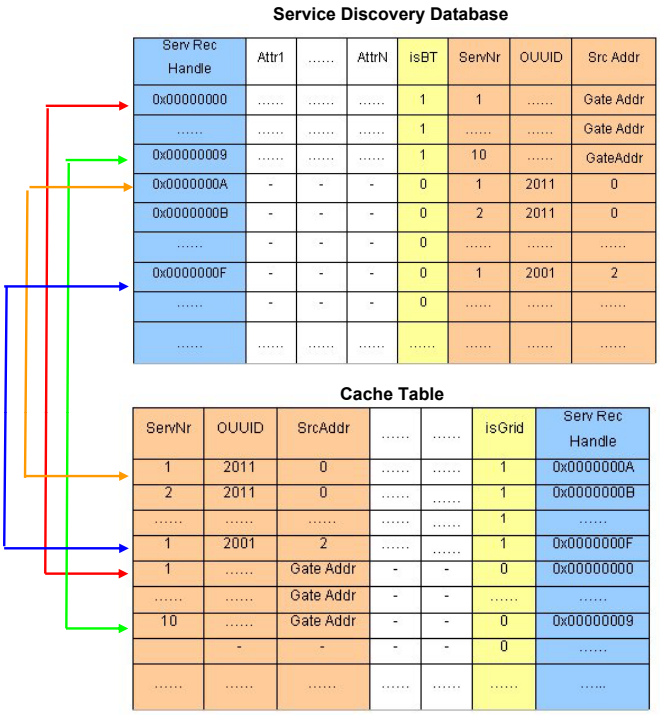

Figure 9 Correspondence between *Cache Table* of ZigBee FFD and *Service Discovery Database* in Bluetooth nodes

The correct correspondence between the resource tables is established by the middleware running on front end nodes. At startup, it inserts a new record in the *Cache Table* for each *Service Discovery Database* record, assigning a key value allowing unambiguous identification. The other fields are still unfilled except the *Service Record Handle* referring to the resource in the *Service Discovery Database*. The same procedure is repeated in a symmetric fashion for the "Bluetooth side" by adding to the *Service Discovery Database* all the entries of the *Cache Table*, so filling the key fields ($ServNr$, $OUUID$, $SrcAddr$).

In order to distinguish the resources belonging to the Grid from the ones hosted by Bluetooth devices, each cache entry contains the flag isGrid (1 bit) which will be asserted when a resource is managed by a ZigBee node whereas it will be deasserted for resources coming from Bluetooth piconets. This modification exploits the previously unused *local* field, so not troubling the original cache structure. Similarly, each tuple in the *Service Discovery Database* has the attribute $isBT$. Also this field does not alter the original structure of the Bluetooth database as it has been incorporated as attribute of the $ServiceRecord$[6].

Reasoner interface. The middleware featuring the cross-protocol sublayer of a front-end has to manage the bi-directional communication with the matchmaker when it is on board. It must enable the send/receive primitives allowing the basic interaction, properly building the input of the user and organizing the matchmaking results after computation.

Integration between dissemination and discovery protocols. In what follows, the role of the middleware in forwarding Bluetooth requests to Grid nodes and vice versa is clarified.

– *Replying to a Bluetooth Ontology Search Request*. When a Bluetooth node receives an Ontology Search Request PDU, it must check the required OUUIDs in its database. It will reply attaching found OUUIDs (none or more). The alignment between *Cache Table* and *Service Discovery Database* (as described above), allows to automatically take into account Grid resources in replies. Hence, actually the client request is satisfied without explicitly involving the middleware.

– *Replying to a Bluetooth Semantic Service Search Request*. When the middleware receives such a request, it extracts from the payload the reference information (i.e., max number of resources, OUUID, semantic annotation and contextual parameters). Those elements will be properly converted into the ZigBee data format and, if the gateway does not host a reasoner, it builds a cache entry packet attaching the request and sends it to a reasoner node which performs further processing. The best resource(s) satisfying the request will be retrieved. Note that the "ZigBee side" of the gateway acts in this case as requester w.r.t. Grid providers. Fig. 10 sketches the communication exchange in the above case.

  If the gateway hosts a reasoner, it will directly require resource annotations to the interested nodes. After the matchmaking procedure, the Semantic Service Reply PDU will be built for the user. Fig. 11 sketches the communication exchange in this case.

– *Replying to a ZigBee Service Request (hotspot on the gateway)*. In the FFD reply, also resources referred to the Bluetooth hosts will be considered thanks to the correspondence

[6] Following the original Bluetooth standard [4], various service attributes can be associated to each $ServiceRecord$.

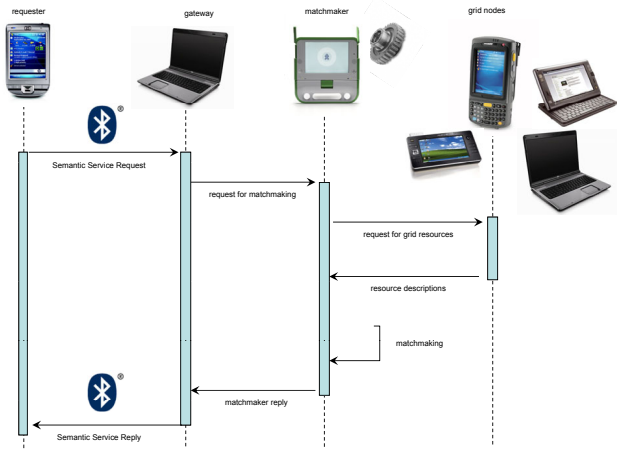

Figure 10 Possible interaction among Grid actors in case of matchmaker not hosted by the gateway

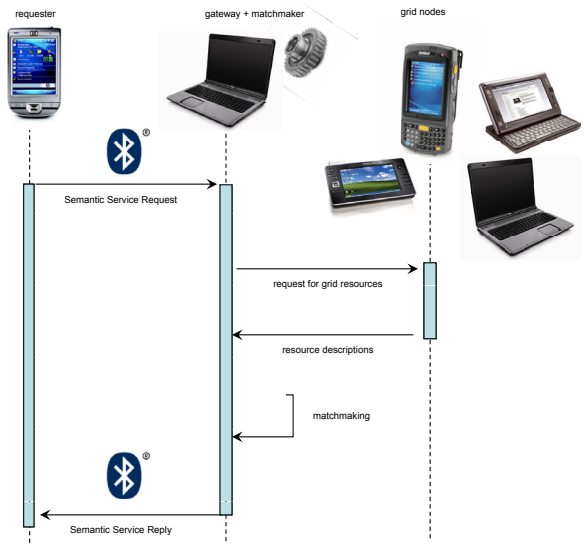

Figure 11 Possible interaction among Grid actors in case of matchmaker hosted by the gateway

between the resource registries of both Bluetooth and ZigBee in each node. The data dissemination procedure outlined above allows to spread resource descriptors and then to retrieve correspondent semantic annotations in the matchmaking phase.

– *Replying to a ZigBee Service Request (hotspot not on the gateway).* In this case, the "Bluetooth side" of the gateway does not manage a *Service Discovery Database*, hence it does not know information about possible piconet resources. So, the middleware running on the gateway extracts the reference OUUID from the request and then it builds an *Ontology Search Request* frame. The obtained sequence is passed to the "Bluetooth side" of the node which will forward the PDU to the hotspot. It will answer with an *Ontology Search Reply* containing retrieved OUUIDs which will be exploited by the ZigBee front-end to update the *Cache Table* of local services. Particularly, for each OUUID, a tuple is added which contains the OUUID value in the *Service Name* field. Finally, when the requester receives a reply, it can use either Grid resources (the interaction proceeds as detailed before) or piconet ones. In the latter case, the requester will send to the gateway a request for a discovery procedure based on the semantic-enhanced version of the Bluetooth SDP [5]. Fig. 12 sketches the communication exchange in the above case.

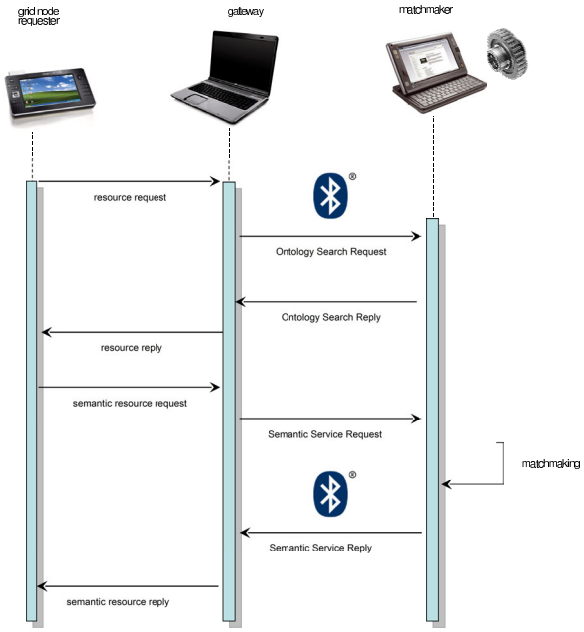

Figure 12 Possible interaction among Grid actors in case of hotspot not hosted by the gateway

5. ILLUSTRATIVE EXAMPLE

Let us consider an *Intelligent Transport System* as example application to illustrate the proposed Mobile Semantic Grid infrastructure. In that scenario, the goal is to provide an integrated service platform for multimodal information of users of public/private transport systems in an urban area.

Grid system architecture is based on a dynamic ad-hoc network comprising multiple device types, both fixed and mobile. Fixed nodes may include public access points, connectivity-endowed bus stops and traffic monitoring sensors deployed in the city area. Mobile nodes include buses equipped with GPS and mobile user devices (phones, palmtops or laptops).

Service annotation in semantic-based languages enables the advertisement and discovery of complex services. Let us consider the use case illustrated in Fig. 13, showing a user that wants to program her evening while walking on a street. As shown in the sequence diagram in Fig. 14, she uses her mobile phone to send a query to the Grid, like "I would like to watch an American action movie show starting in the next 3 hours at a movie theater with Dolby sound system, located within 2 km". In such a use case, the pedestrian acts as a requester in the mobile semantic Grid, while a node in her wireless range (e.g., a nearby bus stop) receives her request and takes the role of *front-end* toward the Grid infrastructure. This node will search the best matching services within the Grid; service discovery will take into account both the semantic-based descriptions and contextual parameters, which are dependent on the particular application. In the above case, movie show times, cinema locations and traffic condition data collected by sensors are used.

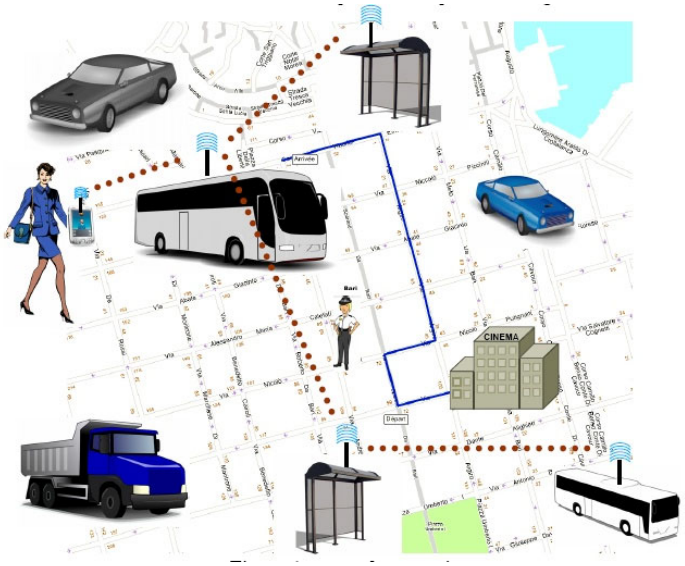

Figure 13 Illustrative application scenario.

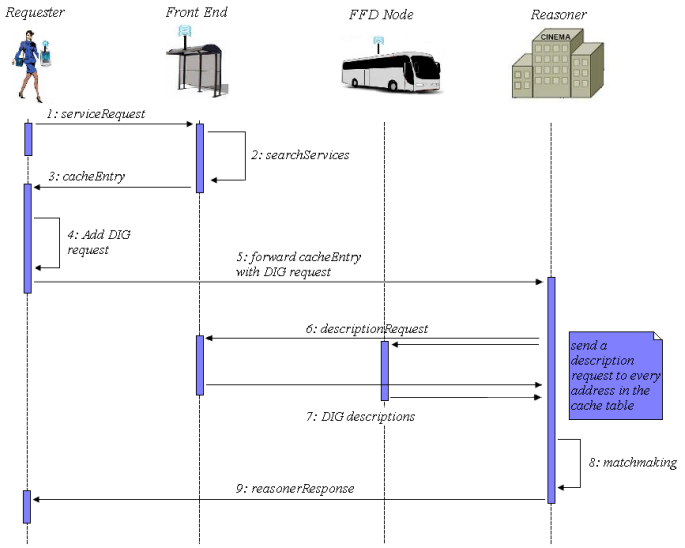

Figure 14 Sequence diagram for our example use case.

This application scenario was developed within the mobile semantic Grid infrastructure simulated using ns-2 network simulator[7]. OUUID ontology identifiers in the user request mark the desired service classes; in our example the movie show, traffic information and reasoning service classes are considered. The front end coordinates service discovery, that includes the following steps.

1. Movie shows beginning after the user-specified interval are discarded.

2. If user device is GPS-enabled, her location (coordinates) can be submitted together with the request; otherwise, the front end approximates the user location to its own. In the latter case the approximation error is limited by the wireless communication range of about 100 meters at most, for either ZigBee or Bluetooth technology.

[7] ns-2, the network simulator, available at http://www.isi.edu/nsnam

3. For each candidate service, the route to destination and its total length are computed.

4. For each candidate service, the ETA (Estimated Time of Arrival) is computed as $t_A = \sum_{i=1}^{n} t_i$ where $t_i = c_i d_i$, $i = 1, \ldots, n$ being d_i the length of the i^{th} route segment and c_i the corresponding *road congestion coefficient*. If the user added an OUUID for traffic info service to her request, the most up-to-date traffic information collected from sensors is used to compute c_i as $c_i = \frac{f_i}{F_i}$ where f_i is the current measured vehicle flow rate and F_i is the *maximum admissible flow* computed using the model provided in the *Highway Capacity Manual* [24]. If traffic service was not requested, a worst-case estimate is applied, with $c_i = 1 \, \forall i = 1, \ldots, n$

5. For each service j, contextual parameters are used to compute a *weigh*, defined as follows:

$$w_j = u(\frac{(T_{s,j} - T_0) - t_{A,j}}{(T_{s,j} - T_0)}) \frac{(T_{s,j} - T_0) - t_{A,j}}{(T_{s,j} - T_0)}$$

where $u(\cdot)$ is Heaviside step function, $T_{s,j}$ is the starting time of movie show j, T_0 is current time and $t_{A,j}$ is the ETA to the cinema. A larger time margin corresponds to a higher weight, while a service is discarded if the ETA is higher than the movie show start time.

6. Semantic matchmaking is executed by applying *rankPotential* algorithm [23], returning a match degree score f_j between the user request and every remaining candidate service.

7. A ranked list of overall scores $s_j = \frac{f_j}{w_j}$ is obtained. Cache entries of best matching services are returned to the requester node.

It should be observed that service/resource parameters are exploited in two ways. Firstly, they are used to discard services whose parameters do not satisfy user-imposed constraints. This pre-filtering stage avoids unnecessary processing by reasoner module, which has relatively high performance costs. Secondly, they allow a refinement of the semantic matchmaking outcome through the application-specific weighing function.

6. PERFORMANCE EVALUATION

Performance of the proposed framework was analyzed using ns-2 network simulator to assess its applicability to concrete scenarios. First of all, effectiveness of data dissemination was considered for semantically annotated advertisements in ZigBee-only mobile Grids. Secondly, performance of service/resource discovery in both ZigBee-only and hybrid Bluetooth/ZigBee mobile semantic Grids was evaluated. All tests were performed on an Acer Travelmate 3004 notebook with Intel Pentium M 760 CPU and 1 GB RAM. The following subsections describe in detail methods, results and discussion for each analysis carried out in the simulation campaign.

6.1 Data dissemination

Three Grid simulation scenarios were set up for the analysis of ZigBee data dissemination performance. They were populated with 12, 20 and 50 nodes respectively, in a 75 m × 75 m area. Each node could either be a ZigBee RFD or FFD. Node motion was disabled.

Each ZigBee FFD was given between 2 and 4 semantically annotated services to advertise. Effectiveness of data dissemination was assessed by measuring the 100% *settling time* of the system, defined as the time when dissemination of metadata is complete, i.e., when each of the n FFDs has received advertisement packets generated by $n-1$ other distinct FFDs. Settling time can thus be equivalently defined as the instant of reception of the R^{th} advertisement packet determining the insertion of a new *Cache Table* entry, with

$$R = n(n-1)$$

Generalizing, the $x\%$ *settling time* $T_{S(x)\%}$ can be defined as the instant of reception of the S^{th} advertisement packet determining the insertion of a new *Cache Table* entry, with

$$S(x) = \frac{Rx}{100}$$

For each scenario, the simulation was run 5 times and average values were computed. Results are reported in Table 7. The data dissemination protocol shows good performance in scenarios with up to 20 nodes, reaching stability within few seconds. On the other hand, in the 50-node scenario the system does not reach stability within the simulation time limit. This performance degradation is likely due to the blow-up of exchanged packets deriving from the simplistic flooding strategy adopted for packet forwarding.

Table 7 Data dissemination performance in three different scenarios

Nodes	Avg advert. length (B)	Received packets	$T_{S_{85\%}}$ (s)	$T_{S_{90\%}}$ (s)	$T_{S_{100\%}}$ (s)
12	117	658	0.881	0.984	1.142
20	118	2775	2.681	2.747	N/A
50	87	292673	673.546	N/A	N/A

Figures 15-17 show the average percent settling time $T_{S(x)\%}$ w.r.t. x for the three different scenarios. A least square error linear model fits well the simulation data in the first two scenarios, while system behavior is better approximated by an exponential function in the third case.

6.2 Service discovery

Performance of service/resource discovery was evaluated for all the mobile semantic Grid configurations allowed by the proposed framework, namely:

1. ZigBee-only Grid;
2. hybrid ZigBee/Bluetooth Grid with Bluetooth-side requester;
3. hybrid ZigBee/Bluetooth Grid with ZigBee-side requester.

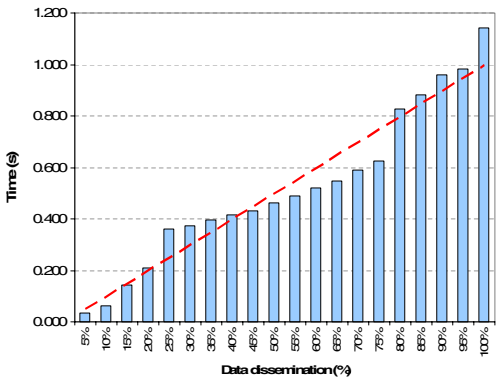

Figure 15 Average system percent settling times in 12-node data dissemination scenario. Linear regression function is plotted in red

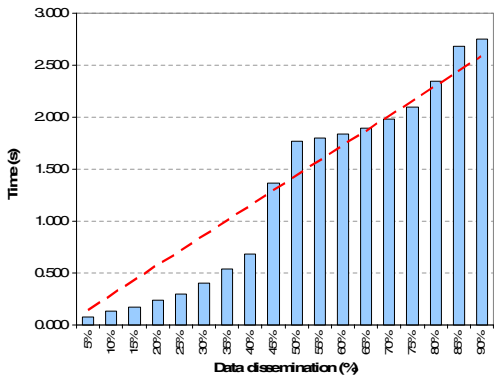

Figure 16 Average system percent settling times in 20-node data dissemination scenario. Linear regression function is plotted in red

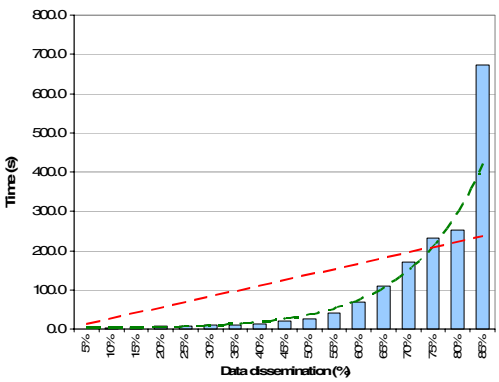

Figure 17 Average system percent settling times in 50-node data dissemination scenario. Linear and exponential best-fit functions are plotted in red and green respectively

1. ZigBee-only
The first mobile semantic Grid simulation scenario comprised 12 ZigBee nodes, with 7 FFDs and 5 RFDs. RFDs do not provide services and only act as radio relays. One FFD took the role of requester, the other 6 acted as front-ends. Each FFD had between 2 and 4 services, either standard or semantic-based. The average length of DIG annotations for semantic-based services was 587 ± 49 B, while the length of the request was 250 B. It was hypothesized that, among all Grid services, only three correspond to the OUUIDs issued in the request packet and are therefore selected as candidate for semantic-based matchmaking (as OUUIDs characterize service classes, basically a request of

three possible classes is a very common one in our experience-based idea).

Two different Grid behaviors and packet flows are possible according to the reasoner is on the requester node or not. As explained above, in the former case the requester collects and processes semantic-enhanced descriptions directly. The latter case is more complex, since the requester has to add the OUUID identifying the reasoning service to its request and service annotations must be sent to a front-end equipped with a reasoner for processing. In order to assess performance differences, *reasoner-on-requester* and *no-reasoner-on-requester* scenarios were simulated five times each in the above testbed, taking average times for every individual step of the service discovery process.

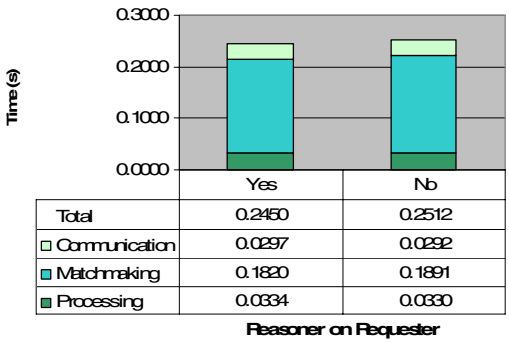

Figure 18 Service discovery performance in a ZigBee-only mobile semantic Grid

Results are shown in Fig. 18. The total value is the overall duration of the service discovery session, which is divided into three main components:

- request processing by the front-end, to preselect available services based on OUUID and context parameter values;
- reasoner matchmaking;
- communication, i.e., aggregate time for assembling and transmitting all packets.

Arguably, in our simulations the absence of reasoner on the requester node does not affect service discovery performance significantly. Further investigation could be useful to assess whether similar considerations are still valid or not for larger Grids, comprising hundreds of nodes and thousands of services.

Obtained absolute times are very small: service discovery was completed in a fraction of a second in all cases. This can be deemed as very satisfactory for a dynamic, collaborative and semantically rich service discovery paradigm. Semantic-based matchmaking is by far the longest sub-task. Within our reference testbed, approximately 0.06*s* were required to process each semantic service. This was expected, due to the inherent complexity of reasoning procedures.

It must be remarked that actual performance of request processing and matchmaking may vary significantly in real scenarios, depending on the particular application and available computational resources of involved Grid nodes. On the other hand, ns-2 yields realistic performance estimates for communication operations, regardless of the hardware used to run simulations. In the present case, communication took only 0.03*s* (about 12% of total time). This provides reliable evidence of the efficiency of the proposed service discovery protocol for ZigBee-based mobile semantic Grids.

2. Hybrid Grid with Bluetooth-side requester

The simulated scenario comprised 6 ZigBee nodes, 1 Bluetooth-ZigBee gateway and 1 Bluetooth requester. Among ZigBee nodes, there were 2 RFDs (with no services on board) and 4 FFDs, each one providing from 1 to 5 services. The average lengths of DIG annotations for semantic-based services in the ZigBee mesh and in the Bluetooth piconet were 598 ± 55 B and 573 ± 7 B, respectively. The size of the request was 250 B. Also the ZigBee side of the gateway is an FFD, while its Bluetooth side acts as Grid service provider for Bluetooth clients. It was hypothesized that Ontology Search has already been performed before service discovery, so that the requester directly includes the OUUID of the desired service class in its request.

As explained in Section 4.3, service discovery steps vary depending on the presence of a reasoning engine on the gateway. If not, the reasoning service will be provided by another node in the ZigBee side of the Grid. *Reasoner-on-gateway* and *no-reasoner-on-gateway* simulations were run five times each and average discovery durations were taken.

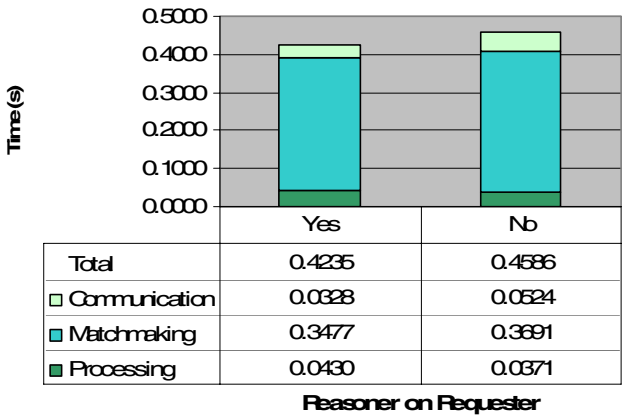

Figure 19 Service discovery performance in a hybrid ZigBee/Bluetooth mobile semantic Grid with Bluetooth-side requester

Fig. 19 shows obtained results. Like in the above case, the overall service discovery task is divided into request processing, reasoner matchmaking and communication. Communication time is 37% shorter in the reasoner-on-gateway case, since fewer packets are exchanged. Consequently, when the reasoning engine in not on the gateway, the incidence of request processing and semantic matchmaking on the overall discovery process decreases. These results shows the flexibility of the hybrid discovery protocol in adapting to a particular Grid architecture. The overall absolute times are quite acceptable and consistent with the ZigBee-only case.

3. Hybrid Grid with ZigBee-side requester

In this simulation we refer to the same mobile semantic Grid testbed used in the previous test. The difference is that now the service request originates from a ZigBee node and it is broadcast in the Grid. We considered the most general case, where neither the requester nor the gateway have on-board reasoning capabilities. In fact, if either node included an inference engine, the packet exchange would be similar to the above ZigBee-only case, with consequently reduced discovery execution time.

Three different cases were evaluated. In the first one (case *a*), the gateway plays the role of a service provider within the Bluetooth piconet. In the second case (*b*) the ZigBee node performs the first request with a particular OUUID, causing an Ontology Search phase to be performed in the piconet. Finally, third case (*c*) refers to subsequent requests of services with the same OUUID, so that Ontology Search is unnecessary. The related Grid protocol behavior is described in Section 4.3.

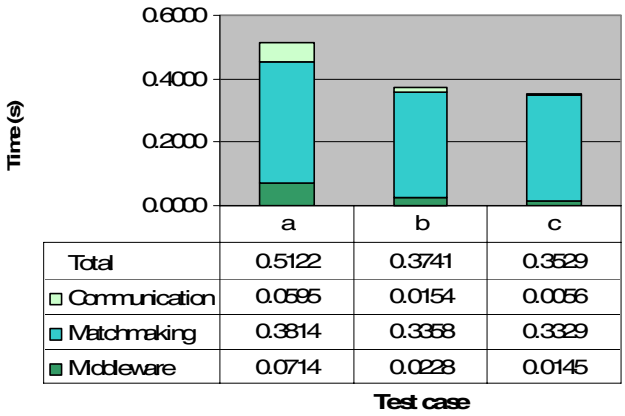

Figure 20 Service discovery performance in a hybrid ZigBee/Bluetooth mobile semantic Grid with ZigBee-side requester

Like in previous tests, each simulation was run five times and average performance was computed. Results are shown in Fig. 20 where performance measures for each sub-task were reported. Case (a) is confirmed as the most complex scenario, though absolute performance can be deemed as acceptable w.r.t. the number of both involved nodes and managed services. As expected, communications and middleware processing are shorter in case (c) than (b); the difference, however, has minimal practical impact, since semantic matchmaking is the dominating performance factor (89.7% and 94.3% of the overall transaction duration, respectively).

The issue of reasoning performance and scalability in semantic-based pervasive service discovery protocols was pointed out by [25]. They proposed optimizations to reduce on-line reasoning time, mainly off-line pre-classification of ontology concepts and concept encoding. Both strategies, though, are applicable only to matchmaking schemes based on pure subsumption (and therefore able to provide only binary yes/no answers). Hence, they are not viable options in our approach.

7. CONCLUSION AND FUTURE WORK

We have presented a cross-protocol ZigBee/Bluetooth grid infrastructure leveraging Knowledge Representation techniques and technologies for disseminating data and for performing discovery tasks issued by the user. By taking into account the heterogeneous nature of mobile semantic grids as well as the volatility of their structure, the proposed approach copes with an efficient and flexible resource discovery. Differently from discovery frameworks recalling Web approaches, the proposed one goes along with intrinsically evanescent nature of a mobile grid also granting an adequate QoS. A case study is presented together with experimental results on a prototype implementation to prove the validity of the proposal.

Future work will aim at a wider experimentation on a concrete mobile semantic grid integrating resource constrained mobile devices deployed in the field. Furthermore, a general optimization campaign of the proposed protocol will be performed in order to improve its performances. Finally, an extensive comparison with competing approaches (such as Jini or UPnP) will be taken into account to individuate development direction of protocol features.

Acknowledgments

The authors acknowledge partial support of Apulia Region Strategic Project PS 121.

REFERENCES

1. Newhouse, S., Mayer, A., Furmento, N., McGough, S., Stanton, J., Darlington, J.: Laying the Foundations for the Semantic Grid. Proceedings of AISB Workshop on AI and Grid Computing (2002)
2. ZigBee Alliance: ZigBee Specification. ZigBee Alliance. 053474r17 edn. (January 17 2008)
3. Storz, O., Friday, A., Davies, N.: Towards Ubiquitous Ubiquitous Computing: an alliance with the Grid. In: Ubisys 2003: System Support for Ubiquitous Computing Workshop. (2003)
4. Bluetooth: http://www.bluetooth.com
5. 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
6. Foster, I., Kesselman, C.: Computational Grids. The Grid: Blueprint for a New Computing Infrastructure (1999) 15–51
7. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications 15(3) (2001) 200
8. Foster, I.: What is the Grid? A Three Point Checklist. Grid Today 1(6) (2002) 32–36
9. Cannataro, M., Talia, D.: Towards the next-generation Grid: a pervasive environment for knowledge-based computing. Information Technology: Coding and Computing [Computers and Communications], 2003. Proceedings. ITCC 2003. International Conference on (2003) 437–441
10. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic Web. Scientific American 284(5) (2001) 28–37
11. De Roure, D., Jennings, N., Shadbolt, N.: The semantic grid: A future e-science infrastructure. Grid Computing-Making the Global Infrastructure a Reality (2003) 437–470
12. Lyytinen, K., Yoo, Y.: Issues and Challenges in Ubiquitous Computing. Communications of the ACM 45(12) (2002) 63–65
13. Waldburger, M., Stiller, B.: Toward the Mobile Grid: Service Provisioning in a Mobile Dynamic Virtual Organization. Computer Systems and Applications, 2006. IEEE International Conference on. (2006) 579–583
14. Chu, D., Humphrey, M.: Mobile OGSI .NET: Grid Computing on Mobile Devices. 2004 Grid Computing Workshop (associated with Supercomputing 2004)
15. Litke, A., Skoutas, D., Varvarigou, T.: Mobile grid computing: Changes and challenges of resource management in a mobile grid environment. Proc. of Practical Aspects of Knowledge Management (PAKM 2004), Austria (2004)
16. Ni, L., Zhu, Y., Ma, J., Li, M., Luo, Q., Liu, Y., Cheung, S., Yang, Q. In: Semantic Sensor Net: An Extensible Framework. Springer Berlin / Heidelberg (2005) 1144–1153

17. Hughes, D., Greenwood, P., Coulson, G., Blair, G.: GridStix: supporting flood prediction using embedded hardware and next generation grid middleware. In: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks. (2006)
18. Coulson, G., Hughes, D., Blair, G., Grace, P.: The Evolution of the GridStix Wireless Sensor Network Platform. In: Proceedings of the International Workshop on Sensor Network Engineering (IWSNE 08), co-located with International Conference of Distributed Computing in Sensor Systems (DCOSS 08). (2008)
19. Capo-Chichi, E., Guyennet, H., Friedt, J.: IEEE 802.15. 4 Performance on a Hierarchical Hybrid Sensor Network Platform. In: Proceedings of the 2009 Fifth International Conference on Networking and Services-Volume 00, IEEE Computer Society Washington, DC, USA (2009) 303–308
20. Lee, J., Su, Y., Shen, C.: A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi. In: Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE. (2007) 46–51
21. Mulyadi, I., Supriyanto, E., Safri, N., Satria, M.: Wireless Medical Interface Using ZigBee and Bluetooth Technology. In: Proceedings of the 2009 Third Asia International Conference on Modelling & Simulation-Volume 00, IEEE Computer Society (2009) 276–281
22. 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)
23. Colucci, S., Di Noia, T., Pinto, A., Ragone, A., Ruta, M., Tinelli, E.: A non-monotonic approach to semantic matchmaking and request refinement in emarketplaces. International Journal of Electronic Commerce 12(2) (2007)
24. Transportation Research Board: Highway Capacity Manual. National Research Council, Washington, DC (2000)
25. Mokhtar, S., Kaul, A., Georgantas, N., Issarny, V.: Efficient Semantic Service Discovery in Pervasive Computing Environments. Proceedings of the ACM/IFIP/USENIX 7th International Middleware Conference (2006)