Semantic-based Resource Discovery, Composition and Substitution in IEEE 802.11 Mobile Ad Hoc Networks

M. Ruta [1] G. Zacheo [1] L.A. Grieco [1] T. Di Noia [1] G. Boggia [1]
E. Tinelli [1,2] P. Camarda [1] E. Di Sciascio [1]

(1) Dipartimento di Elettrotecnica ed Elettronica
Politecnico di Bari
via E. Orabona 4, I-70125 Bari, Italy

(2) Dipartimento di Informatica
Università degli Studi di Bari
via E. Orabona 4, I-70125 Bari, Italy

Abstract

We present a general framework for resource discovery, composition and substitution in mobile ad-hoc networks (MANET), exploiting knowledge representation techniques. Key points of the proposed approach are: (1) reuse of discovery information at network layer in order to build a fully unified semantic-based discovery and routing framework; (2) use of semantic annotation of resources in order to perform the orchestration of elementary resources for building personalized services adopting a concept covering procedure, and to allow the automatic substitution of no more suitable/available components. Using ns-2 simulator, we evaluated performances of the proposed framework with reference to a disaster recovery scenario. In particular, the impact of the number of available services and active clients has been investigated in various mobility conditions and for several service covering threshold levels. Obtained results show that: (1) the proposed framework is highly scalable, given that its overall performance is improved by increasing the number of active clients; (2) the traffic load due to clients is negligible; (3) also for a very small number of available services very high hit ratios can be reached; (4) increasing the number of servers can lead to hit ratios very close to 100% at the expense of an increased traffic load. Finally, the effectiveness of cross-layer interaction between routing and resource discovery protocols has been also evaluated and discussed.

---

[1] Email: `[m.ruta, g.zacheo, a.grieco, t.dinoia, g.boggia, e.tinelli, camarda, disciascio]@poliba.it`
[2] Email: `tinelli@di.uniba.it`

1 Introduction

Mobile Ad-hoc NETworks (MANETs) are made by smart nodes, equipped with Wireless Network Interface Cards, able to automatically build up a multi-hop communication infrastructure when placed in the same area [36]. MANETs are typically employed in scenarios where it is not possible or convenient to deploy a wired network infrastructure, *e.g.*, military or disaster recovery operations. MANETs have to provide flexibility, fault tolerance, and ability of self-configuration. These objectives are classically pursued by carefully designing each layer of the protocol stack. Performances can be also improved using cross-layer protocol design techniques, as discussed in [16,35]. In this paper we investigate the issue of cross-layer design of routing and resource discovery protocols in MANETs, using a novel semantic based approach. The rationale is that resource discovery algorithms exploit data dissemination protocols as well as routing ones, to cope with node mobility. As a consequence, routing and resource discovery algorithms can be jointly exploited to minimize overhead.

In our proposal, given a semantic based user request and a set of available services/resources both described using a subset of the W3C standard Ontology Web Language OWL-DL [3], the framework allows us to carry out discovery and composition of mobile resource components covering as much as possible the needs of the requester. The proposed approach also copes with non exact matches. That is, those cases where a user request is partially satisfied or, in other words, it is not completely covered. Furthermore, when discovered resources are unsuitable to fulfill the request, an explanation of what is missing to fully satisfy the request can be provided. In order to increase the flexibility and the fault tolerance of the system, our composition protocol integrates the substitutability among resource components. The analysis of the compatibility between two services allows us to support the dynamic substitution in case of unavailability or unsuitability. As soon as a set of substitutes for a resource is identified, the orchestrator can get a candidate for the substitution automatically taking into account preconditions it wants and effects it produces.

In IEEE 802.11 Mobile Ad-hoc NETworks [1] a decentralized approach to Service Discovery is strongly recommended. A node should not be depending on some other one to advertise/register resources or services. Services/resources should be autonomously exposed and, at the same time, applications running on the other nodes should be able to discover them. Thus, resource discovery in ad-hoc environments is based on either broadcasting of requests or service advertisement propagation [4,9].

A widespread use of broadcasting mechanisms to advertise services/resources could result inefficient in terms of bandwidth usage and power consumption (both fundamental and precious resources in ad-hoc networks). Hence, in the proposed

---

[3] http://www.w3.org/TR/owl-features/

approach, only resource identifiers are advertised in broadcast throughout the network in order to unambiguously identify the location and the category of the service/resource. Knowing resource identifiers, if a node explicitly requires a given resource, it will download in unicast the semantically annotated description of that resource. With this approach, the advertisement flooding (due to the broadcasting mechanism) is reduced without affecting the correct location and use of a resource.

Using ns-2 [4] simulations, we have evaluated performances of the proposed framework with reference to a disaster recovery scenario, in which a rescue team has to accomplish complex resource discovery operations with the support of a IEEE 802.11 MANET. In particular, the impact of the number of available services and active clients has been investigated in many mobility conditions and for several service covering threshold levels. Results have clearly shown that: (1) the proposed framework is highly scalable, given that its overall performance is improved by increasing the number of active clients; (2) the traffic load due to clients is negligible; (3) also for a very small number of available services very high hit ratios can be reached; (4) increasing the number of servers can lead to hit ratios very close to 100% at the expense of an increased traffic load. Finally, the effectiveness of the cross-layer interaction between routing and resource discovery protocols has been also evaluated and discussed.

The remaining of this paper is organized as follows: in next Section we revise some significant related work; in Section 3 we describe our data dissemination protocol, and outline the interactions between routing and discovery protocols in Section 4. Sections 5 and 6 present composition and substitution features our framework provides. Section 7 proposes a case study illustrating the whole approach and presents a thorough performance evaluation in the case study context. Future work and conclusions close the paper.

2 Background

In this section we begin recalling basics of the knowledge representation formalisms and languages we adopt, together with inference services exploited in our approach, then move on to revise relevant related works.

2.1 Useful KR principles and tools

Description Logics (DLs) are a family of logic formalisms for Knowledge Representation [3], also known as Terminological languages, in a decidable fragment of First Order Logic. Basic syntax elements are: *concept* names, *role* names, and *individuals*. These basic elements can be combined using *constructors* to form concept and role *expressions*. Each DL has a different set of constructors. A constructor used in every DL is the one allowing the *conjunction* of concepts, usually

---

[4] ns-2, the network simulator– http://www.isi.edu/nsnam

denoted as $\sqcap$; some DL include also disjunction $\sqcup$ and complement $\neg$ to close concept expressions under boolean operations. Roles can be combined with concepts using *existential role quantification* and *universal role quantification*. Other constructs may involve counting, as *number restrictions*. Many other constructs can be defined, up to create n-ary relations [8], so increasing the expressiveness of the DL. Nevertheless, this usually leads to a growth in computational complexity of inference services [7]. Hence a trade-off is worthwhile. *OWL-DL* is based on DLs theoretical studies, it allows a great expressiveness keeping computational completeness and decidability.

In a DL framework, an ontology $\mathcal{T}$ is a set of axioms in the form: $A \sqsubseteq D$ or $A \equiv D$ where A is an atomic concept and D is generic $\mathcal{ALN}$ concept. Such ontologies are called Terminological Box (TBox). In particular, we call simple-TBox all those set of axioms such that if A appears in the left hand side (lhs) of a concept equivalence axioms then it cannot appear also in the lhs of any concept inclusion axiom.

In this paper we refer to the $\mathcal{ALN}$(Attributive Language with Unqualified Number Restrictions) subset of OWL-DL, which has polynomial computational complexity for "bushy" TBoxes [21] standard and non-standard inferences. Constructs of $\mathcal{ALN}$ DL are reported hereafter (see Table 1 for further details):

- $\top$, *universal concept*. All the objects in the domain.
- $\bot$, *bottom concept*. The empty set.
- A, *atomic concepts*. All the objects belonging to the set A.
- $\neg A$, *atomic negation*. All the objects not belonging to the set A.
- $C \sqcap D$, *intersection*. The objects belonging both to C and D.
- $\forall R.C$, *universal restriction*. All the objects participating in the R relation whose range are all the objects belonging to C.
- $\exists R$, *unqualified existential restriction*. There exists at least one object participating in the relation R.
- $(\geq\ n\ R)$ [5], $(\leq\ n\ R)$, $(=\ n\ R)$ [6], *unqualified number restrictions*. Respectively the minimum, the maximum and the exact number of objects participating in the relation R.

Given an ontology $\mathcal{T}$ and two generic concepts C and D, DL reasoners expose at least two basic standard reasoning services: concept **subsumption** and concept **satisfiability**. In a nutshell they can be defined as in the following:

concept subsumption. Check if C is more specific than (implies) D with respect to the information modeled in $\mathcal{T}$. In formulae we write $\mathcal{T} \models C \sqsubseteq D$.

concept satisifiability. Check if the information in C is not consistent with respect

---

[5] Notice that $\exists R$ is equivalent to $(\geq\ 1\ R)$

[6] We write $(=\ n\ R)$ for $(\geq\ n\ R) \sqcap (\leq\ n\ R)$

Table 1
Syntax and semantics of $\mathcal{ALN}$constructs and simple-TBoxes

name	syntax	semantics
top	$\top$	$\Delta^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
atomic negation	$\neg A$	$\Delta^{\mathcal{I}} \backslash A^{\mathcal{I}}$
universal quantification	$\forall R.C$	$\{d_1 \mid \forall d_2 : (d_1, d_2) \in R^{\mathcal{I}} \rightarrow\ d_2 \in C^{\mathcal{I}}\}$
number restrictions	$(\geq\ n\ R)$	$\{d_1 \mid \sharp\{d_2 \mid (d_1, d_2) \in R^{\mathcal{I}}\} \geq n\}$
	$(\leq\ n\ R)$	$\{d_1 \mid \sharp\{d_2 \mid (d_1, d_2) \in R^{\mathcal{I}}\} \leq n\}$
concept inclusion	$A \sqsubseteq D$	$A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept equivalence	$A \equiv D$	$A^{\mathcal{I}} = D^{\mathcal{I}}$

to the information modeled in $\mathcal{T}$. In formulae we write $\mathcal{T} \models C \sqsubseteq \bot$.

In a discovery process, subsumption and satisfiability may be powerful tools in case a Boolean answer is needed. Suppose you have your ontology $\mathcal{T}$ modeling information related to the services available in your MANET and services capabilities are described with respect to such ontology. In case you have a service description C and a request for service D, whenever $\mathcal{T} \models C \sqsubseteq D$ holds we know that service capabilities implies the ones requested by the user. In other words, since the description of the service C implies the description of the request D, it results that D is fully satisfied by C. On the other hand, $\mathcal{T} \models C \sqcap D \sqsubseteq \bot$ means that service capabilities are not compatible with the requested ones. This means that, due to some incompatibility in the description of both the service and the request, the former cannot be considered as a candidate to satisfy the latter.

However, in more advanced scenarios, Boolean answers do not provide satisfactory results. Often a result explanation is required. In [21] Concept Abduction Problem (CAP) $\langle \mathcal{L}, C, D, \mathcal{T} \rangle$ was introduced and defined as a non standard inference problem for DLs, to provide an explanation when subsumption does not hold. In a few words, given an ontology $\mathcal{T}$ and two concepts C and D in a DL $\mathcal{L}$, if $\mathcal{T} \models C \sqsubseteq D$ is `false` then we compute a concept H (for hypothesis) such that $\mathcal{T} \models C \sqcap H \sqsubseteq D$. That is, H represent a possible explanation to the reason why service capabilities do not imply requested ones or, in other words, H represents missing capabilities in the service C in order to completely satisfy a request D with respect to the information modeled in $\mathcal{T}$. Actually, given a CAP there is more than one valid solution. Some minimality criteria have to be defined. We refer the interested reader to [21] for further details.

Since H represents missing capabilities in C w.r.t. the ones request by D, one may think to ask a second service D' if it exposes such capabilities. Following this basic idea, relying on the definition of Concept Abduction Problem, in [30] the notion of Concept Covering is introduced and described.

Definition 1 *Let* D *be a concept,* $\mathcal{R} = \{S_1, S_2, ..., S_k\}$ *be a set of concepts, and* $\mathcal{T}$ *be a set of axioms, all in a DL* $\mathcal{L}$*, where* D *and* $S_1, \ldots, S_k$ *are satisfiable in* $\mathcal{T}$*. Let also* $\prec_{\mathcal{T}}$ *be an order relation over* $\mathcal{L}$ *taking into account the ontology* $\mathcal{T}$*. The* Concept Covering Problem *(CCoP) for* $\mathcal{V}=\langle\mathcal{L}, \mathcal{R}, D, \mathcal{T}\rangle$ *is finding a pair* $\langle\mathcal{R}_c, H\rangle$ *such that*

(i) $\mathcal{R}_c \subseteq \mathcal{R}$*, and the conjunction of concepts in* $\mathcal{R}_c$*,* $C = \sqcap_{S\in\mathcal{R}_c} S$ *is satisfiable in* $\mathcal{T}$*;*

(ii) $H \in SOL(\langle\mathcal{L}, C, D, \mathcal{T}\rangle)$ *(solution of a CAP* $\langle\mathcal{L}, C, D, \mathcal{T}\rangle$*), and* $\mathcal{T} \not\models H \sqsubseteq D$.

We call $\langle\mathcal{R}_c, H\rangle$ *a* solution *for* $\mathcal{V}$*, and say that* $\mathcal{R}_c$ (partially) covers D*. Finally, we denote* $SOLCCoP(\mathcal{V})$ *the set of all solutions to a CCoP* $\mathcal{V}$*.*

Intuitively, $\mathcal{R}_c$ is the set of concepts that partially cover D w.r.t. $\mathcal{T}$, while the abduced concept H covers what is still in D and is not covered by C. There can be several solutions for a single CCoP, depending also on the strategy adopted for choosing concepts in $\mathcal{R}_c$. However, observe that –differently from the standard Set Covering Problem– a complete cover may not exist. Hence, minimizing the cardinality of $\mathcal{R}_c$ is not the aim of a CCoP; the aim is maximizing the covering, hence minimizing H.

Hereafter, we will formalize examples by adopting DL syntax instead of OWL-DL or DIG [5] for compactness, whereas in our prototypes DIG is exploited because it is less verbose (a good feature in mobile ad hoc contexts) with respect to OWL-DL.

2.2 Related Work

There is a widespread request for improved discovery features in wireless contexts and in particular in mobile ad-hoc networks. In [2] w.r.t. Bluetooth piconets, the need for discovery mechanisms more powerful than those of the original standard, inadequate for modern ubiquitous scenarios, was pointed out for the first time. In that paper, also a proposal for ranking approximate matches in the absence of exact ones was discussed, but no formal framework was identified.

In recent years, dynamic distributed systems have been developed adopting various technologies and for different purposes. Existing service discovery systems usually do not support a well defined common ontology infrastructure. Architectures like Jini allow to “capture” the ontology shared by some services. For this purpose, mechanisms like Java classes are adopted. In spite of their usefulness, they are hardly adaptable to several different discovery scenarios. This limitation, as admitted in [15] and in [12], is due to the lack of an efficient ontology support in the Jini framework. In [12] it is assumed that a client request is described by means of the same ontology a service uses for describing itself. This assumption is fundamental because it restricts the discovery only to services classified in the same manner, but there is no mention to the technique to reach such an objective. In this paper we propose a simple method for ontology class agreement prior to service discovery. The preliminary ontology matching grants a quick restriction of the available services only to those semantically suitable.

In [9] a Group-based Service Discovery (GSD) is presented. Different services are classified in groups according to the class/subclass hierarchy present in DAML. An advertising mechanism is adopted to spread resource descriptions in the MANET, where each advertisement includes a list of some group of services that a node has seen in its neighborhood. Caching of advertisements is foreseen to reduce packet flooding. No composition features are presented and resource classification is based on a basic taxonomy, which is a reductive hypothesis for the dynamic environment an ad-hoc network is expected to be. Furthermore, though the proposed cache management policy reduces the advertisement flooding, it is presumably quite demanding for devices with reduced computational capabilities and battery power.

Chakraborty et al. in [11] proposed a routing and session management protocol for ad-hoc networks integrated in the service discovery infrastructure. The core-idea is in re-using –at the routing layer– the path determined by the propagation in the MANET of both requests and advertisements. This feature aims at increasing packet delivery ratio and decreasing delays w.r.t. traditional frameworks (where the discovery layer is separated from the routing one). Since the new integrated protocol is based on the previous GSD, all the shortcomings of it are inevitably inherited. Furthermore, also in this case, composition of mobile services is not contemplated. Finally –as admitted by the authors– all issues related to the lack of modularity of the protocol and to its difficult upgrading, remain unsolved.

Service composition in ubiquitous environments is discussed in [14], where a service composition protocol based on mobile broker agents is presented. The composition of mobile services starts with the election of a broker node, which exploits the previous GSD infrastructure to locate mobile resources. It manages the integration and the execution of composite requests. The "broker arbitration phase" can be expected to be quite expensive for an ad-hoc network, as it requires determining characteristics of each candidate broker, thus increasing the network burden. Also, protocol transactions appear computationally demanding to be executed on simple mobile hosts like cellular phones. The approach does not deal with non-exact covering nor with substitutability issues.

In [13] a distributed system for dynamic service composition in mobile scenarios is presented. The framework is basically the same of [14] and a central role is again assigned to a distributed broker, which can be executed in any node of the network. Such broker is chosen adopting an election mechanism and it manages the composition. The proposed prototype is implemented within a Bluetooth piconet and it implements a substitution feature. The broker gradually increases its search radius (repeating the arbitration phase and then transferring the control to other brokers). The final purpose is the discovery of all service components of the MANET, returning a failure message if some of them is lacking. A solution to the problem of approximate covering when some of the requested resources are missing is not considered. Finally, the exploited fault tolerance system (by using check points) appears very expensive in terms of both cache and computational burden for mobile devices.

A further extension of the previous paper is reported in [10]. Service composition is accomplished taking into account various factors including mobility, variability of service topology as well as the resources actually available on devices within the piconet. Also in this paper the composition is based on a distributed brokerage mechanism using the service discovery paradigm presented in [9]. The "Composition Manager arbitration" implies that each node exactly knows its state (in terms of battery power consumption, remaining computational capabilities, cache occupancy, local resources availability). These information are used for election. Nevertheless, to maintain them, further overhead is introduced in each node. In fact, the high volatility of the environment as well as the intrinsic nature of mobile devices, require frequent updates of each state. Also in this approach no formal solutions are proposed for dynamic substitution of failed or unreachable services. In the case of an incomplete retrieval of component resources, the discovery is newly performed and composition is restarted. This procedure can be repeated until a time-out expires. Hence a significant overload for the network and the subsequent reduction of the bandwidth may result. The authors do not propose approaches neither for managing partial covering of the request nor for providing to the requester information about left uncovered part.

In [20] the compatibility between e-Services is analyzed for building the support to the dynamic substitution of failed or modified services and with the aim of granting an adequate result to the final users. A virtual district is the applicative scenario. Each e-Service interface is studied in both syntactic and semantic aspects whereas the behavior of services are modeled w.r.t. preconditions and effects. The proposed model requires an intrusive intervention by the user, i.e., it is only partially automated. Furthermore, the framework is expressly thought for wired contexts and no hypotheses are made about a possible extension of it to mobile environments.

Ponnekanti et al. in [33] present an application to solve the interoperability among services issue in ubiquitous computing. In this paper the authors aim to show how applications can interact with services even when globally unique interfaces are not provided. Nevertheless the proposed framework requires a substantial human intervention. No proposals are presented for managing partial dissimilarities among all the needed services within the environment. A mechanism of software stubs and adapters is outlined to solve the adaptation problem for two or more different service interfaces, but basically this approach appears not adequate for the specific purposes of a MANET.

In [23] the use of a rule engine to dynamically determine a near-optimal provider for each requested service is implemented. This multi-provider platform reduces dependencies; it increases geographic and functional coverage and it allows load balancing. The choice of a provider is essentially based on performances, reliability and coverage offered by it. This approach is interesting in order to realize the substitutability among different providers, but it is unlikely applicable to a pervasive environment where each node can provide multiple different services, resources are limited and similarity classes have to be built w.r.t. single services rather than w.r.t. complex providers.

The work in [26] introduces a framework for resource retrieval based on a set of self-organized discovery agents which manage a directory service where resources can be searched out by using a hash indexing. In addition, the proposed system enables a dynamic selection of the best service provider according to supplied QoS. The agents subdivide the network into domains and collect intra/inter domain QoS information to choose appropriate providers. Nevertheless the proposed framework is unfortunately based on a purely text matching discovery. Hence all the drawbacks of such an approach are inevitably inherited.

In [25], the notion of contextual attribute is defined to extract and then to manage environmental information during the resource discovery. As devised in the paper, a contextual attribute could include network or client condition, service quality parameters as well as other specific variables. Such attributes are dynamically determined and evaluated by lookup services and they contribute to refine the basic discovery (performed by means of static attributes). Although this is an improvement w.r.t. syntactic resource discovery, a complete and formal framework to support context awareness is still to come. An articulate description of the context is unavailable, and state-of-the-art matchmaking techniques are not introduced.

The compatibility among web services and the subsequent substitutability are also discussed in [6]. The authors focus on static properties of a service. In the proposed approach, a description of the service behavior –based on process-algebraic or automata formalisms– is provided with the purpose of detecting possible incompatibilities in the interaction among web services. Moreover, by checking for compatibility among services a method for service substitutability is proposed. The authors introduce some techniques to replace a web service with another one in a simple and direct fashion. Differently from the approach in [33], here no formal solutions are proposed to restore compatibility between two web services originally not compatible.

A platform to both manage cooperation among processes and compose e-Services, is outlined in [28]. The definition of compatibility between e-Services by taking into account their external behavior is introduced. E-Services specifications and components are separately presented. The first ones are described according to a UML-like model, whereas the others are developed on the basis of a technological component model which allows to deploy such e-Services. The proposed platform is enriched by a special repository where process schemas, e-Service specifications and instance information are stored. Apart from this theoretical approach to the composition and reconfiguration of services, no formal proposals about incomplete covering and approximate orchestrations are formulated in the paper.

3 Data Dissemination Framework

The framework we propose exploits a controlled dissemination of advertisements containing resource locations followed by an "on demand" download of resource descriptions. In other words, only information strictly required for the unambiguous identification of a resource are spread within the network, and the semantically annotated description of the resource itself will be sent in unicast only to nodes explicitly interested in it, so that they can proceed with further semantic-based matchmaking and composition phases.

An efficient data dissemination protocol is fundamental to support resource retrieval in a MANET. The Resource Discovery Protocol we implement is based on an advertisement mechanism. Resource providers periodically send *advertisement packets* (containing –among other fields– an unambiguous resource identifier) also specifying the maximum number of hops for the advertisement travel (`MAX_ADV_DIAMETER`). During their travel, the advertisements are forwarded using MAC broadcasts and can be stored in the cache memories of the nodes they go through.

If descriptions are still insufficient to cover the request, the node sends a *solicit PDU* with a specified maximum travel diameter (`MAX_REQ_DIAMETER`) in order to get new resource locators. A node receiving a solicit replies (in unicast) providing cache table entries matching parameters contained within the solicit frame. If a node does not manage any information satisfying the solicit, it will reply with a "no matches" message.

A node which is starting a resource composition, will usually attempt to cover the request by using resource descriptions stored within its own cache memory. If some semantically annotated description is missing, it can be retrieved in unicast using specific *demand PDUs*. During their travel, replies to the demand and solicit PDUs are used to update the cache memory of forwarding nodes.

After receiving required information from hosts in its search range, the requester newly attempts to cover the resource composition, and if the covering level is still under a given threshold, the node could try to forward a new solicit request by increasing the maximum search diameter. So these steps can be repeated until either the threshold or the maximum search diameter is reached. The covering level represents the degree of semantic correspondence (in percentage) between the request and the composite service/resource. When the composition starts, the covering level is set to 0% and the overall request is uncovered. During the further orchestration phases, with the involvement of more and more component resources, the covering level increases and conversely the uncovered part of the request is reduced.

In order to decrease the protocol overhead, routing, and discovery paradigms are integrated, so that sequence numbers of the routing protocol are used by the service discovery in order to verify if its own information about a resource is up to date. Moreover, the routing protocol updates its tables using the packets produced by the discovery protocol, even if a Route Discovery session has not been started. These features will be thoroughly analyzed in Section 4.

As hinted before, we hypothesize each resource in the mobile ad-hoc network is unambiguously identified. This is obtained by means of the triple [*SOURCE ADDRESS*, *OUUID*, *RESOURCE ID*], where the first value is the IP address of a node hosting the resource, the second one stands for *Ontology Universally Unique IDentifier* and marks the specific reference ontology the resource is associated with, the last one is a value to distinguish different services/resources coming from the same host node and referred to the same ontology. The previous triple labels in a different fashion different descriptions referred to the same ontology.

A simple example will clarify this feature. Let us suppose we have three nodes providing resources: *host#1* manages the resource *X* referred to the ontology *i*, *host#2* manages both the resource *Y* and the resource *U* both referred to the same ontology *j* in addition to the resource *W* referred to the ontology *k*, and the *host#3* manages the resource *Z* also referred to the ontology *i*. Corresponding descriptions will be classified respectively as:

- (host#1_address, i_ontology-ID, resource-ID=0)
- (host#2_address, j_ontology-ID, resource-ID=0)
- (host#2_address, j_ontology-ID, resource-ID=1)
- (host#2_address, k_ontology-ID, resource-ID=0)
- (host#3_address, i_ontology-ID, resource-ID=0)

Recall that, as in [39], we postulate the existence of a unique identifier (OUUID) for each ontology, thus allowing an unambiguous identification in the whole Semantic based Web. In the following subsections we will give a closer look to advertisement, request, and solicit PDUs as well as to the cache content organization of nodes.

3.1 Advertisement PDU

In Fig. 1 the structure of an advertisement PDU at the application layer is sketched. All resources of a node are advertised by means of a unique advertisement PDU and then the size of the packet increases proportionally with the number of resources hosted by a node.

Hereafter, we will analyze PDU fields.

- `TYPE`: the kind of PDU (see Table 2).

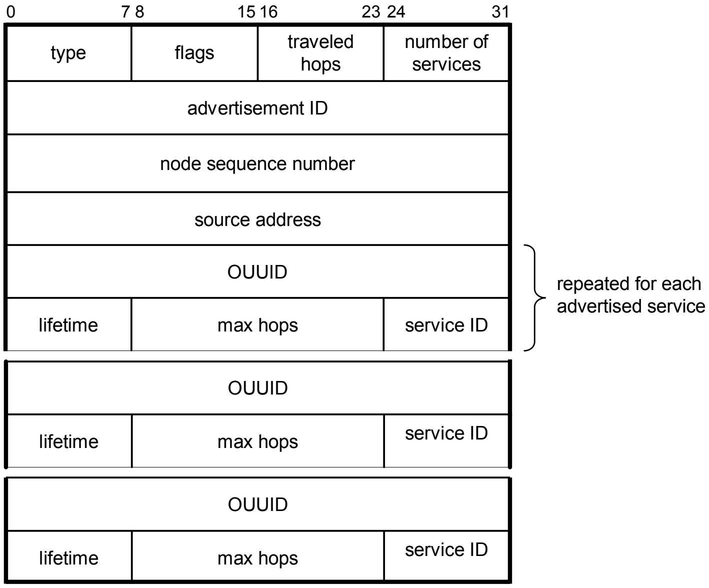


Fig. 1. The advertisement PDU structure.

- `FLAGS`: status flags used to distinguish the kind of transmission (uni or broadcast); the remaining flags are reserved for future purposes.
- `TRAVELED HOPS`: the number of hops already traversed by the packet. The server node sets this value to 1 and it is increased every time a node forwards the packet.
- `NUMBER OF SERVICES`: how many resources are hosted by the node.
- `ADVERTISEMENT ID`: the server's sequence number.
- `NODE SEQUENCE NUMBER`: the sequence number of the node forwarding the packet. If the packet has been sent by the server node this field value is equal to the previous one.
- `SOURCE ADDRESS`: the IP address of the provider.
- `SERVICE PARAMETERS`: a composite, variable length field depending on the number of advertised resource. In particular, it contains the OUUID value, the remaining life-time of a resource, the maximum hops number for the advertisement travel, and an identifier for the single service/resource. As each resource description is referred to a specific ontology, it is initially used to select suitable resources.

Every `DEFAULT_RTIME` milliseconds, a mobile node hosting a resource broadcasts an advertisement (values of exploited constants are reported in Table 3). Nearby nodes forward the packet by broadcasting it to their neighbors; as a consequence, the server nodes listen to the echo of the advertisement packet it originally transmitted. Thus, a server node can obtain a confirmation of the presence of other

Table 2
Various PDU types used in the proposed framework

TYPE	BIT SET	KIND OF PACKET
A	0	Advertisement
B	1	Cache entry
C	2	Solicit
D	3	Demand
E..L	4..7	reserved

Table 3
Constant values used in the framework

NAME	MEANING	VALUE
`DEFAULT_RTIME`	Time interval between two consecutive advertisement packet transmissions	2000 ms
`POLLING_TIME`	Time a server node waits for the echo of the advertisements	500 ms
`MAX_ADV_JITTER`	Maximum value for random time waited when forwarding advertisement packets	40 ms
`ONE_HOP_WAIT`	Timer set by a node after sending a solicit packet waiting for cache contents reception	2000ms
`HOP_TRAVERSAL_TIME`	Time a node needs to process and forward a solicit packet sent by a neighbor	50ms
`ACK_RTT`	Timer set by a node waiting for ack after a solicit has been sent	50 ms
`DISCOVERY_DIAMETER`	Current search diameter (in hops) during discovery phase	4
`MAX_RETRIES`	Maximum number of retransmissions before a server node assumes there are no neighbors	5

nodes in its neighborhood. If the server node does not receive any echo within `POLLING_TIME` milliseconds (less than `DEFAULT_RTIME`), it will retransmit the advertisement, assuming that a collision or a transmission error has occurred. After `MAX_RETRIES` retries it can be assumed there are no neighbors, so the transmission of the advertisement can be scheduled for a longer timeout in order to reduce power consumption.

When a node receives an advertisement, it extracts the routing information, as thoroughly explained in Section 4. Then, information about the resources is processed. If the service was previously unknown, a new entry may be created in the cache; otherwise, the node, before updating stored data, verifies if the information received is more recent or has ran across a shorter path than the existing one. If the cache is updated and the maximum advertisement diameter has not been reached the advertisement is forwarded. Otherwise, the whole packet will be silently discarded. This simple mechanism grants that each mobile node in the network sends the same advertisement at most once. Furthermore, in order to reduce the collision probability (recall that MAC 802.11 protocol does not provide any acknowledgment frame for broadcasting transmission), each host waits a random time t before transmitting, with $t \in [0, \texttt{MAX\_ADV\_JITTER}]$.

3.2 Demand PDU

A node starting a service composition, initially checks for possible compatible entries within its cache table and, in that case, it retrieves in unicast the corresponding semantically annotated descriptions, which are requested sending specific demand PDUs (see Fig. 2) to provider nodes. In what follows the meaning of introduced PDU fields is summarized.

- `TYPE`: it is set to 3.
- `FLAGS`: similar to the corresponding field of the advertisement PDU.
- `TRAVELED HOPS`: number of hops the frame has already gone across.
- `LAST HOP SEQUENCE NUMBER`: the sequence number of the last node processing the request.
- `DESTINATION SEQUENCE NUMBER`: the sequence number of the destination node.
- `DESTINATION ADDRESS`: the address of the last node processing the request.
- `PROVIDER ADDRESS`: the address of the destination node.
- `OUUID`: the ontology unique identifier.
- `NUMBER OF REQUESTS`: the number of service descriptions requested to the server node by the client node.
- `DATA`: the size of this field depends on the number of requests; it contains the identifiers of the services whose descriptions the client is requesting. Each service ID, as in the advertisement PDU, is one byte long.

A client requires all missing resource descriptions at the same time and then waits for replies up to $\texttt{ONE\_HOP\_WAIT} \cdot \texttt{max\_distance}$ seconds (where `max_distance` indicates the maximum hop number between requester node and each provider). When this time has expired or all requested PDUs have been received, the requester starts the composition.

3.3 Solicit PDU

If the result of the resource composition procedure is still under a specified threshold, a node should require further descriptions in order to attempt a new composition. Thus, it will transmit a solicit packet (see Fig. 3) to nearby nodes with a mechanism basically similar to the advertising one. The meaning of its fields are

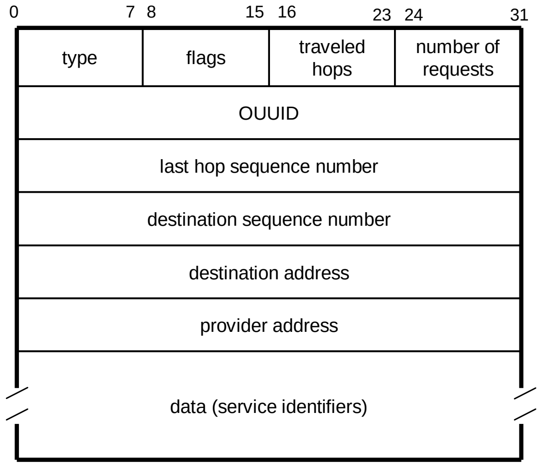


Fig. 2. The demand PDU structure.

hereafter summarized.

- `TYPE`: it is set to 2.
- `FLAGS`: it maintains the ordinary structure and functionality.
- `TRAVELED HOPS`: hops the packet has already gone across.
- `TOTAL HOPS`: total hops the PDU has to skip. Together with the *TRAVELED HOPS* field, it regulates the frame travel.
- `REQUEST ID`: unambiguously labels the PDU in order to distinguish different solicit requests.

The other PDU fields are basically identical to the ones described for the demand-PDU

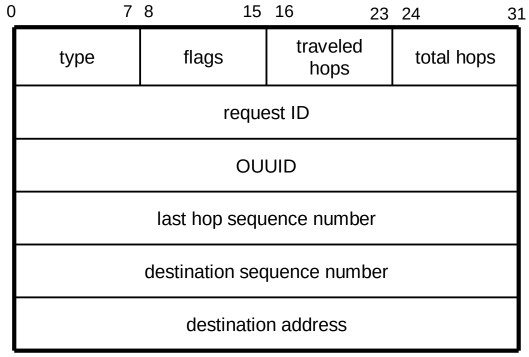


Fig. 3. The solicit PDU structure.

A node generating a solicit packet waits for an acknowledgment from each neighbor for `ACK_RTT` seconds. In this way, the requester can exactly know the number of neighbors. Therefore, before starting the composition phase or requesting new resource descriptions, it will wait for all the expected cache content PDUs for a time t_w (see Table 3) defined as:

$$t_w = \texttt{ONE\_HOP\_WAIT} + \texttt{HOP\_TRAVERSAL\_TIME} \cdot (\texttt{current\_hops} - 1) \quad (1)$$

where `current_hops` is the hops number the solicit has still to traverse to reach the `DISCOVERY_DIAMETER`.

This procedure is recursively repeated by all forwarding nodes along the solicit path up to the `DISCOVERY_DIAMETER`. Each node with a distance from the requester greater than `DISCOVERY_DIAMETER`, after receiving a solicit PDU, replies in unicast with a cache content PDU toward the node the solicit cames from. Nodes receiving a cache content PDU update their own cache and recursively send back their own cache content PDU, till the original requester node receives the cache content PDUs it needs.

3.4 Receiving/transmitting a cache content PDU – Cache table management

Each node manages a cache table where it stores characteristics of both resources it owns and resources it has "seen" in the network. Fig. 4 shows the structure of a typical entry. Here, we make explicit the content of each field.

- `Source address`: the address of the resource provider.
- `Size`: the description size (in byte).
- `OUUID`: a numeric identifier for the specific ontology.
- `Lifetime`: the remaining time to live of a service/resource.
- `Timestamp`: it marks the last reference to the entry (read/write). That is when a new resource is stored within the cache or when an existing one is invoked, this field is updated.
- `Traveled hops`: distance (in terms of hops number) between the provider and the cache holder.
- `Sequence number`: it is referred to the last resource provider.
- `Resource ID`: the identifier of a specific resource.
- `Resource description`: the semantically annotated description of a resource. It will have a variable length, but in some cases there could be a pointer to a text file containing its DIG description.

An entry can be added to the cache table whenever the node receives an advertisement or a cache content frame arrives.

A cache content packet (see Fig. 5) carries various information. The frame has a variable length according to the number of resource handles the PDU transports. Hence the cache update could involve more records. PDU fields are outlined in what follows.

- `TYPE`: it is set to 1.
- `FLAGS`: it maintains the ordinary meaning.

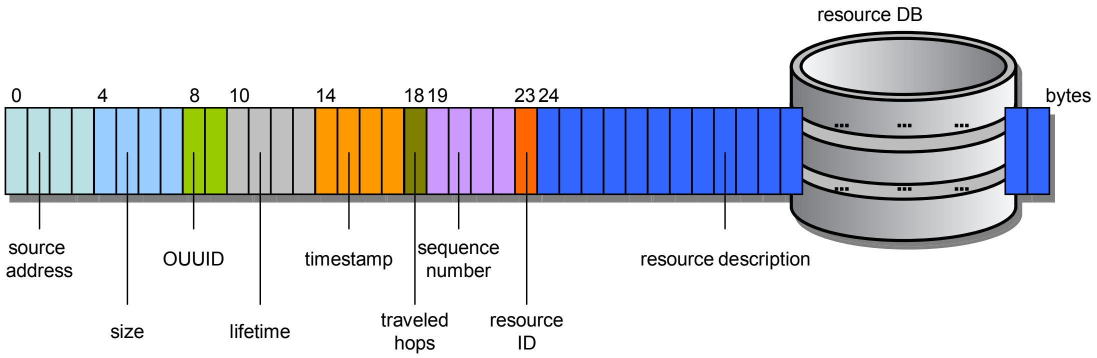


Fig. 4. The structure of a record in the resource database.

- `N`: the number of resources handles (and then cache tuples) the packet transports.
- `REQUEST ID`: identifier of the original request.
- `OUUID`: identifier of the reference ontology.
- `LAST HOP SEQUENCE NUMBER`: the sequence number of the node sending the packet.
- `DESTINATION ADDRESS`: requester IP address.

Last fields are the resource records.

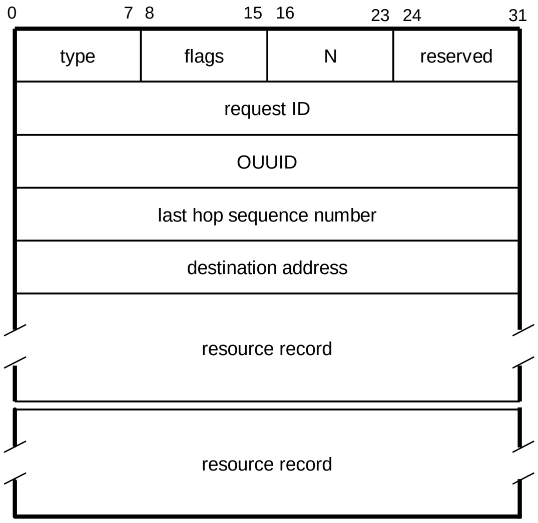


Fig. 5. The structure of a cache content PDU.

4 Interactions between Service Discovery and Routing protocols

In mobile ad-hoc contexts, the routing algorithm monitors the network topology and keeps track of the best paths toward any known destination by means of a set of metrics. Each node maintains a routing table containing an entry for each active route. This record hosts the destination address, the next hop address, the route length, the last known sequence number produced by the destination, etc. [19].

AODV [31] is a widespread Distance Vector routing protocol, implemented and used on many platforms [7] and architectures [8]. It is a reactive protocol (routing information is not sent at regular intervals, but only when a node needs a new route or an updated one), which guarantees a good scalability on small to medium sized networks, with reasonably low latencies, as a route is built only when required. AODV stores just one active route toward a destination, and for each destination endpoint a node knows the total number of hops to traverse and the address of the next hop.

The way AODV stores its information led us to choose it as the routing protocol in our approach, as our framework exchanges information about the server or the client node between adjacent nodes during the advertisement and solicit phase. Thus, we have the opportunity to regularly update AODV entries at no extra cost in terms of number of frames sent, while improving the overall reliability of the routing protocol itself.

When a route toward a destination is needed and an active one does not exist yet (or it has expired), the AODV routing protocol starts a path discovery procedure broadcasting *Route Request* packets [31]; the destination node replies to the request with unicast frames processed hop by hop in order to create an active bidirectional route. Moreover, as nodes of a MANET may move, links between them can break. Whenever a node detects a broken link, it decides to initiate either a *Route Repair* process or a new *Route Discovery*, depending on the routing protocol. Nevertheless, MAC protocols for MANETs do not provide any handshake mechanism for broadcasting (such as the four-way handshake sequence used in IEEE 802.11), then the transmitting node ignores if *Route Request* messages have been actually delivered during the path discovery phase. Hence *Route Request* packets are periodically sent until a timer expires or a *Route Reply* packet is received. Note that reactive routing protocols like AODV [31] build a route toward a destination only in case of an explicit request, whereas proactive ones (as DSDV [32]) keep track of at least one route toward each node in the network.

Also resource discovery protocols can use multicast/broadcast messages during the advertisement/discovery phase [24,29,40] in order to propagate information about all the available resources. In the proposed approach, routing information are piggybacked in the advertisement, solicit, demand, and cache content PDUs. In this way, paths toward server nodes are pro-actively set, thus minimizing latencies due to the route creation phase.

---

[7] AODV-UU module for Linux and Embedded Linux, available at http://core.it.uu.se/core/index.php/AODV-UUm; UoB-WinAODV, AODV implementation for Microsoft Windows, available at http://www.comnets.uni-bremen.de/ adu/

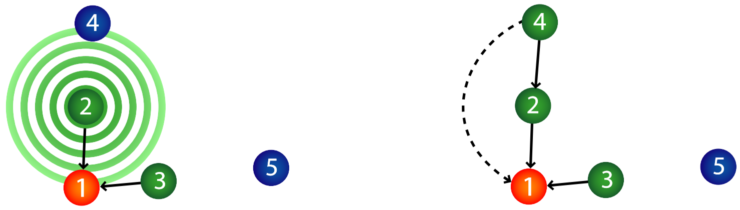
[8] The ZigBee Alliance, http://www.zigbee.com

Each node receiving an advertisement is primarily able to build an active route toward the provider at one hop distance. Furthermore, also a node receiving a forwarded advertisement can build a route toward a provider (by exploiting the provider sequence number and the `traveled hops` value). When the advertisement phase has been accomplished, each node which has received an advertisement maintains an active route toward every resource provider within a `max_adv_hops` range.

With reference to Fig. 6, node 1 is a provider advertising managed resources to its neighbors. When nodes 2 and 3 receive the advertisement, they add (or update) an entry for node 1 within their routing table (setting the distance value to one hop). Next, node 2 forwards the received advertisement to node 4, which updates its routing table adding (or updating) a record for node 3 and another one for node 1; the latter sets a distance of two hops and exploits node 3 as next hop. Notice that all nodes have to update routing information before deciding if the packet has to be processed by the upper Service Discovery layer. A route toward the requester is built in a similar way when it sends a solicit frame.

1. Node 1 sends an advertisement packet, received by nodes 2 and 3

2. Node 2 and node 3 build reverse routes to node 1



3. Node 2 forwards the advertisement packet, received by node 4

4. Node 4 builds reverse routes to node 1

⟵ Active one-hop route
⇠ Active two-hops route
🟥 Server node
🟩 Forwarding node

Fig. 6. Building of reverse route during the advertisement phase.

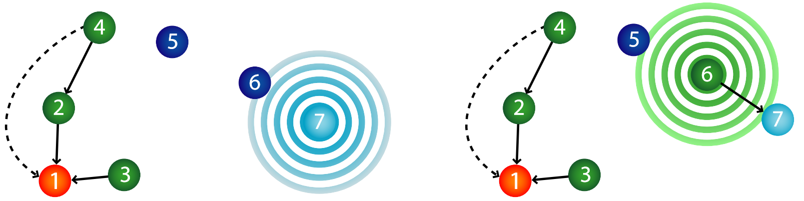
If a node receiving a solicit manages one or more cache entries matching contained parameters, it replies with a cache content PDU to the node the solicit comes from. When a node receives a cache content PDU it first updates its routing table building a route toward the node transmitting that PDU, and the resource record database is updated. Moreover, since cache content PDUs are not simply forwarded but processed at every hop by both Routing and Service Discovery protocols, every node receiving a cache content will build a route toward each known provider.

Routes are built using solicits as shown in Fig. 7, where route creation and packet transmission are jointly shown for the sake of brevity even if they do not occur simultaneously. Node 4 receives an advertisement broadcast by node 1, so it has an active route toward the node. Node 7 broadcasts a solicit to require specified services/resources (suppose resource hosted by node 1 matches the requested parameters); the solicit packet is received by node 6 and then forwarded to node 5, so that both nodes will build routes toward 7. When the solicit reaches node 4, it will use the available route in order to send a cache content packet –containing resource descriptions whose OUUID matches the query– toward node 5. Node 5 will update its routing table as well as the resource record database, furthermore it will send a cache content frame to node 6 and so on. Finally, node 7 will have an active route toward node 1. The route from node 1 to node 7 is built when node 7 sends a demand PDU. Hence, both cache content and demand PDUs complete the route discovery process ideally started with advertisement and solicit PDU broadcast propagation.

5 Semantic Based Resource Composition

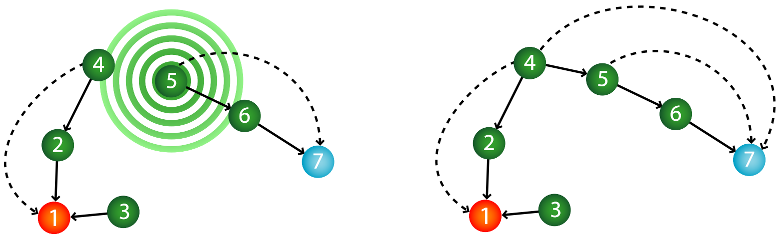
Here we show how to compute a semantic based automated composition of mobile resources and how to orchestrate them for building personalized services. Let us suppose a generic client has to search for a resource in an ad-hoc context also setting both a minimum covering threshold (in percentage) and a maximum discovery distance (in terms of number of hops).

We basically hypothesize a requester-centric composition system. That is, in the proposed approach, the requester is exactly the resource orchestrator and other nodes in the ad-hoc environment assume only a passive role. The requester is searching for a complex service and collects various component resources from nodes at one hop. It attempts to cover the request starting the composition algorithm described later on and, if it fails to overcome the minimum covering threshold, it will request to nearby nodes new service descriptions in their respective cache. This step has to be repeated until the threshold is surpassed or the maximum hops number is reached. We are not necessarily interested in a full satisfaction of the request which, in our mobile environment, may take a prohibitive time; on the other hand, we want to satisfy it as much as possible. If retrieved resources do not allow to completely fill the request, an approximate solution has to be taken into account, possibly providing an explanation of the approximation. Observe that the composition process is totally decentralized. Indeed, various resource providers take



1. Node 7 sends a solicit packet with maximum search diameter 3, received by node 6

2. Node 6 builds reverse route to node 7 and forwards the solicit packet to node 5



3. Node 5 builds reverse routes to node 6 and node 7 and forwards the solicit packet to node 4

4. Node 4 builds reverse routes to node 5 and node 7. Now forward route from 7 to 1 can be built.

⟵	Active one-hop route
⇠	Active multi-hop route
🟥	Server node
🟩	Forwarding node
🟦	Node requesting services

Fig. 7. Building of forward route during the solicit phase.

part to it. Each host contributes to cover the whole request or part of it by means of resource descriptions in its cache. We do not have single resource registries or repositories, but resources are disseminated within the network. Each node hosting one or more resources can be considered as a registry for its neighbors.

To explain and motivate the approach and the rationale behind it, we present a semantic based service discovery model enhanced with resource composition features. In the initial architecture we define both the request D and the description of each available resource within the network as DL concepts both referred to an ontology $\mathcal{T}$ shared among some network users. We perform a preliminary selection procedure based on OUUIDs in order to select devices managing the same "resource class". Hence, given a request D modeled w.r.t. $\mathcal{T}$, the second step discovery is performed following semantic based criteria. Main steps of the composition process are reported hereafter:

(i) Put all the retrieved descriptions in a set $\mathcal{R}$.

(ii) Call $resourceComposer$ algorithm with input $\mathcal{R}$, D and $\mathcal{T}$.

(iii) Compute the $Covering\_Level$.

(iv) Is there an exact solution?
    (a) If yes, the algorithm outputs the exact solution. Exit.

(b) If not, continue.

(v) $Covering\_Level < Minimum\_Threshold_{Covering\_Level}$?
   (a) If yes, continue.
   (b) If not, go to step (viii).

(vi) Has maximum discovery range been reached?
   (a) If yes, go to step (viii).
   (b) If not, continue.

(vii) Collect new resource descriptions and repeat steps from (i) to (v).

(viii) The algorithm outputs an approximate solution to the retrieval problem and, in case, an explanation on why the solution is not an exact one. Exit.

Note that the orchestration process can be halted at each moment (both the *covering level* and the *discovery range* will be used as triggers to stop the algorithm). In this case, the composite mobile resource partially covering the request will be considered as approximate solution and the uncovered part of the request as explanation of what is still missing to fully cover initial needs (see Section 7.1 for further details). In particular, the covering level is a value measuring the similarity degree between the composite service/resource and the request indicating that it might potentially satisfy the request itself in an approximate fashion.

Here we extend the service composition model in [34] to deal with a pervasive scenario. For the sake of clarity, we only recall main terms and definitions. In particular we define:

Mobile Resource: a triple $\langle MR_D, \boldsymbol{P}, \boldsymbol{E}\rangle$ where MR_D is the description of provided resource, P its preconditions and E the effects, all expressed with respect to the ontology $\mathcal{T}$.

Furthermore, indicating with AI_i the available information for the i-th mobile resource mr_i and with E_j the effects produced by mr_j, with $j < i$, the following relation ensues: $AI_i = P_0 \sqcap E_1 \sqcap E_2 \sqcap ... \sqcap E_{i-1}$.

Mobile Resource Flow (with respect to some initial preconditions P_0): is a finite sequence of mobile resources $\mathcal{MRF}(P_0) = (mr_1, mr_2, ..., mr_i, ..., mr_n)$, where for each $mr_i \in \mathcal{MRF}(P_0)$ the following conditions ensue:

(i) for mr_1, $P_0 \sqsubseteq P_1$;

(ii) for mr_i, $i > 1$, $AI_i \sqsubseteq P_i$;

(iii) for mr_i, $i > 1$, for each concept name CN occurring in E_i, $AI_i \not\sqsubseteq CN$.

Based on the definition of $\mathcal{MRF}(P_0)$, here we define a **composite mobile resource** with respect to a request D. In particular, a composite mobile resource for $\langle D, P_0\rangle$ w.r.t. the set of discovered resources $\mathcal{R}$, from now on $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)$, is a mobile resource flow such that for each mr_j in the execution flow: $D_{\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)} = \{MR_D(j) | mr_j \in \mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)\}$ covers D.

An **executable mobile resource** mr^{ex} for $\mathcal{MRF}(P_0)$ is a mobile resource which can be invoked after the execution of $\mathcal{MRF}(P_0)$, *i.e.*, its preconditions are satisfied after the execution of $\mathcal{MRF}(P_0)$, and such that its effects are not already provided by $\mathcal{MRF}(P_0)$.

Given a mobile resource flow $\mathcal{MRF}(P_0)$ and a set of mobile resources $\mathcal{R} = mr_i$, an **executable set** of $\mathcal{MRF}(P_0)$ is the set of all the $mr_i \in \mathcal{R}$ such that mr_i is an executable mobile resource for $\mathcal{MRF}(P_0)$. $\mathcal{EX}_{\mathcal{MRF}(P_0)} = \{mr_i^{ex} | mr_i^{ex}$ is an executable resource for $\mathcal{MRF}(P_0)\}$.

The *resourceComposer* algorithm is at the core of the system. It allows the composition of mobile resources, and it returns a complex service taking into account preconditions that must be satisfied.

The maximum discovery distance (search diameter) and the minimum threshold covering level are exploited as external parameters regulating the way the whole discovery process happens. They are exploited to determine the spatial involvement of nodes in the discovery, and the approximation level in case of non exact matches, respectively [38].

In order to allow a primary composition with resources in the cache of the requester device, the algorithm will be run for the first time on the requester device itself. It outputs the $\mathcal{CMR}$ as well as the uncovered part of the request $D_{uncovered}$. If the covering level is under a specified threshold, the uncovered part of the request as well as the temporary resulting $\mathcal{CMR}$ are stored and the requester broadcasts a solicit packet in an expanding ring fashion to require other resource descriptions.

In the proposed composition framework, we take into account the influence of distance between offered and demanded service as well as the dynamic structure of the context. Note that the set of components resources is not assigned a priori, but it changes according to network evolution. Hence, we adapt a Semantic Web services composition approach [17] to our “unstable scenario”. In this case we have subsequent composition processes. The orchestration goes through several steps producing a progressive refinement of results. For each step, after the $\langle \mathcal{CMR}, H \rangle$ calculation (by means of *resourceComposer* algorithm), the $Covering\_Level$ is computed as:

$$Covering\_Level(\mathcal{CMR}_i) = 100 \cdot [1 - \frac{s\_match(D, \mathcal{CMR}_i)}{max(s\_match)}]$$

where $s\_match(D, \mathcal{CMR}_i)$ is the semantic distance from request to computed Composite Mobile Resource; $max(s\_match) \doteq s\_match(D, \top)$ is the maximum semantic distance from the request, which depends on axioms in the domain ontology [21].

```
Algorithm $resourceComposer(\mathcal{R}, \langle D, P_0 \rangle, \mathcal{T})$

input a set of resources $\mathcal{R} = \{mr_i = \langle MR_D(i), P_i, E_i \rangle\}$, a
request $\langle D, P_0 \rangle$ - where $D$ and $MR_D(i)$ are satisfiable in $\mathcal{T}$-
output $\langle \mathcal{CMR}, H \rangle$

begin algorithm
```

```
2     $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle) = \emptyset$;
3     $D_{uncovered} = D$;
4     $H_{min} = D$;
5     do
6       compute $\mathcal{EX}_{\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)}$;
7       $MR_{Dmin} = \top$;
8       for each $mr_i \in \mathcal{EX}_{\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)}$
9         if $\mathcal{D}_{\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)} \cup \{MR_D(i)\}$ covers $D_{uncovered}$ then
10          $H = solveCAP(\langle \mathcal{L}, MR_D(i), D_{uncovered}, \mathcal{T}\rangle)$;
11          if $H \prec_{\mathcal{T}} H_{min}$ then
12            $MR_{Dmin} = MR_D(i)$;
13            $H_{min} = H$;
14          end if
15        end if
16        if $MR_{Dmin} \not\equiv \top$ then
17          $\mathcal{R} = \mathcal{R} \backslash mr_i$;
18          $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle) = (\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle), mr_i)$;
19          $D_{uncovered} = H_{min}$;
20        end if
21      end for each
22    while($MR_{Dmin} \not\equiv \top$);
23    return $\langle \mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle), D_{uncovered}\rangle$;
24  end algorithm
```

The resource composition algorithm.

The algorithm starts with the initialization of $\mathcal{CMR}$, $D_{uncovered}$ and H_{min} (lines 2-4). *resourceComposer* returns the composite mobile resource $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)$ and the part of the request D remained -in case- uncovered, *i.e.*, $D_{uncovered}$. This latter is computed solving a CAP (line 10), hence it is a possible explanation for the uncovered part of the request. In particular it depends on the minimality criterion adopted to solve the CAPs during the algorithm execution. In line 4 $H_{min} = D$ means that the initial *minimal* hypothesis about what is needed to fully cover the request must be obviously fixed to D. The system implements $rankPotential$ algorithm (see [22] for further details) in line 11 to rank concepts deriving from the Concept Abduction Problem solution (line 10). That is the comparison in line 11 is made possible thanks to the numerical evaluation of $|H|$ via *rankPotential* :

$$|H| = rankPotential(MR_D(i), D_{uncovered})$$

In spite of increasing covering possibilities, the involvement of nodes farther and farther in the MANET implies a greater risk in terms of persistence of links between requester and providers. Hence, it is useful to define a metric which takes into account distance (in hops number) from requester to providers for correcting the semantic matching result. Resources which are “located” on mobile devices in proximity of requester should be better ranked than the far off ones, given the same semantic similarity degree. In other words *rankPotential* outcome, allowing the comparison in line 11, could be corrected by exploiting a logarithmic function. In fact, such type of function presents a growth almost proportional with the distance for a short number of hops, but has values almost constant over a specified limit [38]. A $(1 + log_{10}n)$ factor –where n is the number of hops from requester to provider– could be introduced:

$$|H| = \textit{rankPotential}\ (MR_D(i), D_{uncovered}) \cdot (1 + log_{10}n).$$

This functionality is a possible further improvement of the proposed approach, but it is not yet implemented in the prototype we refer here.

6 Service Substitutability

In a pervasive environment, it is hard to hypothesize that all discovered resources are simultaneously available. In fact, throughout execution, either a resource could fail or it could become unreachable because of the host mobility. Furthermore, while the composition is in progress, a better resource could be detected, as well as newer releases of already discovered ones could become available [20]. Hence, in these cases it could be necessary to obtain the automatic substitution –in a dynamic fashion– of resource components no more available, or suitable, with new ones.

In order to implement this feature, we define a *Similarity Group* as a collection of components which can be substituted with each other [20]. Obviously the classification of an object to determine whether it belongs to the group is a fundamental issue. A set of rules for substitution of a mobile resource with another one has to be defined [6]. Notice that the *Similarity Group* ($\mathcal{SG}$) is a simple set of resources without any order relation (a MANET has a too instable structure). Also notice that an $\mathcal{SG}$ is created with respect to each resource in the $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)$, so that each constituent resource can have a set of substitutes.

Information about resource interfaces (*i.e.*, required preconditions and provided effects) are needed prior to admit a resource in a substitutability class [28]. This is mandatory to evaluate its correct insertion in the $\mathcal{MRF}(P_0)$. In order to decide if a generic resource can belong to an $\mathcal{SG}$, two conditions about preconditions and effects have to be verified. Let us suppose $(mr_1, \ldots, mr_N)$ are services in a $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)$ and we have to create the mr_i similarity group $\mathcal{SG}(i)$. We say $mr_j^{sub}(j = 1...L) \in \mathcal{SG}(i)$ iff the following conditions hold:

(i) mr_j^{sub} is an *executable mobile resource* for $(mr_1, mr_2, ..., mr_{i-1})$

(ii) for $h = i+1, \ldots, N$, mr_h is an *executable mobile service* for $(mr_1, mr_2, ..., mr_{h-1})$

This is the theoretical framework but, in practice, the substitutability is implemented in a more compact way. To verify if a substitute resource mr_j^{sub} is suitable, we take into account the already available AI_i (if mr_i is the resource to substitute) and we recalculate next AI_k (with $k = i + 1...N$) checking the satisfiability of the respective mr_k. In other words, if AI_i is the available information for mr_i and if we want to substitute the same mr_i, the new $AI'_{i+1} = AI_i \sqcap E_j^{sub}$ has to be determined. Hence, it will be used for checking the satisfiability of the next mr_{i+1} and so on. If one of these checks fails, we can conclude that the mr_j^{sub} is unsuitable. In the progressive substitution of services in a $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)$, it is useful to start with first services in the flow to reuse the already determined AI in the next substitution steps, and consequently reduce the overhead deriving from this processing. Hence, if the generic mr_i belonging to $\mathcal{CMR}(\langle D, P_0, \mathcal{R}\rangle)$ suddenly turns unavailable, we can take a generic mr_j^{sub} from the $\mathcal{SG}(i)$ with $j = 1, \ldots, L$ and substitute it.

Notice that the similarity group of the resource mr_i is created at discovery phase, but it is more and more enriched while the discovery progresses, that is new resources are discovered extending to the next hop the search. Hence, if $\mathcal{SG}^k(i)$ is the similarity group of the service m_i at hop k, we can finally say $\mathcal{SG}(i) = \mathcal{SG}^1(i) \cup \mathcal{SG}^2(i) \cup \ldots \cup \mathcal{SG}^{MAX_{hop}}(i)$.

7 Case study

During calamities or disasters involving urban environments, rescue and recovery operations can be hampered by logistic flaws and communications failures. The damage or the destruction of main infrastructures during a disaster makes time critical for missions aiming at reestablishing basic communications for first aid. In those cases, an ad-hoc network, with support for service/resource discovery, could be quickly set up to support the command and control needs of the rescue and recovery teams [41]. Mission-critical data exchanged among rescuers and toward the headquarter should grant an efficient and quick coordination using a combination of wireless network technologies and discovery applications to meet typical constraints of unpredictable and unreliable scenarios [27]. In the last few years new approaches to disaster/recovery (d/r) are addressing the possibility to provide assistance in areas damaged by calamities thanks to autonomous robots opportunely organized in teams [42]. Robots are more suitable with respect to military forces to face hazardous operations where their contribution is fundamental in supporting life-threatening human tasks. Hence robotics has a promising role in the so called *Urban Search And Rescue* (USAR) field. For instance, robots can provide a significant help to search and rescue survivors trapped under collapsed buildings after earthquakes.

Rescue robots are usually operated remotely from outside the calamity perimeter. Nevertheless in those cases, the teleoperation requires a noteworthy human intervention with a large number of trained controllers. The coordination among robots and human experts is difficult, and it may result in a limitation of the operations speed, due to the hardness in taking shared decisions. Furthermore complex environmental conditions, as for instance low visibility, make difficult the human manoeuvering of robotic devices, and teleoperation relies on continuous availability of robust connectivity and power supplies which are often unavailable in disaster scenarios. Hence, in order to get closer to survivors, recently the exploitation of auto-piloted mobile robots with various shapes, sizes and capabilities [42] has been tested. One unavoidable challenge is that search and rescue teams must be self-guided and self-coordinated. Decisions must be made on the field by a coordinator robot which has to interact with human controllers only for higher-level commands. The features and capabilities of the components of the rescue team have to be orchestrated by a composer robot, which acts as team leader.

The proposed approach and framework have been evaluated in a case study for a simulated d/r planning. The scenario encompasses an environment where a search and rescue robot team has to be coordinated by a robot supervisor. In what follows a complete illustrative example is presented to better explain the rationale behind our approach and to let emerge its added value with respect to classical resource discovery paradigms. A performance evaluation referred to the case study is also presented allowing to give a closer look to the network level performances provided by the proposed approach.

7.1 Illustrative Example

A s/r mission is hypothesized where a robot crew has to be coordinated to rescue survivors trapped under buildings collapsed after a major disaster. Different robots are equipped with various facilities and devices. The orchestrator, according to the final goal of the mission and to preconditions deriving from an examination of the disaster scenario has to perform the composition of the single useful features provided by each component of the team in order to reach the planned objective. The general goal of the robot team is to venture into a hostile area, collect and report relevant information and then bring the area back to safety. Each team is composed by robots with different attitudes and capabilities, equipped with tools and facilities to cope with specified tasks. The controller robot has to properly orchestrate individual appliances and devices managed by each robot. The wireless 802.11 connectivity allows an easier cooperation and control among robots of the team with the aim of reaching a specified goal. Hence, among a set of robotic units, each endowed with its capabilities and limitations, the best team has to be selected in order to perform the s/r operation.

A knowledge base was developed to express relevant concepts and properties for this application family. Since we pursue a service-oriented and mission-driven approach, robot units are described focusing on provided and required capabilities rather than a technically-oriented description of their components derived from datasheets. Modeled capabilities include on-board sensing and acting devices, supported wireless communication protocols and the kinds of terrain and obstacles that can be passed. For the sake of conciseness, Figs. 8 and 9 report only a subset of the ontology axioms particularly useful to understand the following example (classical DL notation is adopted hereafter for better readability).

Let us formalize the above reference scenario through a small example. “*An accident has occurred in a chemical plant during the night. An explosion has set the building on fire, and workers are trapped in. A robot team has been sent to search and rescue survivors. Besides fire, environmental hazards are not precisely known a priori but may include toxic gases and the presence of debris, liquid pools*

Physical_quantity ⊑ ⊤	Mass ⊑ Physical_quantity
Temperature ⊑ Physical_quantity	Vibration ⊑ Physical_quantity
Concentration ⊑ Physical_quantity	Relative_humidity ⊑ Concentration
Angle ⊑ Physical_quantity	Longitude ⊑ Angle
Latitude ⊑ Angle	Length ⊑ Physical_quantity
Altitude ⊑ Physical_quantity	Distance ⊑ Length
Speed ⊑ Physical_quantity	Pressure ⊑ Physical_quantity
Connectivity ⊑ ⊤	WLAN ⊑ Connectivity
WPAN ⊑ Connectivity	WWAN ⊑ Connectivity
WiFi ⊑ WLAN	Bluetooth ⊑ WPAN
ZigBee ⊑ WPAN	Telephony ⊑ WWAN
Cellular_telephony ⊑ Telephony	Satellite_telephony ⊑ Telephony
Sensor ⊑ ∃measures ⊓ ∀measures.Physical_quantity	Geographical ⊑ Sensor
Weather ⊑ Sensor	Scales ⊑ Sensor ⊓ ∀measures.Mass
Acoustic ⊑ Sensor ⊓ ∀measures.Vibration	Imaging ⊑ Sensor
Chemical ⊑ Sensor ⊓ ∀measures.Concentration	Rangefinder ⊑ Sensor ⊓ ∀measures.Distance
Air_analysis ⊑ Chemical	NH3_air_analysis ⊑ Toxic_gas_analysis
CO2_air_analysis ⊑ Toxic_gas_analysis	SO2_air_analysis ⊑ Toxic_gas_analysis
Cl2_gas_analysis ⊑ Toxic_gas_analysis	CO_gas_analysis ⊑ Toxic_gas_analysis
Water_analysis ⊑ Chemical	Bacteriological_water_analysis ⊑ Water_analysis
Toxic_metal_water_analysis ⊑ Water_analysis	Mercury_water_analysis ⊑ Toxic_metal_water_analysis
Lead_water_analysis ⊑ Toxic_metal_water_analysis	Radioactive_metal_water_analysis ⊑ Toxic_metal_water_analysis
Radar ⊑ Rangefinder	Sonar ⊑ Rangefinder
Laser_rangefinder ⊑ Rangefinder	Seismometer ⊑ Acoustic
Microphone ⊑ Acoustic	Videocamera ⊑ Imaging
Nocturnal_videocamera ≡ Videocamera ⊓ Nigth_vision	Telephoto_lens ⊑ Imaging
Camera ⊑ Imaging	Color_camera ⊑ Camera
Black_white_camera ⊑ Camera	Anemometer ⊑ Weather ⊓ ∀measures.Speed
Barometer ⊑ Weather ⊓ ∀measures.Pressure	Hygrometer ⊑ Weather ⊓ ∀measures.Relative_humidity
Thermometer ⊑ Weather ⊓ ∀measures.Temperature	
Computational_resource ⊑ ⊤	Processor ⊑ Computational_resource
DSP ⊑ Processor	CPU ⊑ Processor
Realtime_guarantee ⊑ Computational_resource	Soft_realtime ⊑ Realtime_guarantee
Hard_realtime ⊑ Realtime_guarantee	
Obstacle ⊑ ⊤	Step ⊑ Obstacle
Slope ⊑ Obstacle	Debris ⊑ Obstacle
Pool ⊑ Obstacle	Hole ⊑ Obstacle
Tunnel ⊑ Obstacle	

Fig. 8. Axioms in the disaster recovery ontology used in the case study.

Movement $\sqsubseteq$ $\exists$overcomes_obstacle $\sqcap$ $\forall$overcomes_obstacle.Obstacle	Air_movement $\sqsubseteq$ Movement $\sqcap$ $\forall$overcomes_obstacle.(Debris $\sqcap$ Hole $\sqcap$ Step $\sqcap$ Pool $\sqcap$ Slope)
Water_movement $\sqsubseteq$ Movement $\sqcap$ $\forall$overcomes_obstacle.Pool	Ground_movement $\sqsubseteq$ Movement $\sqcap$ $\forall$overcomes_obstacle.(Tunnel $\sqcap$ Slope)
Helicopter $\sqsubseteq$ Air_movement	Aerostat $\sqsubseteq$ Air_movement
Wheels $\sqsubseteq$ Ground_movement	Caterpillar_track $\sqsubseteq$ Ground_movement
Amphibious_movement $\equiv$ Ground_movement $\sqcap$ Water_movement	
Action $\sqsubseteq$ $\top$	Push $\sqsubseteq$ Action
Light $\sqsubseteq$ Action	Pierce $\sqsubseteq$ Action
Pull $\sqsubseteq$ Action	Estinguish_fire $\sqsubseteq$ Action
Process $\sqsubseteq$ Action	Lift $\sqsubseteq$ Action
Cut $\sqsubseteq$ Action	Plough_through_debris $\sqsubseteq$ Action
Detection $\sqsubseteq$ Action	Fire_detection $\sqsubseteq$ Detection
Presence_detection $\sqsubseteq$ Detection	Mine_detection $\sqsubseteq$ Detection
Actuator $\sqsubseteq$ $\exists$performs_action $\sqcap$ $\forall$performs_action.Action	Torch $\sqsubseteq$ Actuator
Signalling $\sqsubseteq$ Actuator	Dozer_blade $\sqsubseteq$ Actuator $\sqcap$ $\forall$performs_action.Push
Drill $\sqsubseteq$ Actuator $\sqcap$ $\forall$performs_action.Pierce	Hammer_drill $\sqsubseteq$ Drill
Forklift $\sqsubseteq$ Actuator$\sqcap$$\forall$performs_action.Lift$\sqcap$ $\exists$has_capacity	Forklift_200_kg $\sqsubseteq$ Forklift $\sqcap$ $\forall$has_capacity.($\leq$ $200\ kg$)
Pincers $\sqsubseteq$ Actuator	Fire_extinguisher $\sqsubseteq$ Actuator $\sqcap$ $\forall$performs_action.Extinguish_fire
Powder_fire_extinguisher $\sqsubseteq$ Fire_extinguisher	
Power_source $\sqsubseteq$ $\top$	Internal_power_source $\sqsubseteq$ Power_source
Solar_panel $\sqsubseteq$ Internal_power_source	Battery $\sqsubseteq$ Internal_power_source
Fuel $\sqsubseteq$ Internal_power_source	Outlet $\sqsubseteq$ Power_source
Outlet $\sqsubseteq$ $\neg$Internal_power_source	
Supply $\sqsubseteq$ $\top$	Food_supply $\sqsubseteq$ Supply
Water_supply $\sqsubseteq$ Supply	Water_purifier $\sqsubseteq$ Supply
First_aid_kit $\sqsubseteq$ Supply	
Defense $\sqsubseteq$ $\top$	ECM $\sqsubseteq$ Defense
Fireproof_plate $\sqsubseteq$ Defense	Armor_plate $\sqsubseteq$ Defense
Camouflage $\sqsubseteq$ Defense	

Fig. 9. Axioms in the disaster recovery ontology used in the case study (continued).

and narrow tunnels. The robot team is coordinated by a mobile headquarter deployed at short distance from the disaster area. Communication is provided by an ad-hoc IEEE 802.11 network. Before deployment into the field, energy has been properly supplied to robot units by means of batteries and/or fuel. The headquarter is the sink of collected data by the sensor array of robot units deployed in the field, providing information processing through dedicated computational resources such as DSPs."

Each mobile unit has different capabilities and operational requirements, hence the overall search and rescue mission has to be divided into tasks which must be properly orchestrated. "*Disaster management rules for the rescue mission include* *an air analysis, then exploration of the area (nocturnal visibility is required) by moving through debris. Fire sources should be detected and extinguished in order to proceed with exploration. Human presence must be detected and people have to be aided and extracted from the disaster area.*"

The above mission goal is the request for the service orchestration algorithm that has to find the most suitable composition of operational units. Similarly, facilities provided by the headquarter are modeled as the initial supplied preconditions. With respect to the ontology, they can be expressed as follows:

Request: $D = Air\_analysis \sqcap Night\_vision \sqcap \exists performs\_action \sqcap \forall performs\_action.(Plough\_through\_debris \sqcap Detect\_fire \sqcap Extinguish_{f}ire \sqcap Detect\_presence) \sqcap \exists overcomes\_obstacle \sqcap \forall overcomes\_obstacle.(Debris \sqcap Pool \sqcap Slope \sqcap Tunnel) \sqcap First\_aid\_kit$

Initial_preconditions: $P_0 = WiFi \sqcap Fuel \sqcap Battery \sqcap DSP$

The *resourceComposer* algorithm can be applied. Preconditions and effects are described hereafter for each available robot unit in the mission area:

$ms_1 : Environmental\_unit = \langle P_1, E_1\rangle = \langle WiFi \sqcap Internal\_power\_source, GPS \sqcap Altimeter \sqcap Anemometer \sqcap Barometer \sqcap Hygrometer \sqcap Thermometer \sqcap CO2\_air\_analysis \sqcap SO2\_air\_analysis \sqcap CO\_air\_analysis\rangle$

$ms_2 : Meteorological\_unit = \langle P_2, E_2\rangle = \langle WiFi \sqcap Internal\_power\_source, GPS \sqcap Altimeter \sqcap Anemometer \sqcap Barometer \sqcap Hygrometer \sqcap Thermometer\rangle$

$ms_3 : Mine\_unit = \langle P_3, E_3\rangle = \langle Fuel, Wheels \sqcap Beacon \sqcap Loudspeaker \sqcap Metal\_detector \sqcap \exists performs\_action \sqcap \forall performs\_action.Mine\_detection\rangle$

$ms_4 : Fire\_victim\_detection\_unit = \langle P_4, E_4\rangle = \langle Battery \sqcap GPS \sqcap Videocamera \sqcap Microphone \sqcap Thermometer \sqcap CO2\_Air\_Analysis \sqcap DSP, \sqcap \exists performs\_action \sqcap \forall performs\_action.(Detect\_presence \sqcap Detect\_fire)\rangle$

$ms_5 : Rescue\_unit = \langle P_5, E_5\rangle = \langle \exists performs\_action \sqcap \forall performs\_action.(Detect\_presence \sqcap Detect\_fire), Pincers \sqcap Powder\_fire\_extinguisher \sqcap First\_aid\_kit\rangle$

$ms_6 : Scout\_unit = \langle P_6, E_6\rangle = \langle GPS \sqcap Altimeter \sqcap Anemometer \sqcap WiFi \sqcap Fuel, Amphibious\_movement \sqcap Caterpillar\_track \sqcap Hammer\_drill \sqcap Forklift\_200\_kg \sqcap Nocturnal\_videocamera \sqcap Laser\_rangefinder \sqcap Microphone\rangle$

Let us suppose to assign a covering threshold of 90%. The composition process will be stopped only when the $Covering\_Level$ will overcome this value.

a. The composition process starts with:

$\mathcal{CMR} = \emptyset$

$\mathcal{D}_{uncovered} = D$

$Covering\_Level = 0\%$

b. After the first step, the following results are obtained:

$\mathcal{EX}(\mathcal{CMR}) = \{\mathtt{ms}_1, \mathtt{ms}_2, \mathtt{ms}_3\}$

$\mathcal{CMR} = (\mathtt{ms}_1)$

$\mathcal{D}_{uncovered} = Night\_vision \sqcap \exists performs\_action \sqcap \forall performs\_action.Plough\_through\_debris \sqcap Detect\_fire \sqcap Extinguish\_fire \sqcap Detect\_presence \sqcap \exists overcomes\_obstacle \sqcap \forall overcomes\_obstacle.Debris \sqcap Pool \sqcap Slope \sqcap Tunnel \sqcap First\_aid\_kit$

$Covering\_Level = 9.1\%$

Notice that, among executable mobile resources, only ms_1 contributes to cover the request since it provides resources for air analysis.

c. The second step of the algorithm then produces:

$\mathcal{EX}(\mathcal{CMR}) = \{\mathtt{ms}_2, \mathtt{ms}_3, \mathtt{ms}_6)$

$\mathcal{CMR} = (\mathtt{ms}_1, \mathtt{ms}_6)$

$\mathcal{D}_{uncovered} = \exists performs\_action \sqcap \forall performs\_action.Detect\_fire \sqcap Extinguish\_fire \sqcap Detect\_presence \sqcap First\_aid\_kit$

$Covering\_Level = 63.6\%$

The mobile service unit ms_6 can now be triggered, since its required preconditions P_6 are satisfied (by E_1). It provides sensors and actuators that perform the required mission tasks for exploration of the disaster area. It can also be noticed that ms_2 would cause an unnecessary effect duplication with ms_1.

d. Third step:

$\mathcal{EX}(\mathcal{CMR}) = \{\mathtt{ms}_2, \mathtt{ms}_3, \mathtt{ms}_4)$

$\mathcal{CMR} = (\mathtt{ms}_1, \mathtt{ms}_6, \mathtt{ms}_4)$

$\mathcal{D}_{uncovered} = \exists performs\_action \sqcap \forall performs\_action.Extinguish\_fire \sqcap \sqcap First\_aid\_kit$

$Covering\_Level = 81.8\%$

The mobile service unit ms_4 becomes executable and it is added to the composite flow because it provides resources for detection of fire sources and human presence.

e. Fourth step:

$\mathcal{EX}(\mathcal{CMR}) = \{\mathtt{ms}_2, \mathtt{ms}_3, \mathtt{ms}_5)$

$\mathcal{CMR} = (\mathtt{ms}_1, \mathtt{ms}_6, \mathtt{ms}_4, \mathtt{ms}_5)$

$\mathcal{D}_{uncovered} = \top$

$Covering\_Level = 100\%$

Service unit ms_5 provides further required tools so it is selected. A full covering of the request has been reached and the composition now stops. Each required subtask is covered by one service unit. The output of the algorithm is a service flow, which represents the correct order of intervention of the robots in the team in order to satisfy the mission goal.

7.2 Performance Evaluation

The effectiveness of the proposed framework has been evaluated assuming a rescue team made by 50 robots, each one equipped with a IEEE 802.11 Wireless Network Card Interface, that move in a 1000 square meters area, thus forming a MANET, using ns-2 simulator.

Our attention has been mainly focused on network load, resource discovery effectiveness and responsiveness, overhead due the AODV protocol, and hit ratio, by varying the number of available servers and active clients, for many mobility conditions.

As pointed out in previous sections, at the application level we have a *hit* when the composition process allows surpassing the covering threshold. In order to translate this definition into one that can be easily referred to lower layers of the protocol stack, we have to correlate the covering threshold with the average number of component resources required to overcome the threshold itself.

For that purpose, we have simulated 250 service compositions for three different covering thresholds, *i.e.*, 40%, 70%, and 90%, and exploiting 30 different individuals expressed with reference to the ontology sketched in the previous subsection.

According to the case study and the ontology previously sketched, we have calculated that in order to reach a 90% covering threshold up to 7 component resources may be required. Similarly, up to 6 components are needed to reach a 70% threshold while a 40% covering level needs up to 3 descriptions. Those values have been exploited to consider a hit in the following experiments.

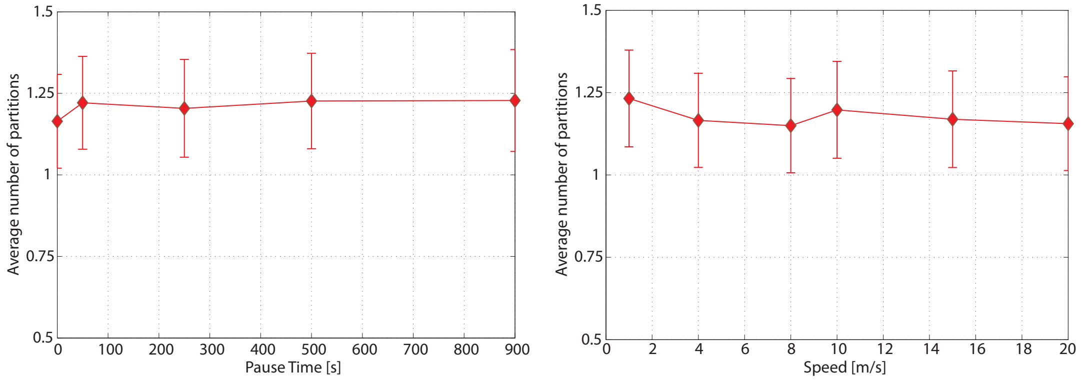
Node mobility is driven by the "random waypoint" model [37] which is characterized by two main parameters, namely `pause time` and `speed`.

The ns-2 simulation starts with nodes remaining stationary for `pause time` seconds, then each node selects a random destination and moves toward it with a fixed speed, randomly chosen in the range [0, `speed`]. After reaching the destination, each node pauses again for `pause time` seconds, then selects another destination and repeats the previous steps till the end of the simulation, which lasts 6000 seconds.

The simulations are arranged in two sets: in the first one the `speed` time has been varied from 1 to 20 m/s while keeping the `pause time` fixed to 0.01 s; in the second set, the `speed` parameter has been set to 1 m/s while the `pause time` has been varied from 0 to 900 s. In both simulation sets, we have considered scenarios with 7, 9, and 11 nodes hosting resources (server nodes) and 15, 30 and 45 client nodes. Resources are activated at the beginning of each simulation, whereas requests are generated at randomly chosen instants, uniformly distributed within the simulation time. Moreover, for each combination of reference parameters (pause time, speed, server and client nodes) we run 8 simulations exploiting different values for the seed of the ns-2 random number generator. Obtained results have been averaged in order to filter out the bias deriving from conditions of single scenarios (*e.g.*, high link breakage ratio or network partitions). In each graph the error bars show the confidence interval at 95% for the considered value.

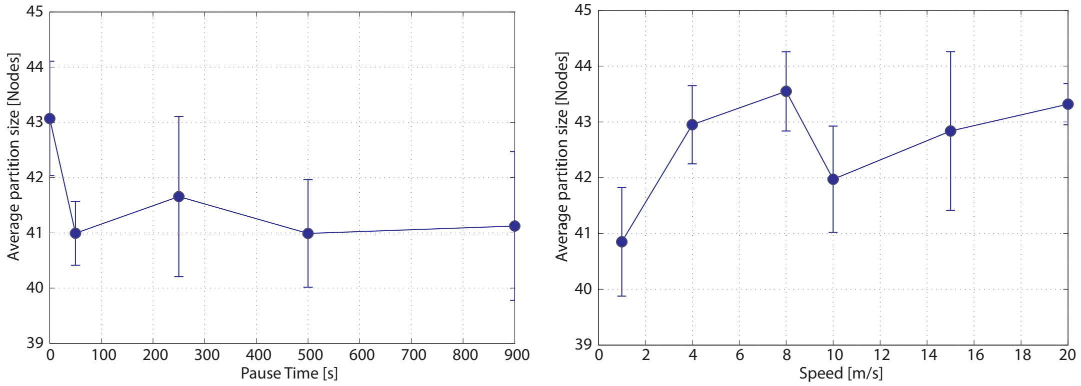
In order to clearly illustrate the impact of node mobility on the networking environment, Figs. 10-12 show the average number of partitions, the average partition size, and the number of joins and splits of partitions, respectively. In particular, Fig. 10 shows that the average number of partitions in each scenario is smaller than two, which means that the considered network has a high degree of connectivity. Obviously, the number of topology splits or joins monotonically decreases (increases) with the pause time (max speed). In fact, increasing the pause time (max speed) leads to a smaller (higher) node mobility with a consequent impact on the number of partition rearrangements.

Observe that a semantic based composition is usually a time and resource con-



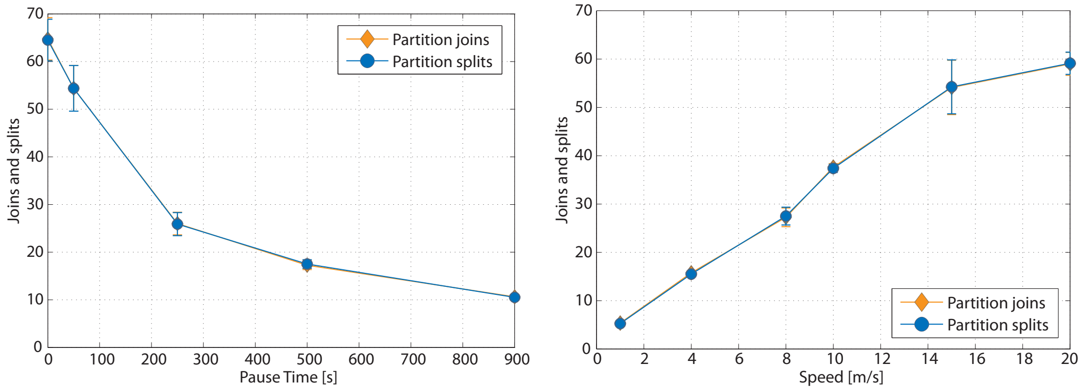
(a) Dependence on the pause time
(b) Dependence on the maximum speed

Fig. 10. Average number of partitions (simulation time = 6000 s).



(a) Dependence on the pause time
(b) Dependence on the maximum speed

Fig. 11. Average partition size (simulation time = 6000 s).



(a) Dependence on the pause time
(b) Dependence on the maximum speed

Fig. 12. Number of partition joins and splits (simulation time = 6000 s).

suming process: simulations should take into account the time employed by each node to run the reasoner which accomplish the orchestration process [18]. We assume all nodes in the network are equipped with CPUs of a widespread architecture found in many PDAs[9]. Produced performances are obviously lower than the ones obtainable by a desktop workstation. In order to deal with a correct estimation of the composition time for a PDA, the processing time measured on the workstation running the simulator has been multiplied by a normalizing constant, which accounts for the performance gap between the simulation PC and a typical PDA architecture. Finally, in order to evaluate the improvement introduced with our cross-layer approach, we run the simulations also disabling the interaction between resource discovery and routing protocols.

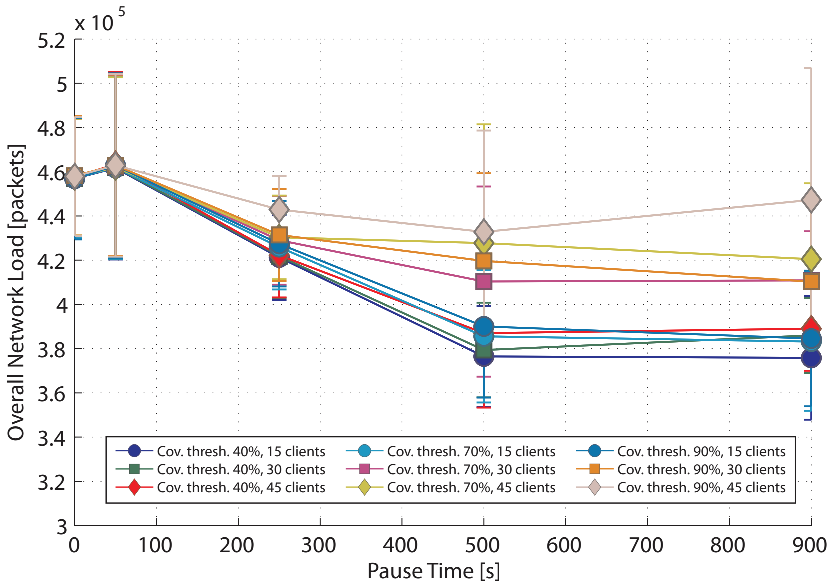
In what follows, results of the overall performance evaluation are reported, starting from those obtained using 7 servers. Then, we will discuss the impact of the number of servers. First of all, Figs. 13 and 14 show the overall packets generated by the resource discovery protocol. We can see that, for a given number of clients, the overall load grows with the covering threshold. This happens because the higher is the covering threshold the more challenging becomes the resource discovery task. In the same way, for a given covering threshold, the load slightly grows by increasing the number of clients. The reason is that a larger number of clients generates a larger number of requests.

Moreover, we note that the interaction between routing and discovery protocols does not affect the overall load: collected values are basically unchanged, with differences in the packets number between the two sets of simulations below 6%; some exceptions have been observed for some scenarios featuring very high pause times, in which the interaction causes an increase of about 10% of the overall packets. This is due to collisions between the unicast traffic generated by clients and the broadcast traffic produced by both clients and servers; in fact, the route discovery process started by the routing protocol needs some time to complete, thus adding time intervals between the transmission of unicast packets, as the clients must wait for a valid route to be built. On the other hand, if clients already have valid routes, they will transmit their packets all at once, causing more collisions and thus increasing the frequency of service advertisement as explained in Subsection 3.1. However, this can be considered a transient effect as in a real scenario other types of traffic may contribute to trigger route discovery mechanisms. We can conclude that the workload due to resource discovery depends in most cases on the application layer regardless of the lower levels.

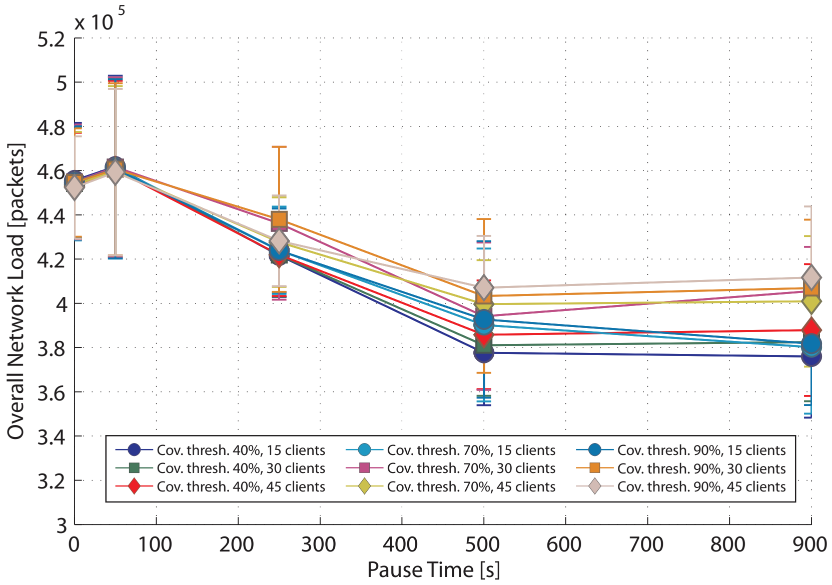
It is important to point out that the traffic due to clients is negligible with respect to the overall load, as shown in Figs. 15 and 16. This is a clear advantage of the approach: it allows a great scalability given that a large number of hosts can share the same set of resources without overloading the network. Moreover, enabling the interaction with the routing protocol, a saving of transmitted packets can be

---

[9] "Intel XScale Microarchitecture". Available at http://www.marvell.com/products/cellular/



(a) Interaction enabled

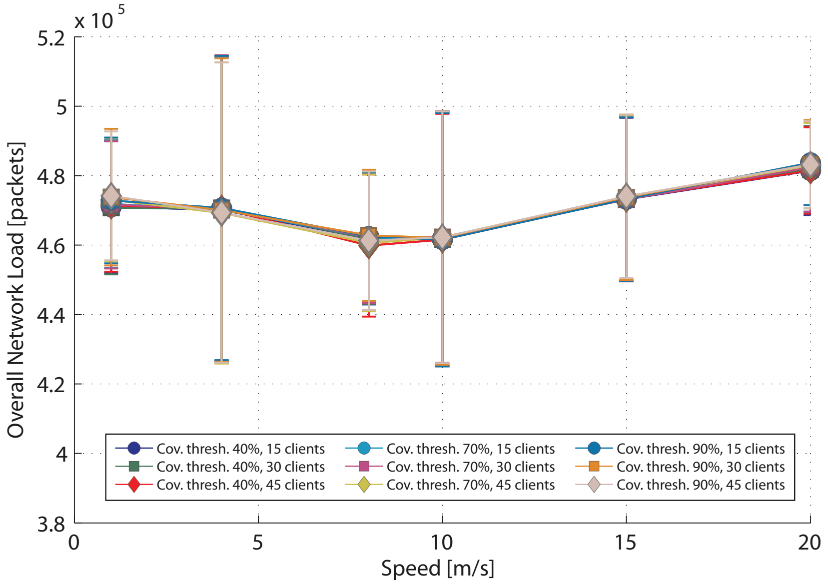


(b) Interaction disabled

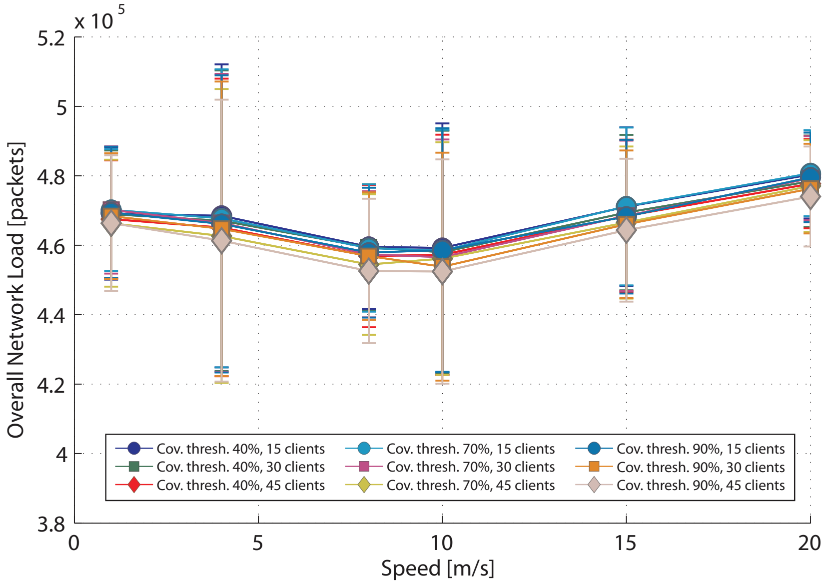
Fig. 13. Resource discovery packets, overall network load (dependence on the pause time, simulation time = 6000 s).

achieved. The reason is that, when the interaction is enabled, network routing is improved and, as a consequence, a smaller number of request relaying is required.

Figs. 17 and 18 report the hit ratio as a function of the pause time and of the maximum speed, respectively. Such figures show that, when the cross-layer interaction is enabled, the percentage of satisfied requests is very high, with values larger than 90%, also considering the challenging 90% covering threshold. When



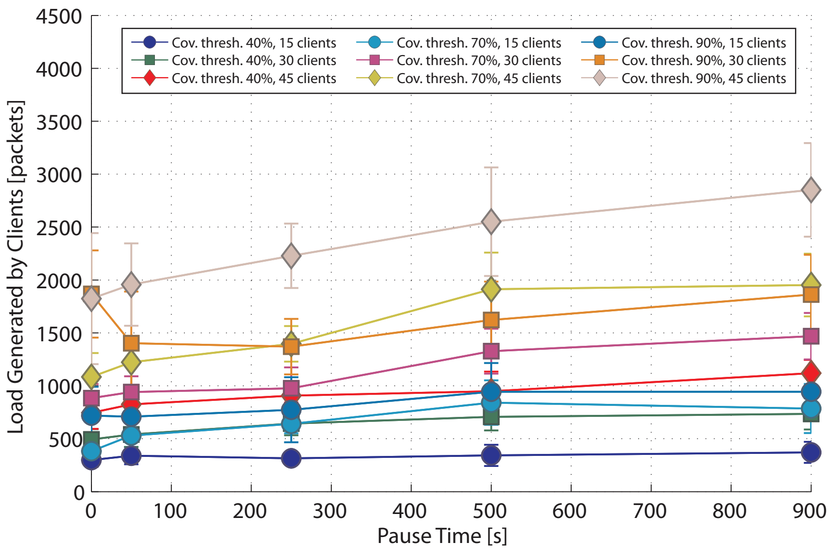
(a) Interaction enabled



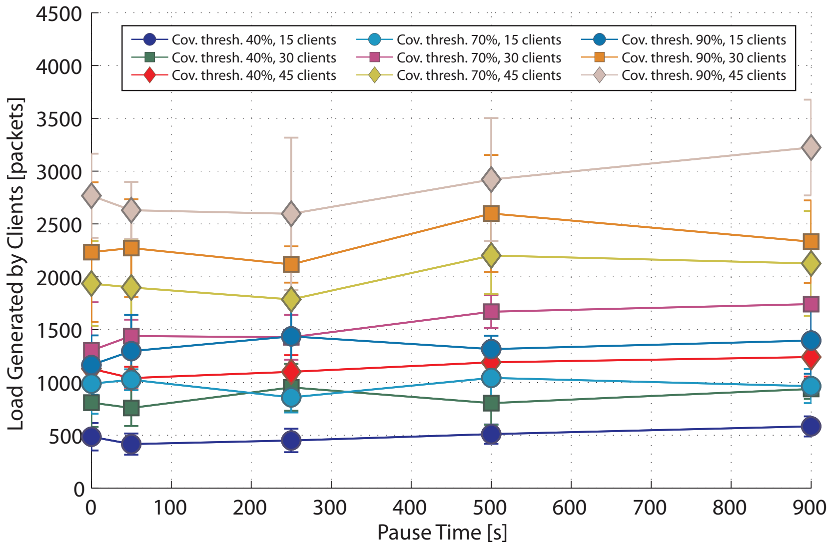
(b) Interaction disabled

Fig. 14. Resource discovery packets, overall network load (dependence on the maximum speed, simulation time = 6000 s).

the cross-layer interaction is disabled, a performance degradation is observed, letting the hit ratio drop down to 85%, with a worst case of 77%, as proved by the error bars. While the cross-layer interaction is enabled, the graphs show a more predictable behaviour (smaller confidence interval). As a general consideration, this result clearly indicates the relevance of the proposed approach. By analyzing results more deeply, we note that the performance gain due to the cross-layer ap-



(a) Interaction enabled

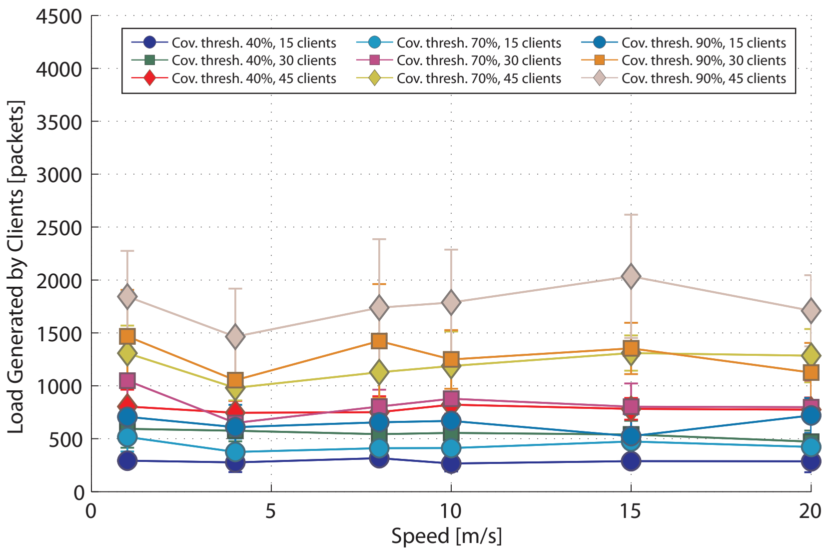


(b) Interaction disabled

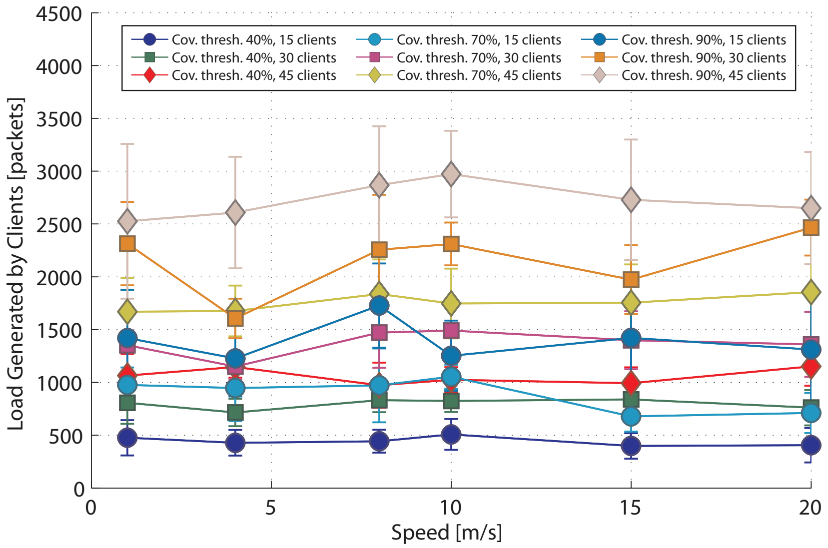
Fig. 15. Resource discovery packets generated by clients (dependence on the pause time, simulation time = 6000 s).

proach is more evident for smaller values of the pause time and larger values of the maximum speed. In other words, the cross-layer approach improves the overall discovery efficiency in scenarios with high mobility. Thus, we can conclude that the cross-layer approach makes the framework less sensitive to the mobility conditions because the resource discovery protocol implicitly disseminates routing information on regular basis using advertisements, thus indirectly making more effective the AODV protocol.

Another feature to look at is the average time elapsed for obtaining a hit (Figs.



(a) Interaction enabled

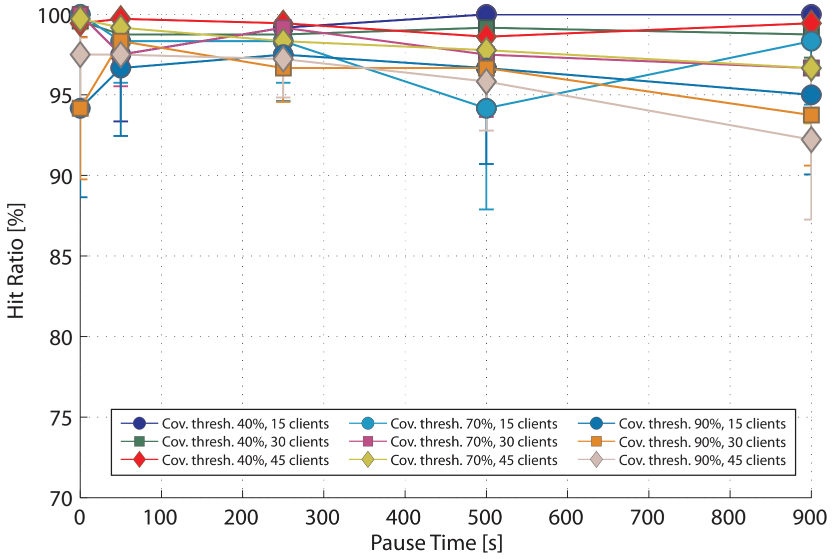


(b) Interaction disabled

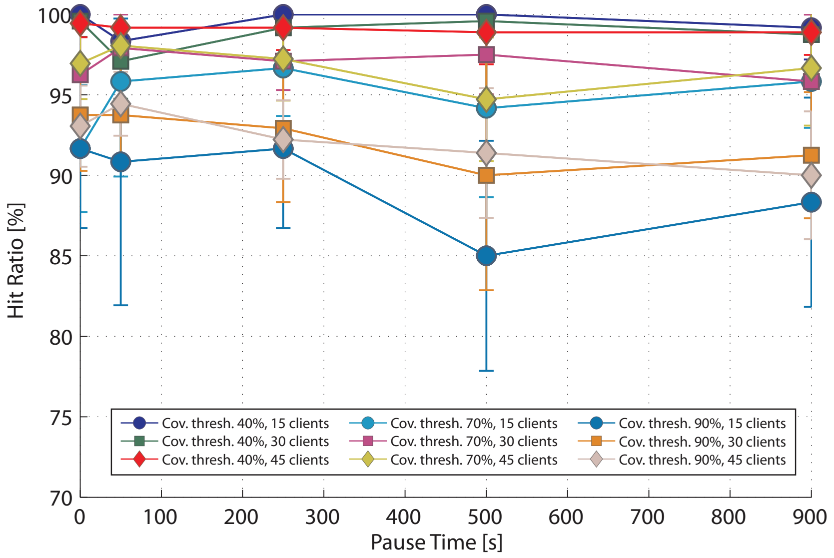
Fig. 16. Resource discovery packets generated by clients (dependence on the maximum speed, simulation time = 6000 s).

19 and 20). In fact, it is straightforward to note that, for a given covering threshold, the elapsed time slightly decreases by increasing the number of clients. This feature enforces the scalability properties of the approach mentioned before. In fact, not only the traffic due to clients is negligible with respect to the overall load, but using a large number of clients improves the discovery responsiveness. The reason is that as the number of clients grows, the dissemination of resource annotations becomes more pervasive, with a positive impact on the average time elapsed to obtain a hit.

As seen so far, the cross-layer approach increases the overall reliability of the



(a) Interaction enabled

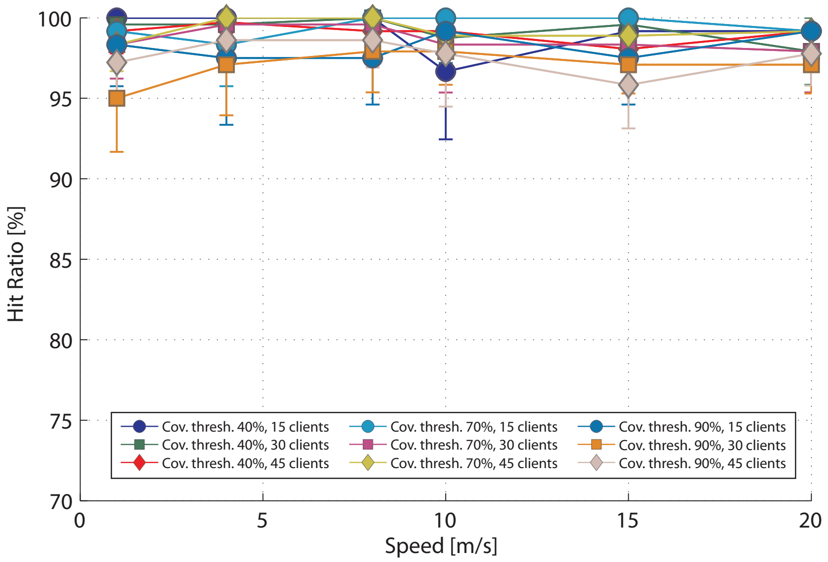


(b) Interaction disabled

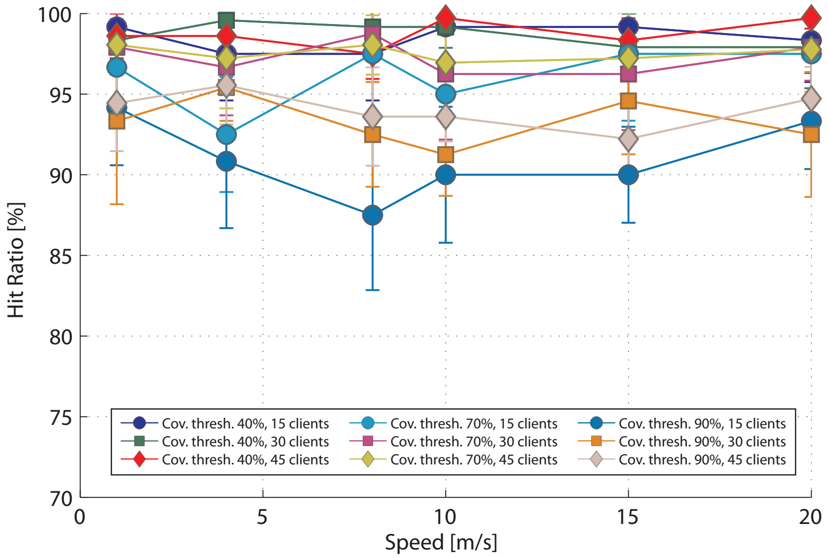
Fig. 17. Hit ratio (dependence on the pause time, simulation time = 6000 s).

resource discovery protocol. It also significantly reduces the routing overhead as shown in Figs. 21 and 22, reporting the packets generated by the AODV protocol. The overall routing overhead is reduced up to one order of magnitude.

Finally, we have analyzed the impact of the number of servers. The general considerations derived above remains still valid by increasing the number of servers. Main differences we have observed are that, increasing the number of servers, improved hit ratios and smaller elapsed times can be obtained, at the expense of a larger network load. We avoid to report all the data, similarly to what done to the



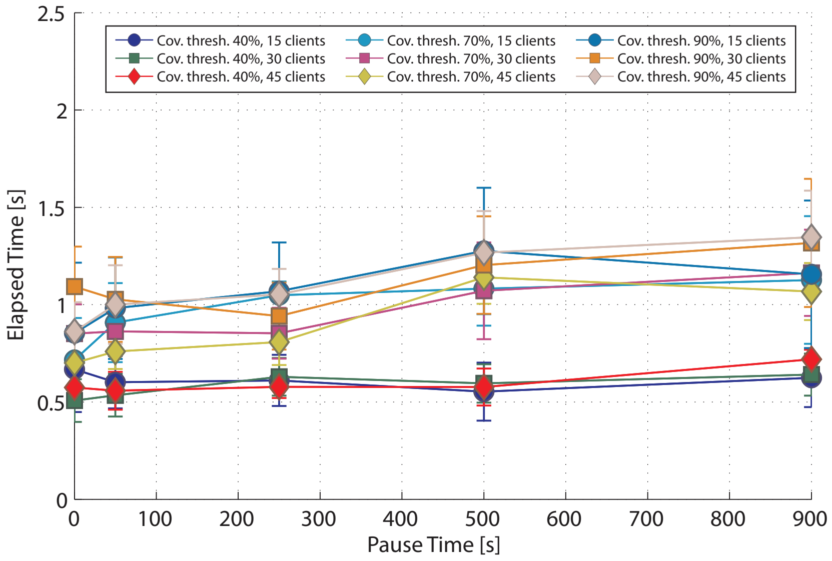
(a) Interaction enabled



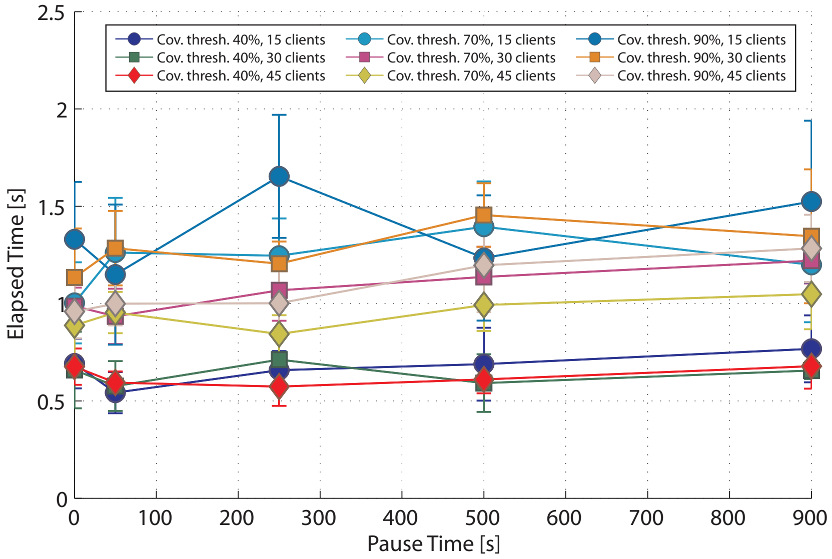
(b) Interaction disabled

Fig. 18. Hit ratio (dependence on the maximum speed, simulation time = 6000 s).

case of 7 servers, but, to give an evidence of the aforementioned considerations, we show here results obtained for 11 servers, when the cross-layer interaction is enabled. Fig. 23 demonstrates that the overall traffic load is higher than in the previous case. The reason is that servers broadcast resource identifiers at regular basis, regardless of the use of such descriptions. Thus the overall traffic is proportional to the number of servers. Also in this case, the traffic due to clients is negligible (see Fig. 24). But, with respect to the case of 7 servers, the traffic due to clients is smaller. In fact, with a larger number of servers, semantic annotations can be



(a) Interaction enabled

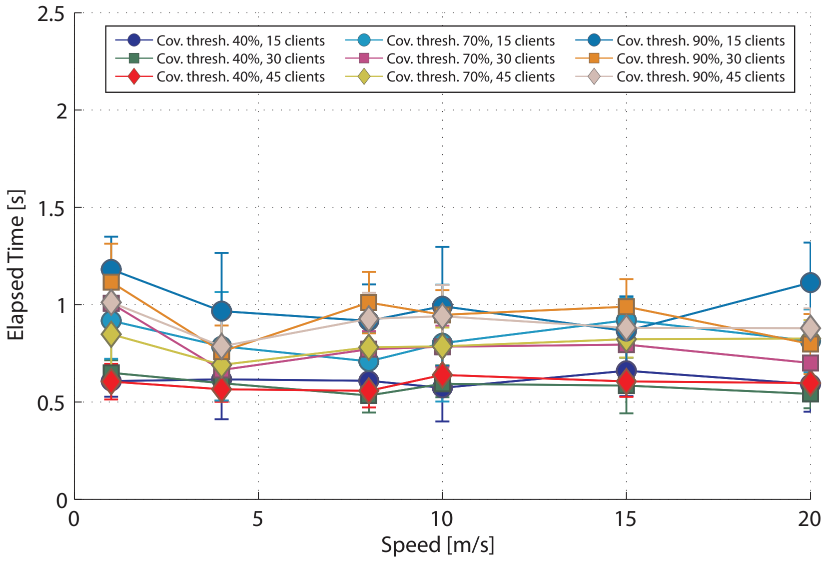


(b) Interaction disabled

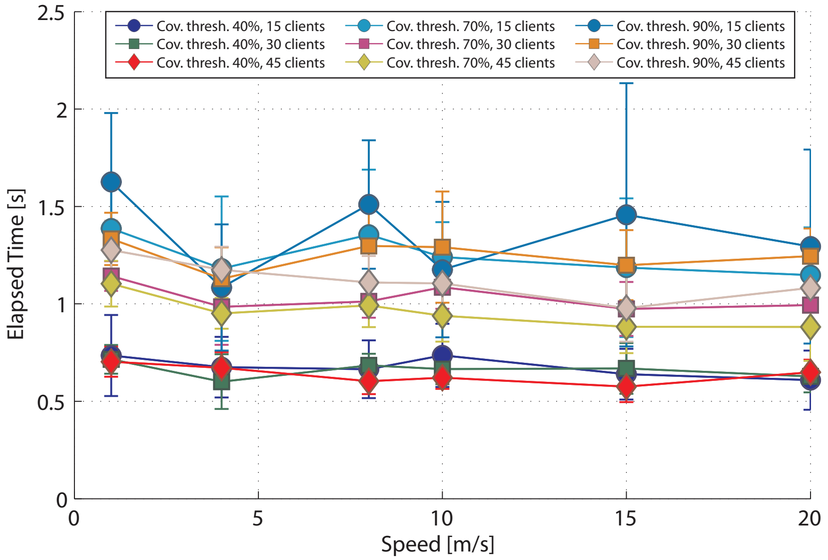
Fig. 19. Average elapsed time for obtaining a hit (dependence on the pause time, simulation time = 6000 s).

downloaded in an easier way, thus producing a smaller amount of traffic. Hit ratios obtained with 11 servers are really impressive. In fact, Fig. 25 shows that almost all requests are satisfied. Moreover, the responsiveness of the framework is improved since the average elapsed time to obtain a hit is smaller than 1.5 s (see Fig. 26). Finally, the number of AODV packets is not influenced by the number of servers (see Fig. 27).

The overall performance evaluation has tested the proposed framework in many



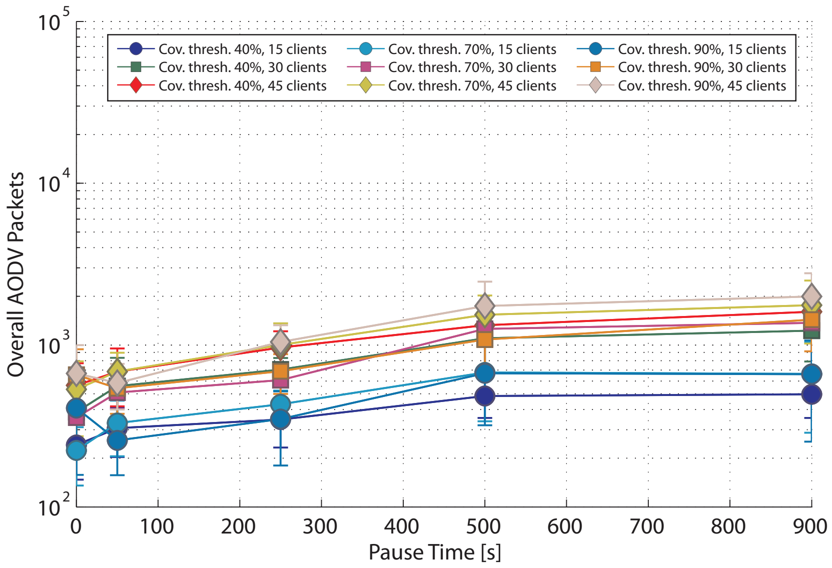
(a) Interaction enabled



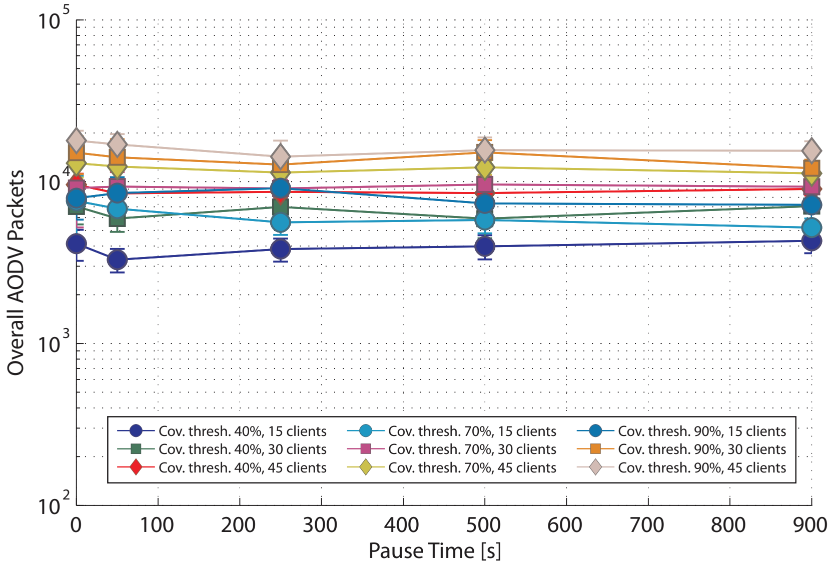
(b) Interaction disabled

Fig. 20. Average elapsed time for obtaining a hit (dependence on the maximum speed, simulation time = 6000 s).

networking scenarios by varying the number of clients/servers, the covering threshold, and the mobility parameters (max speed and pause time). Particularly, results have clearly shown the impact of the number of clients/servers and of the covering threshold. Moreover, several plots evidence the absence of any correlation with respect to mobility parameters. This is a symptom of robustness with respect to time-varying network conditions. Nevertheless, it has been very useful to test the proposed framework in a so broad range of mobility conditions in order to shade as



(a) Interaction enabled



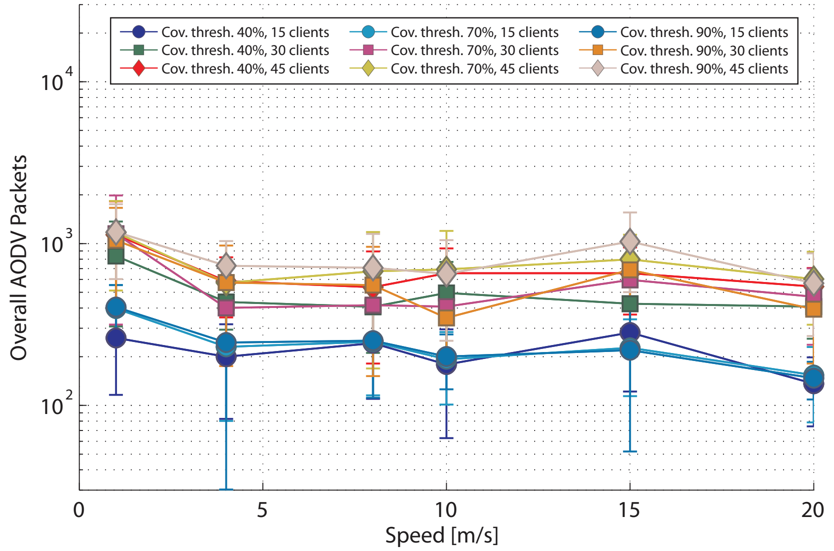
(b) Interaction disabled

Fig. 21. Overall packets generated by AODV (dependence on the pause time, simulation time = 6000 s).

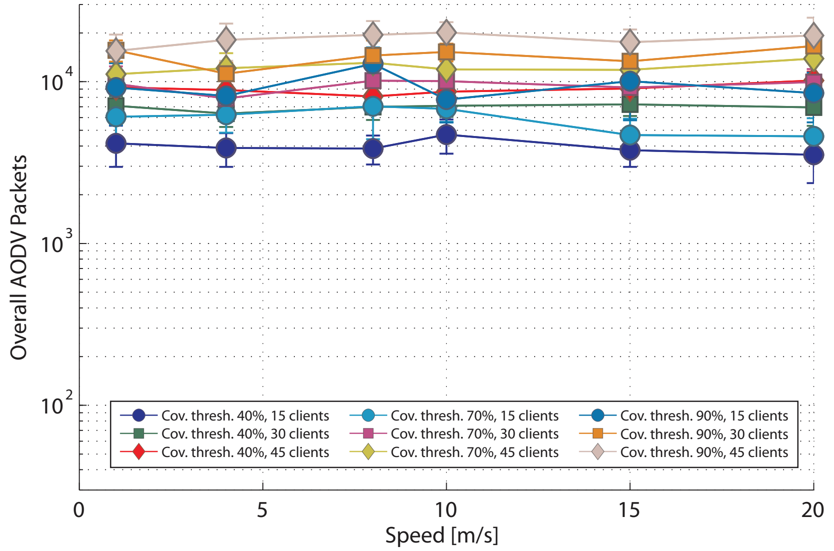
much light as possible on the influence of number of clients/servers and covering threshold.

8 Conclusion

We have proposed an innovative framework to enable the resource discovery, composition and substitution in ubiquitous environments based on IEEE 802.11 ad-hoc



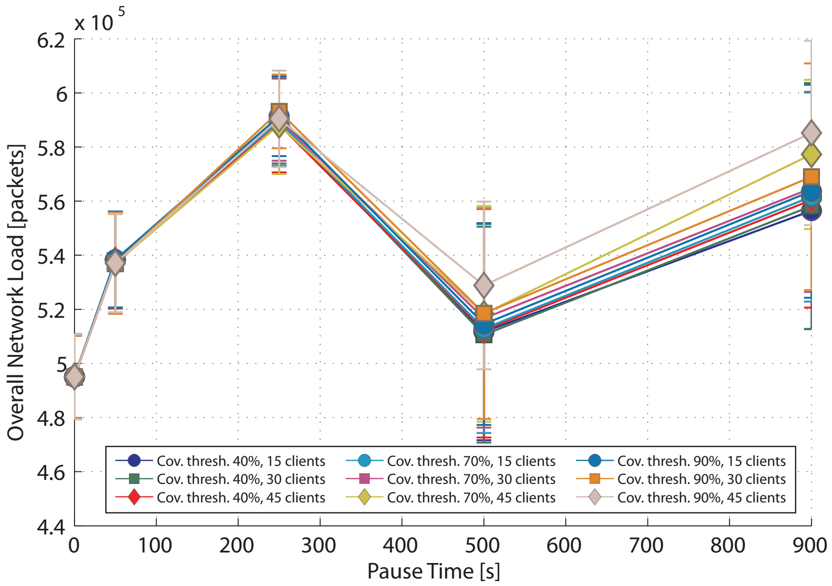
(a) Interaction enabled



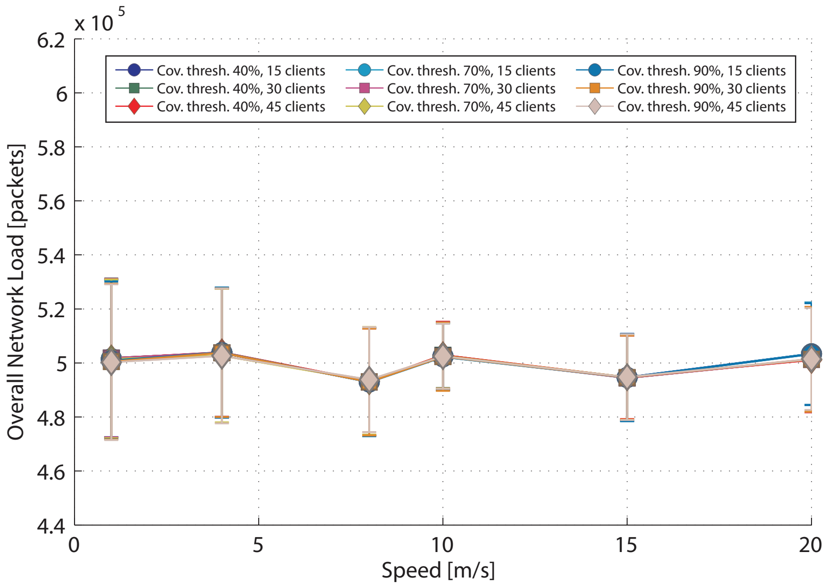
(b) Interaction disabled

Fig. 22. Overall packets generated by AODV (dependence on the maximum speed, simulation time = 6000 s).

networks and exploiting knowledge representation techniques and technologies. The approach allows us to reuse discovery information at network layer in order to enable a cross-layer interaction. At the application level, semantic annotations are exploited to perform the orchestration of elementary resources for building personalized and advanced services and to allow the automatic substitution of unsuitable components. In addition to the fault tolerance feature, the substitutability can also be used as load balancing of resources in case of high concentration of them on a single device or group of devices.



(a) Dependence on the pause time



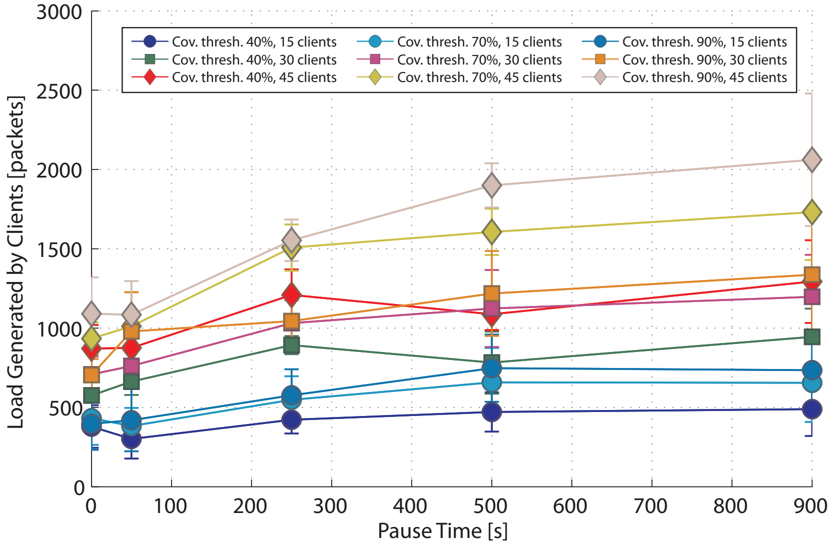
(b) Dependence on the max speed

Fig. 23. Overall network load (11 servers, simulation time = 6000 s).

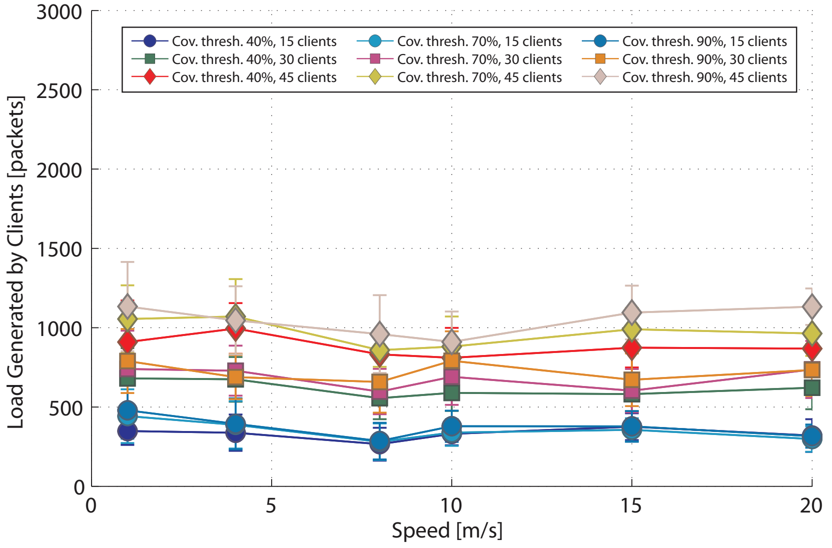
The approach has been tested within a ns-2 simulation environment with reference to a disaster recovery scenario in which a rescue team has to accomplish complex resource discovery operations with the support of a IEEE 802.11 MANET.

Results have clearly highlighted the relevance of our proposal in terms of scalability, hit ratios, and network load due to clients and servers.

We are currently working on DIG and OWL encoding in order to reduce the payload of each packet due to the XML verbosity. Furthermore, future work aims



(a) Dependence on the pause time



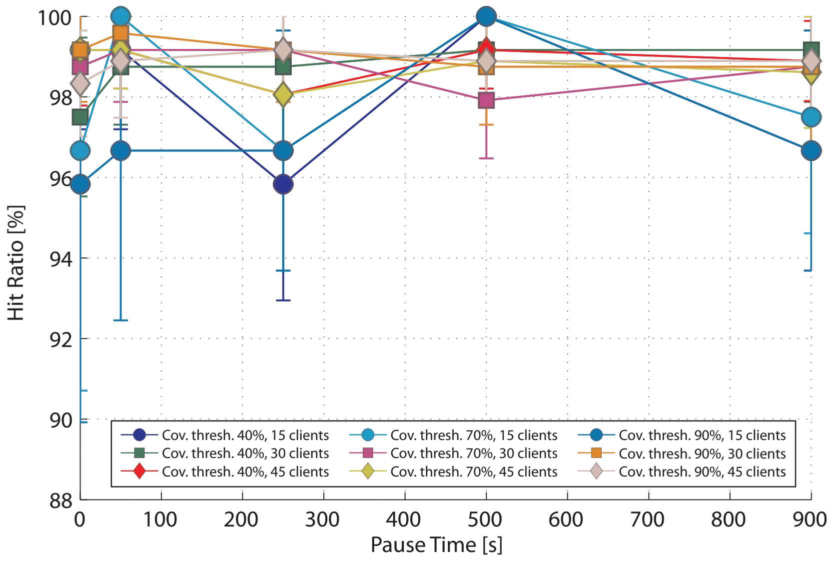
(b) Dependence on the max speed

Fig. 24. Overall load due to clients (11 servers, simulation time = 6000 s).

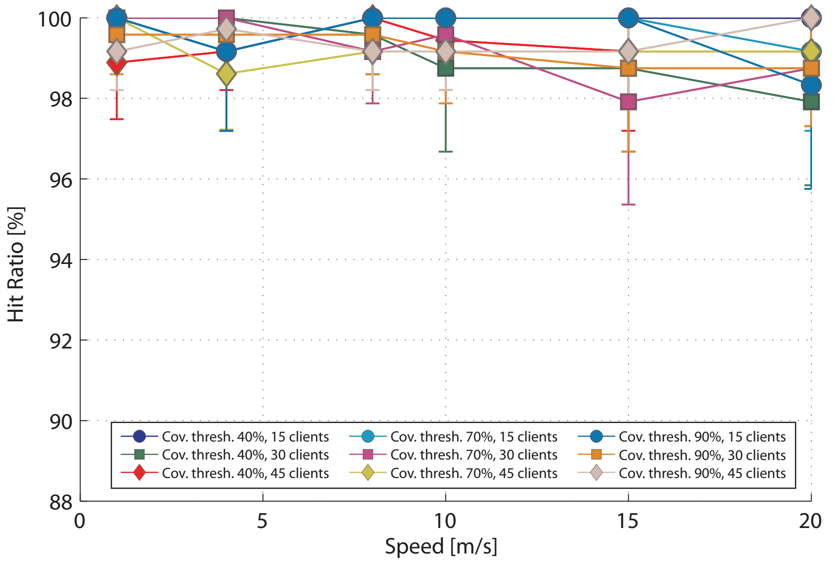
to tune composition and substitution algorithms for devices with reduced memory and processing capabilities as the ones acting as nodes in a mobile ad-hoc network.

References

[1] IEEE 802.11. *Information Technology Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical*



(a) Dependence on the pause time



(b) Dependence on the max speed

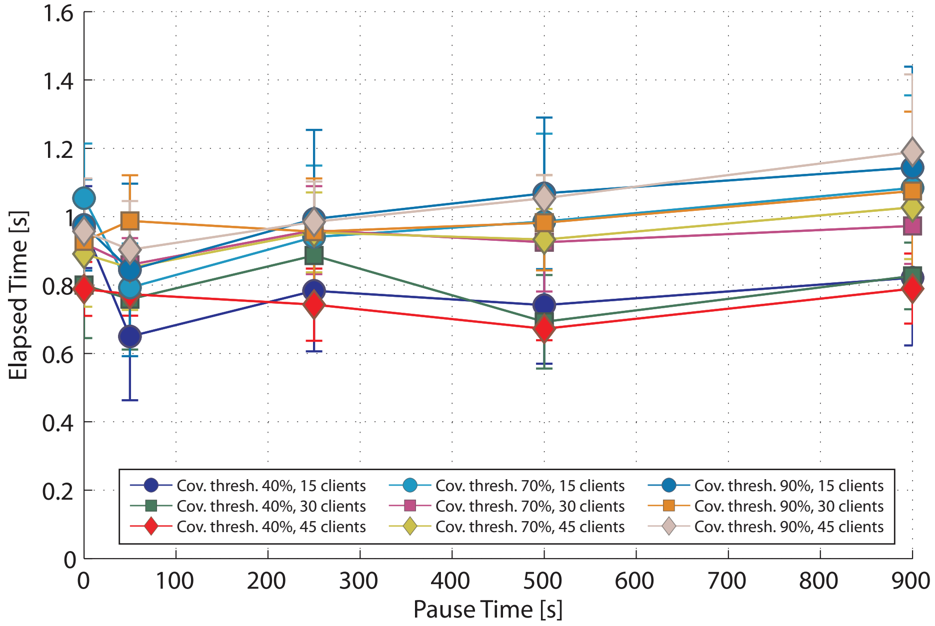
Fig. 25. Hit ratios (11 servers, simulation time = 6000 s).

Layer (PHY) Specifications. ANSI/IEEE Std. 802.11, ISO/IEC 8802-11, first edition, 1999.

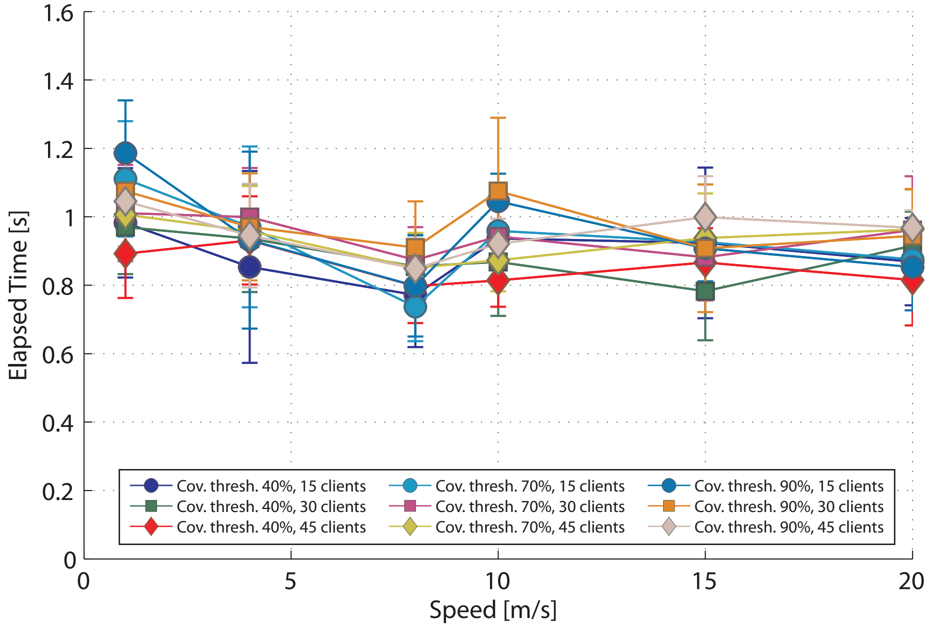
[2] S. Avancha, A. Joshi, and T. Finin. Enhanced Service Discovery in Bluetooth. *IEEE Computer*, 35(6):96–99, June 2002.

[3] F. Baader, D. Calvanese, D. Mc Guinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. Cambridge University Press, 2002.

[4] M. Barbeau. Service Discovery Protocols for Ad Hoc Networking. In *Proceedings of*



(a) Dependence on the pause time



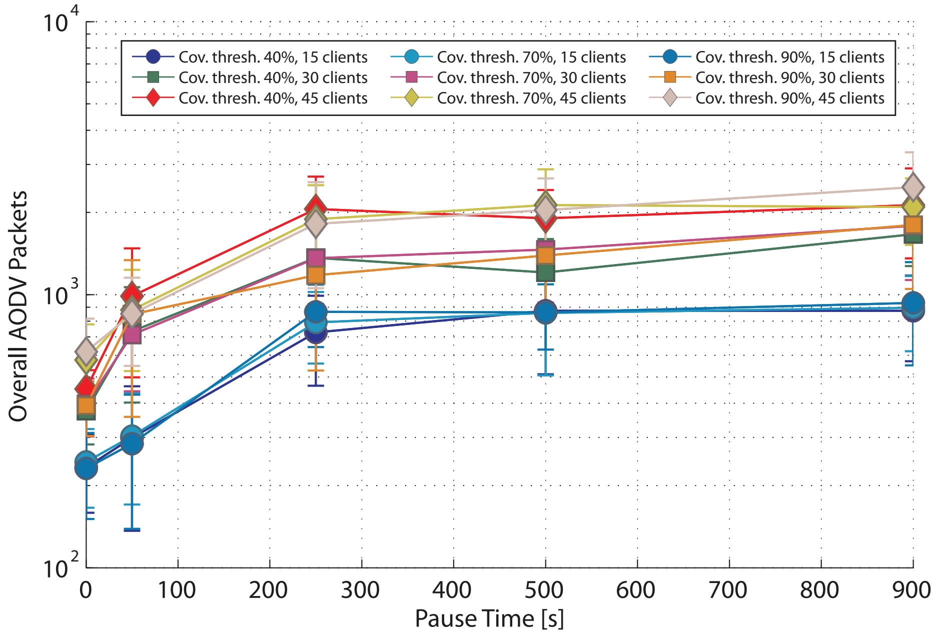
(b) Dependence on the max speed

Fig. 26. Average elapsed time for obtaining a hit (11 servers, simulation time = 6000 s).

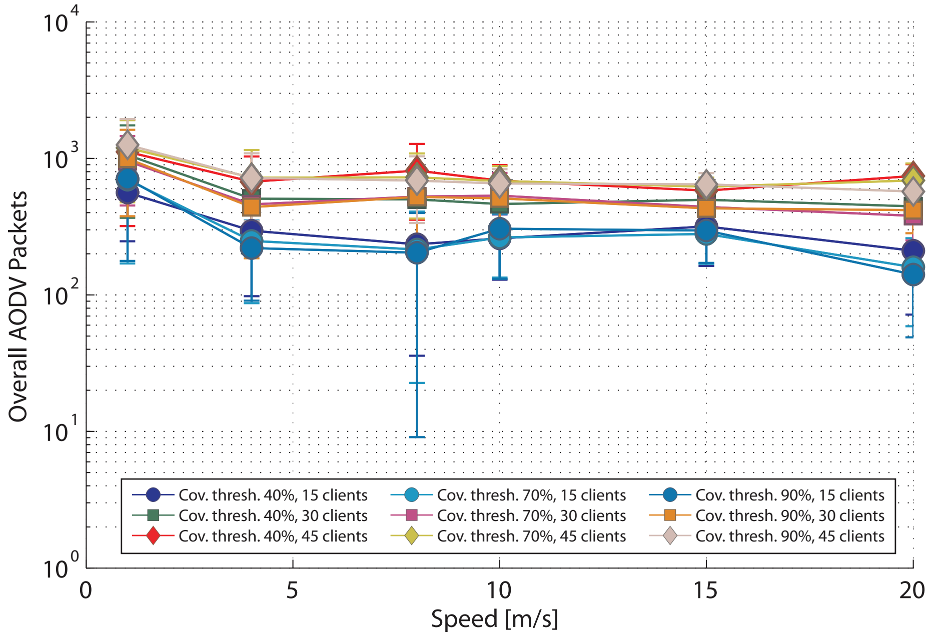
Workshop on Ad-hoc Communications (CASCON ’00), 2000.

[5] S. Bechhofer, R. Möller, and P. Crowther. The DIG Description Logic Interface. In *Proceedings of DL ’03*. CEUR, September 2003.

[6] L. Bordeaux, G. Salaün, D. Berardi, and M. Mecella. When are Two Web Services



(a) Dependence on the pause time



(b) Dependence on the max speed

Fig. 27. Average elapsed time for obtaining a hit (11 servers, simulation time = 6000 s).

Compatible? In *Proceedings of TES*, Lecture Notes in Computer Science, pages 15–28. Springer, 2004.

[7] R. Brachman and H. Levesque. The Tractability of Subsumption in Frame-based Description Languages. In *Proceedings of 4th National Conference on Artificial*

Intelligence (AAAI-84), pages 34–37. Morgan Kaufmann, 1984.

[8] D. Calvanese, G. De Giacomo, and M. Lenzerini. On the Decidability of Query Containment under Constraints. In *Proceedings of 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '98)*, pages 149–158. ACM Press, 1998.

[9] D. Chakraborty, A. Joshi, T. Finin, and Y. Yesha. GSD: A Novel Group-based Service Discovery Protocol for MANETs. In *Proceedings of 4th IEEE Conference on Mobile and Wireless Communications Networks (MWCN)*. IEEE, September 2002.

[10] D. Chakraborty, A. Joshi, T. Finin, and Y. Yesha. Service Composition for Mobile Environments. *Journal on Mobile Networking and Applications*, January 2005.

[11] D. Chakraborty, A. Joshi, and Y. Yesha. Integrating Service Discovery with Routing and Session Management for Ad-hoc Networks. *Ad Hoc Networks Journal*, March 2004.

[12] D. Chakraborty, F. Perich, S. Avancha, and A. Joshi. DReggie: Semantic Service Discovery for M-Commerce Applications. In *Proceedings of Workshop on Reliable and Secure Applications in Mobile Environment*, 2001.

[13] D. Chakraborty, F. Perich, A. Joshi, T. Finin, and Y. Yesha. A Reactive Service Composition Architecture for Pervasive Computing Environments. In *Proceedings of 7th Personal Wireless Communications Conference (PWC)*, October 2002.

[14] D. Chakraborty, Y. Yesha, and A. Joshi. A Distributed Service Composition Protocol for Pervasive Environments. In *Proceedings of Wireless Communcations and Networking Conference (WCNC)*. IEEE, March 2004.

[15] H. Chen, A. Joshi, and T. Finin. Dynamic Service Discovery for Mobile Computing: Intelligent Agents Meet Jini in the Aether. *Cluster Computing*, 4(4):343–354, 2001.

[16] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle. Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks. In *Proceedings of IEEE INFOCOM 2006*, Barcelona, Spain, April 2006.

[17] S. Colucci, T. Di Noia, E. Di Sciascio, F.M. Donini, and A. Ragone. Fully Automated Web Services Orchestration in a Resource Retrieval Scenario. In *Proceedings of IEEE International Conference on Web Services*, pages 427–434, 2005.

[18] S. Colucci, T. Di Noia, A. Pinto, A. Ragone, M. Ruta, and E. Tinelli. A non-monotonic approach to semantic matchmaking and request refinement in e-marketplaces. *International Journal of Electronic Commerce*, 12(2):127–154, 2007.

[19] S. Das, R. Castaneda, J. Yan, and R. Sengupta. Comparative Performance Evaluation of Routing Protocols for Mobile Ad-hoc Networks. In *Proceedings of 7th International Conference on Computer Communications and Networks (IC3N '98)*, pages 153–161, October 1998.

[20] V. De Antonellis, M. Melchiori, B. Pernici, and P. Plebani. A Methodology for e-Service Substitutability in a Virtual District Environment. In *Proceedings of CAiSE '03*, LNCS, pages 552–567. Springer, June 2003.

[21] T. Di Noia, E. Di Sciascio, and F.M. Donini. Semantic matchmaking as non-monotonic reasoning: A description logic approach. *Journal of Artificial Intelligence Research*, 29:269–307, 2007.

[22] T. Di Noia, E. Di Sciascio, F.M. Donini, and M. Mongiello. A System for Principled Matchmaking in an Electronic Marketplace. *International Journal of Electronic Commerce*, 8(4):9–37, 2004.

[23] M. Hörhammer and P. Biswas. A Rule Engine for Location Based Content Syndication. In *Proceedings of MDM '04*, pages 88–93. IEEE, January 2004.

[24] U.C. Kozat and L. Tassiulas. Service discovery in mobile ad hoc networks: an overall perspective on architectural choices and network layer support issues. *Ad Hoc Networks*, 2(1):23–44, January 2004.

[25] C. Lee and S. Helal. Context Attributes: An Approach to Enable Context-awareness for Service Discovery. In *Proceedings of Symposium on Applications and the Internet (SAINT '03)*, pages 22–30, 2003.

[26] J. Liu, Q. Zhang, B. Li, W. Zhu, and J. Zhang. A Unified Framework for Resource Discovery and QoS-Aware Provider Selection in Ad Hoc Networks. *ACM Mobile Computing and Communications Review*, 6(1):13–21, January 2002.

[27] W. Lu, W. K. G. Seah, E. W. C. Peh, and Y. Ge. Communications support for disaster recovery operations using hybrid mobile ad-hoc networks. In *Proceedings of 32nd IEEE Conference on Local Computer Networks*, pages 763–770, 2007.

[28] M. Mecella, B. Pernici, and P. Craca. Compatibility of e-services in a Cooperative Multi-platform Environment. In *Proceedings of TES*, Lecture Notes in Computer Science, pages 44–57. Springer, 2001.

[29] Microsoft Corporation. Understanding Universal Plug and Play - white paper. June 2000.

[30] T. Di Noia, E. Di Sciascio, and F.M. Donini. Extending and computing the concept covering for the semantic web. Technical Report 21/04/S, 2004. Serie Scientifica - Dipartimento di Elettrotecnica ed Elettronica - Politecnico di Bari.

[31] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, 1994.

[32] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. *ACM SIGCOMM*, 24(4), October 2004.

[33] S. Ponnekanti and A. Fox. Application-Service Interoperation without Standardized Service Interfaces. In *Proceedings of PerCom '03*, pages 30–. IEEE, March 2003.

[34] A. Ragone, T. Di Noia, E. Di Sciascio, F.M. Donini, S. Colucci, and F. Colasuonno. Fully Automated Web Services Discovery and Composition through Concept Covering and Concept Abduction. *International Journal of Web Services Research*, 4(3):85–112, 2007.

[35] V. T. Raisinghani and S. Iyer. Cross-layer feedback architecture for mobile device protocol stacks. *IEEE Communications Magazine*, pages 85–92, January 2006.

[36] R. Ramanathan and J. Redi. A brief overview of ad hoc networks: Challenges and directions. *IEEE Communications Magazine*, 40(5):20–22, 2002.

[37] G. Resta and P. Santi. An analysis of the node spatial distribution of the random waypoint model for ad hoc networks. In *Proceedings of ACM Workshop on Principles of Mobile Computing (POMC)*, pages 44–50, Toulouse, France, October 2002.

[38] M. Ruta, T. Di Noia, E. Di Sciascio, and F.M. Donini. Semantic enabled Resource Discovery, Composition and Substitution in Pervasive Environments. In *Proceedings of Mediterranean Electrotechnical Conference - MELECON 2006*, pages 754–760. IEEE, 2006.

[39] M. Ruta, T. Di Noia, E. Di Sciascio, and F.M. Donini. Semantic Based Collaborative P2P in Ubiquitous Computing. *Web Intelligence and Agent Systems*, 5(4):375–391, 2007.

[40] Sun Microsystems. JINI Architecture Specification. November 1999.

[41] S. K. Tan and A. Munro. Adaptive probabilistic epidemic protocol for wireless sensor networks in an urban environment. In *Proceedings of ICCCN*, pages 1105–1110, 2007.

[42] A. Weitzenfeld. From schemas to neural networks: A multi-level modelling approach to biologically-inspired autonomous robotic systems. *Robotics and Autonomous Systems*, 56(2):177–197, 2008.