A logic-based CoAP extension for resource discovery in semantic sensor networks

Michele Ruta, Floriano Scioscia, Giuseppe Loseto, Filippo Gramegna, Agnese Pinto, Saverio Ieva, Eugenio Di Sciascio

Politecnico di Bari
via Re David 200, I-70125
Bari, ITALY
{m.ruta,f.scioscia,agnese.pinto,disciascio}@poliba.it,
{loseto,gramegna,ieva}@deemail.poliba.it

Abstract. Main restraints curbing a deep integration of Semantic Sensor Networks with complex and articulated architectures, basically reside in too elementary allowed discovery capabilities. Several studies agree advanced querying and retrieval mechanisms are needed to truly fulfill the potential of the SSN paradigm. This paper presents a novel SSN framework, supporting a resource discovery grounded on semantic-based matchmaking. Offered contributions are: a backward-compatible extension of Constrained Application Protocol (CoAP) resource discovery; data mining exploitation to detect high-level events from raw data; employment of non-standard inference services for retrieving and ranking resources; adoption of W3C standard SSN-XG ontology to annotate data, events and device features. The effectiveness of the proposed approach is motivated by a case study regarding fire risk prevention and air conditioning control in a university building.

Keywords: Semantic Sensor Networks, CoAP, Resource discovery, Matchmaking, Data mining

1 Introduction

Typically, Wireless Sensor Networks (WSNs) only include homogeneous sensors and are application-dependent and engineering-oriented [1]. This is a strong limit in terms of interoperability and in sight of the integration in large-scale complex architectures. In the mid-2000s semantic technologies were acknowledged as a mean to overcome these issues: in Semantic Sensor Networks (SSNs), *"semantics refers to the critical meaning of sensory data, sensor nodes and application requirements", "effectively enabling the integration, exchange, and reuse of sensory data across various applications and multiple sensor nets"* [1]. Semantics can be leveraged in several aspects: sensory data [1], device management [2], data dissemination and routing [1, 3], query processing [4], application-level services [5].

In latest years, main research studies have shifted toward the integration of SSNs with the Internet and World Wide Web, following the Internet of Things and (Semantic) Web of Things visions [6]. Semantics is devoted to increase both autonomicity and integrability, in order to truly fulfill the potential of the SSN paradigm through enhanced object/subject/event annotation enabling advanced applications. Recent proposals for standard protocols in the field of object networks are gaining acceptance, such as 6LoWPAN and the Constrained Application Protocol (CoAP). Nevertheless, current solutions only allow coarse data-oriented representation and querying mechanisms and still require a significant human intervention for design, deployment and integration of new applications from elementary building blocks, with very similar issues to those affecting Web mashups [7].

In this paper a novel SSN framework is proposed, enabling semantic-based annotation and discovery, by adapting formalisms and technologies borrowed from Semantic Web and Internet of Things research. Data streams, sensors and actuator devices, objects and subjects, high-level events and services can be connoted by a description having a well defined meaning w.r.t. a shared domain conceptualization (*i.e.*, ontology). The proposal is based on slight backward-compatible extensions to CoAP and CoRE Link Format[1] resource discovery protocol for sensor and actor networks. A simple and computationally efficient data mining component permits to detect and annotate high-level events from raw data collected from sensing devices in the field. The SSN-XG ontology of W3C (World Wide Web Consortium) [8] is adopted as reference vocabulary for resource annotations. Non-standard inference services for semantic-based matchmaking [9] allow to retrieve and rank the best matching resources w.r.t. a given request, supporting not only full matches but also approximate ones.

The remainder of the paper is organized as follows. In Section 2 relevant related work is briefly surveyed. The proposed discovery framework is thoroughly outlined in Section 3, also providing details about CoAP extensions and event mining. Section 4 reports on a case study about fire risk prevention and air conditioning control in a university building, in order to highlight features of the proposed approach; finally Section 5 closes the paper.

2 Related work

SSN frameworks are based on reference ontologies to annotate data, devices and services. Many ontology proposals exist, largely varying in aim and scope. OntoSensor [10] and SSN-XG [8] are among the most relevant and widely used ones. Several approaches for publishing sensor data along the Linked Open Data [11] guidelines or via RESTful[2] interfaces are now diffused [12, 7, 13], even as commercial solutions (*e.g.*, Pachube-Cosm, `https://cosm.com/`).

[1] CoRE Link Format, IETF CoRE Working Group Internet-Draft, latest version: 14, 1 June 2012, `http://www.ietf.org/id/draft-ietf-core-link-format-14.txt`

[2] REST: REpresentational State Transfer

From the infrastructure perspective, some interesting protocols are assuming relevance for their lightweight impact on storage and computation. Particularly, CoAP [14] is an alternative to HTTP for interconnected objects, exploiting a binary data representation and a subset of HTTP methods (GET, PUT, POST, DELETE). It follows the REST paradigm for making data and resources accessible. Both 6LoWPAN and CoAP use UDP for transport, as TCP is considered too resource-consuming. 6LoWPAN can be interfaced to IPv6 and CoAP/UDP to HTTP/TCP, so that sensor data can be accessed from the Web.

Recent projects, such as Sense2Web [15] and SPITFIRE [16], combine semantic and networking technologies to build full frameworks, but only allow elementary queries in SPARQL[3] fragments on RDF[4] annotations. More effective techniques such as ontology-based *complex event processing* [17] and *semantic matchmaking* [9] –exploiting logic-based reasoning to support approximated matches, resource ranking and explanation of outcomes– can be used to manage sensory data and events in mobile and pervasive contexts [18–21]. The main issue deriving from the integration of semantics in standard wireless protocols can be inherited by studies in the field of mobile and pervasive computing: *e.g.*, for Bluetooth [18], EPCglobal RFID [21] and ZigBee [20].

Finally, a not negligible related issue is the verbosity of XML-based ontological languages, which calls for devising and implementing efficient compression techniques needed to cope with WSN scenarios, characterized by low throughput of wireless links and strict processing, memory and energy limitations of devices. When evaluating encoding algorithms for such contexts, compression ratio is not enough: processing/memory requirements and efficiency of queries on compressed data become critical parameters, too. High-compression general-purpose approaches such as the PAQ family [22] or XML-specific ones like XMill [23] do not offer the best trade-off. The *EXI* W3C standard (Efficient XML Interchange, http://www.w3.org/XML/EXI/) exploits some basic ideas of XMill, while adding several optimizations. Specific experimental algorithms for the Semantic Web of Things, such as *DIGcompressor* and *COX* [24], aim at either maximum compression ratio or high query efficiency, while keeping the computational costs low.

3 Framework and approach

In what follows the approach we present will be thoroughly described, particularly detailing the proposed CoAP enhancements as well as the semantic matchmaking framework adopted for event mining and resource discovery.

[3] SPARQL Query Language for RDF, W3C Recommendation 15 January 2008, http://www.w3.org/TR/rdf-sparql-query/

[4] Resource Description Framework, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

3.1 CoAP basics

Following the REST architectural style, CoAP adopts a loosely coupled client/server model, based on stateless operations on *resource* representations [14]. Every resource is a server-controlled abstraction, unambiguously identified by a URI (Uniform Resource Identifier). Clients access resources via synchronous request/response interactions, using HTTP-derived methods `GET`, `PUT`, `POST`, and `DELETE` (basically mapping the fundamental Read, Create, Update and Delete operations of data management).

CoAP messages are encoded in a simple binary format. A message consists of a 32-bit *header* followed by *option* fields in Type-Length-Value (TLV) format and a *payload*. The header determines the request method (or response status) and the number of options. Possible request methods, option types and response statuses are distinguished by means of binary codes, listed in CoAP specification[5]. Some CoAP options are derived from HTTP header fields (*e.g.*, content type, headers for conditional requests and proxy support), while some other ones have no counterpart in HTTP. In particular, the target resource URI for a CoAP request (which must refer to the `coap` or `coaps` scheme) is split in a `Uri-Host`, a `Uri-Port` (default UDP port is 5683) and a `Uri-Path` option, with one `Uri-Query` option for each field in the query URI portion. If an option value is longer than 270 B, it is split in consecutive option fields of the same type. Moreover, the `Observe` option[6] allows a client to register w.r.t. the server as an *observer* of the resource, so that the server will notify the client of further changes to the resource state automatically, without the need of polling. This capability is lacking in HTTP; it was introduced in CoAP specifically for scenarios like WSNs, where data have to be monitored over a time span.

In CoAP-based WSN scenarios, each sensor is basically seen as a server, exposing both sensor readings and internal information as resources toward clients, which act on behalf of end-user applications. Different data series will be identified by distinct URIs. Further URIs will identify sensor device status and operating parameters; clients will be able to read or modify them as appropriate. For example, a temperature sensor S can expose the latest temperature reading at the URI `coap://[S-address]/temperature`; in order to access it, a client should issue a `GET` request with `Uri-Host=S1-address` and `Uri-Path=/temperature` options. In case of success, it will receive status code `2.05` and the response message payload will contain the value, *e.g.*, `22.5°C` if it is returned as plain text. Furthermore, since CoAP supports proxies, cluster-head or sink nodes can reply on behalf of a set of (possibly more constrained) sensor nodes deployed in an area, exploiting caching and decreasing the load at the edge of the network. This feature allows also the adoption of data fusion and mining techniques at vari-

[5] Constrained Application Protocol, IETF CoRE Working Group Internet-Draft, latest version: 11, 16 July 2012, `http://www.ietf.org/id/draft-ietf-core-coap-11.txt`

[6] Documented in: Observing Resources in CoAP, IETF CoRE Working Group Internet-Draft, latest version: 5, 12 March 2012, `http://tools.ietf.org/id/draft-ietf-core-observe-05.txt`

ous levels along the path from sensors in the field to nodes managing high-level application logic.

Resource discovery is needed to know what resources a given CoAP server is making available. The CoAP discovery protocol is defined in the CoRE Link Format specification. It allows any host to expose its resources, as well as to act as a directory service for other hosts that want to register their resources. A client will access the reserved URI path `/.well-known/core` on the server with `POST` method to register a resource, or with `GET` to discover available ones. `GET` requests can include `URI-query` fields to retrieve only resources with specific attributes. Standardized query attributes include:

- `href` (hypertext reference): a regular expression to filter resources based on their path (*e.g.*, "/temperature" or "/temperature/*");
- `type` (media type): MIME (Multipurpose Internet Mail Extensions) type/-subtype for the resource;
- `rt` (resource type): an opaque string representing an application-specific meaning of a resource (*e.g.*, "outdoor-temperature");
- `if` (interface description): an opaque string used to provide a name or a URI which indicates what operations can be performed on the resource and their meaning; it typically references a machine-readable document.

Further non-reserved attributes can be freely used. Response payload consists of a comma-separated list of resource paths, each having optionally a list of semicolon-separated attributes.

3.2 Fundamentals of Semantic-based matchmaking

As evident, CoAP resource discovery protocol only allows a syntactic string-matching of attributes, lacking every explicit and formal characterization of the resources semantics. In [9], framework and algorithms were proposed for a logic-based matchmaking between a request and one or more resource descriptions, both expressed using languages grounded on a well known logic. Also a ranking of resource annotations w.r.t. the original request was made possible according on the meaning of descriptions with reference to a shared conceptualization, *i.e.*, an ontology. Description Logics (DL) [25] was the reference formalism and particularly the $\mathcal{ALN}$ (Attributive Language with Unqualified Number Restrictions) DL subset was used, which has polynomial computational complexity for standard and non-standard inferences described hereafter. Given a request Q and a resource R, both consistent w.r.t. a common ontology $\mathcal{T}$ (containing axioms that model knowledge for the reference problem domain), *concept subsumption* [25] standard inference service can be used to identify *full matches*, *i.e.*, resources providing all features requested in Q. Unfortunately, such correspondences are infrequent in practical scenarios involving heterogeneous resources and articulate descriptions. Whenever R is not a full match for Q, *Concept Abduction Problem* (CAP) [9] non-standard inference service allows to determine what should be hypothesized in R in order to completely satisfy Q. The solution H (for *Hypothesis*) to CAP represents "why" the subsumption relation $\mathcal{T} \models R \sqsubseteq Q$ does

not hold. H can be interpreted as *what is requested in Q and not specified in R*. Previous inference services were implemented via structural algorithms based on *Conjunctive Normal Form* (CNF) normalization of concept expressions [26]. Since a concept CNF is unique, a *semantic distance* can be associated to every (Q, R) pair, based on the "size" of the respective CAP solution H. This enables a logic-based relevance ranking of a set of available resources w.r.t. a given request [26].

3.3 Data mining techniques for event annotation

Sensor Networks basically produce large amounts of raw data which have to be collected and interpreted to extract application-oriented information. This is particularly relevant in case of event detection where (semi) automatic procedures are strongly required. The proposed framework includes a simple and resource-efficient data mining process, distributed along several steps and aimed at a semantic-based event annotation from low-level data audit. Each sink device in the SSN collects data from sensors in the field and analyzes them. Whenever an event is detected, a dedicated CoAP resource record is updated by adding a semantic description. The record will also contain extra-logical context parameters such as a timestamp and geographic information about the monitored area (coordinates and size). After detection, the sink will serve as a CoAP gateway, waiting for resource discovery requests coming from client applications searching for either: (a) sensors needed to detect events in the area or (b) actuators that can effectively act upon a given detected event.

As said, the procedure for identification of sensory events via surveys comprises several stages.

1. Data are read from sensors in the field through standard CoAP GET requests, possibly using `Observe` option to be notified of updates. Then a list of elements is built, consisting of three fields: *ID*, storing the identifier of the sensor and therefore the type of data; *value*, where the detected data is stored; *timestamp*. This list will group measurements in time slots of application-defined period T, which are used to compute statistical indexes.

2. For each data set, the average, variance and standard deviation values are computed for the current time slot, to assess the variability of collected information within the monitored area.

3. Statistical indexes of elapsed periods are then exploited to compute an incremental ratio able to evidence trends and significant event changes inside the monitored area.

4. For every data collection, the application defines a binary or multiple classifier, to reveal a situation when given conditions occur. In fact identification is performed by taking into account threshold values for statistical indexes (see Table 2 in the case study section for an extensive explanation).

5. The output of each classifier is mapped to a logic-based expression according to knowledge modeled in the reference ontology. The final semantic description is produced by composing the logical conjunction of all expressions.

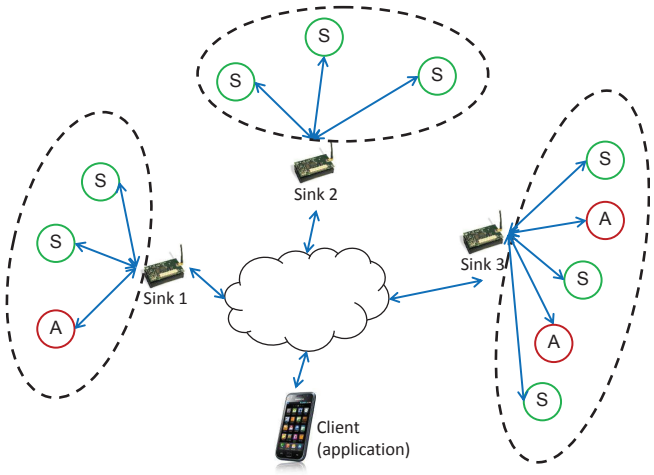

Fig. 1. SSN framework architecture

It is important to note that semantic-enhanced CoAP discovery *per se* does not impose restrictions on where data mining happens, whether in clients running application logic, in sink nodes or in sensors having processing capabilities enough. These three main SSN configurations can even be combined according to application or environmental requirements.

3.4 Advanced resource discovery in Semantic Sensor Networks

The basic SSN architecture is depicted in Figure 1. Each group of sensors and actuators deployed in a given area will communicate through a local *sink* node, acting as a cluster head for resource-constrained devices in the field. Particularly, sink nodes will: (i) allow sensors/actuators to register their semantic annotation as CoAP resources and (ii) embed a lightweight semantic matchmaker [27]. Local or remote applications are CoAP clients and: (i) use semantic-based discovery to search for sensors or actuators, based on annotated descriptions of their properties and capabilities and produced data; (ii) get raw data and/or event descriptions via standard CoAP primitives. The SSN-XG ontology [8] was used as reference model for the selected domain.

In order to support the novel semantic-based resource retrieval, slight extensions were devised to the CoAP discovery protocol outlined in Section 3.1. They are based only on an innovative usage of standard `URI-query` options and on the addition of new ones. Consequently, the resulting framework is still fully backward compatible: servers which do not support semantics will simply reply to requests returning no resource records.

When receiving a request, a CoAP server will start a matchmaking process comparing it w.r.t. all stored annotations referred to the same ontology. The semantic matchmaking is carried out by executing CAP non-standard inference

between request and each resource description, in order to find what elements of the request are lacking in the retrieved resource. A normalized semantic distance in the $[0, 1]$ range is computed for each resource, with lower values for better matches and 0 for full ones. In advanced mobile environments, it is meaningful to involve in a more accurate discovery not only the semantic descriptions, but also data-oriented contextual properties featuring client requests and object/subject/events. In this case an overall *utility function* will be adopted to combine numeric matching with matchmaking results, in order to give a global *score* for resource ranking. Final results will be provided in a normalized ascending $[0, 100]$ % scale, where 100% is the best possible score. Since device and event locations are a key aspect in sensor networks, in the proposed SSN framework the utility function takes into account the geographic distance of each resource from a reference location set in the client request. Hence, a semantic-enhanced CoAP resource is characterized by the following attributes:

- `ro` (reference ontology): it is a new attribute containing the URI of the ontology the resource description refers to. This attribute is mandatory in both resource registration (`POST`) and discovery (`GET`) requests. Its presence allows CoAP servers to discriminate between semantic-enhanced and standard requests.
- `rt`: it is a standard attribute which maintains, instead of an opaque string, the annotated request or the resource description in case of registration or discovery response. It is a concept expression OWL (Web Ontology Language) exploiting the RDF/XML syntax, although the framework does not impose restrictions. In order to cope with the verbosity of XML-based languages, the annotation is compressed and then encoded in base64 for string format compatibility. In the adopted settings *gzip* compression is used, but other schemes can be also employed, such as *EXI*[7] W3C standard or even experimental algorithms for the Semantic Web of Things such as *COX* [24]. Even with compression, an annotation might still be longer than the $270B$ limit of the `URI-query` option field. In such case, multiple `URI-query` fields will be present with `rt` name and the full annotation will come joining such values.
- `at` (annotation type): it is a new attribute indicating ontology language, syntax and encoding scheme of semantic annotation. It is defined in the same way as the standard `type` (media type) attribute. For each language-syntax-encoding, the related MIME type should be added to the CoAP Media Type Registry, which maps MIME type strings to 16-bit codes. The `30004` unassigned code was adopted for the `application/rdf+zip` MIME type used in this work.
- `st` (semantic task): it indicates which kind of reasoning task is required for resource discovery. Each provided inference is identified by a numeric code. At the moment, `1` is assigned to CAP.
- `sr` (semantic threshold): this new attribute is used in discovery requests to specify a minimum score threshold. Resources having an overall score w.r.t.

[7] Efficient XML Interchange, http://www.w3.org/XML/EXI/

the request lower than this threshold will not be returned to the client. This allows to modulate the granularity of discovery and to limit data transfers when many resources are available. In replies to discovery, this field contains the overall score of a resource w.r.t. the request.

- `lg` (longitude) and `lt` (latitude): they are two novel and optional attributes (expressed in degrees). Within a request, they specify a reference geographical location in order to measure distances of discovered resources and grade matchmaking outcomes; in replies or registrations they simply express the resource location.
- `md` (maximum distance): in a request it indicates the maximum acceptable distance (in meters) from the reference location set with the pair $< lg, lt >$. The adoption of a (center,distance) constraint allows the server to pre-filter resources, so avoiding the relatively expensive semantic matchmaking for resources outside the requested area. In replies, the attribute is used to specify the actual distance of a resource from the reference location.

Some toy examples will clarify both structure and content of registration, request and reply messages in the semantic-enhanced variant of CoAP protocol. Semantic annotations will be voluntarily omitted here for the sake of clarity: several examples will be provided in the case study report in Section 4.

Registration. A sensor acting as a CoAP server wants to expose a resource representing its own features and capabilities. The corresponding semantically annotated description is expressed in OWL/RDF language w.r.t. SSN-XG ontology and compressed with gzip. The sensor has (latitude,longitude) coordinates of $(41.109, 16.878)$. The sensor will therefore issue a `POST` CoAP request to:

```
coap://localhost:5683/.well-known/core?rt=xxxxxx
&ro=http://purl.oclc.org/NET/ssnx/ssn&at=30004&lg=16.878&lt=41.109
```

Discovery request. An application queries an SSN sink having 193.204.59.75 IP address to find sensors best matching a semantic description expressed in OWL/RDF language w.r.t. Ontosensor ontology and compressed with gzip. The application is interested only in sensors located within 100m from the location at $(41.1566734, 16.884755)$ coordinates. Furthermore, it sets a score threshold of 70% to retrieve only good matches. The application will therefore send a `GET` CoAP request to:

```
coap://193.204.59.75:5683/.well-known/core?
&ro=http://www.memphis.edu/eece/cas/onto_sensor/OntoSensor.txt
&rt=yyyyyy=&at=30004&lg=41.1566734&lt=16.884755&md=100&st=1&sr=70
```

Discovery reply. Let us suppose that two sensors match the above request. The CoAP server response payload will be:

```
</HumidSens204>;ct=0;ct=41;at=30004;lg=41.1566735;lt=16.884754;
md=19.4;ro=http://www.memphis.edu/eece/cas/onto_sensor/OntoSensor.txt;
rt=aaaaaaa;sr=76.4;title="Humidity-Sensor-204",
</HumidSens111>;ct=0;ct=30004;lg=41.1566732;lt=16.884756;
md=60.9;ro=http://www.memphis.edu/eece/cas/onto_sensor/OntoSensor.txt;
rt=bbbbbbb;sr=82.1;title="Humidity-Sensor-111"
```

4 Case study

In what follows, illustrative examples are presented to better explain flexibility and potentialities our approach provides and to let its novelty emerge. The proposed enhancement has been applied and tested in two different case studies: (i) fire risk prevention; (ii) air conditioning. Both focused on an environment composed of three rooms of the Technical University of Bari equipped with several devices. All of them refer to one or more sink nodes representing the interface of the SSN toward external applications. The CoAP server runs on a sink node based on Android 2.3.3 platform. It is able to accept `GET/POST` messages –for example a sensor discovery request– and send responses to clients. For our experiments the *Californium CoAP framework* [28] was extended with the semantic-based enhancement proposed in Section 3.4. *Copper plugin* [29] for Firefox was used to simulate the requests coming from applications.

Device arrangement is shown in Figure 2: the rooms contain different kinds of sensors –in green– and actuators –in red. Sensor and actuator descriptions are represented by conjunctive concept expressions referring to the same ontology, which extends SSN-XG [8]. In particular, sensors are described by means of their properties and capabilities, whereas actuator descriptions also include features of the event for which they should be activated. Figure 3 shows an example of temperature sensor modeling. We exploited the pattern defined in [8] to describe the measuring features of a sensor, with some differences. In particular, each sensor can "observe" properties modeled as subclasses of `ssn:FeatureOfInterest` and has proper measurement capabilities expressed as subclasses of the `ssn:MeasurementCapability` class. Each specific subclass of `ssn:MeasurementCapability` has a set of measurement properties, represented as subclasses of the `ssn:MeasurementProperty` class. Furthermore, a sensor is related to a subclass of `ssn:EnergyDevice` through the `ssn:hasSubSystem` property to model its energy source.

The first scenario refers to disaster prevention, focusing on discovering possible fire risks in given areas. Thanks to a continuous monitoring of sensed parameters, possible hazards can be quickly detected and recovery procedures rapidly started to take danger under control for both people and structures. The first step is to discover sensors in the environment able to monitor useful data. Application-defined sensor requirements play the role of "request" and the device descriptions the ones of "supplied resources". Let us suppose *a temperature sensor is needed in room C with a large measurement range –able to take both low and high values of temperature–, a medium sampling frequency and low accuracy and resolution –temperature changes quickly during a blaze, so high accuracy and resolution are not required. At the same time, the application requires that sensor has a battery with medium lifetime.* Using concepts defined in the domain ontology the request can described as follows:

$$(\mathbf{R_1})\mathbf{FireRisk\_Request} \equiv TemperatureSensor \sqcap \exists\, hasMeasurementCapability \sqcap \forall\, hasMeasurementCapability.(\exists\, hasMeasurementProperty \sqcap \forall\, hasMeasurementProperty.(HighMeasurementRange \sqcap MediumFrequency \sqcap LowAccuracy \sqcap LowResolution)) \sqcap \exists\, hasSubSystem \sqcap \forall\, hasSubSystem.(Battery \sqcap \forall\, hasSurvivalProperty.(MediumBatteryLifetime)).$$

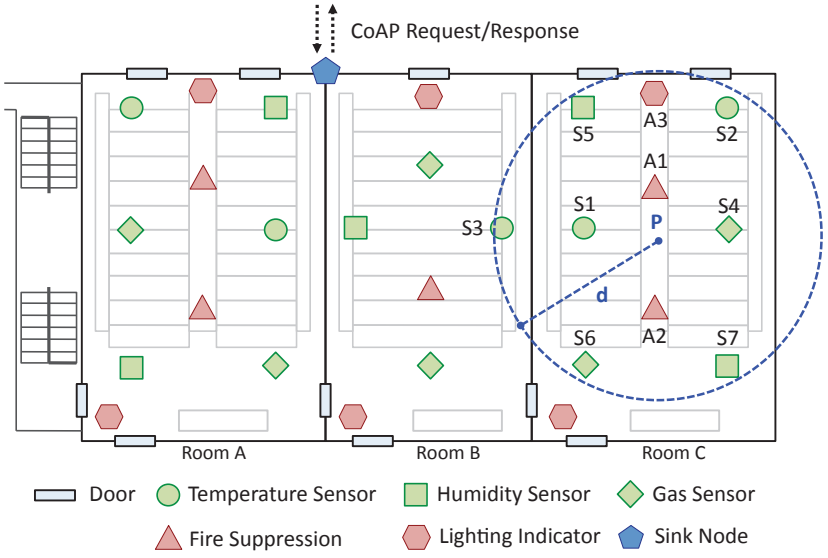

Fig. 2. Rooms map

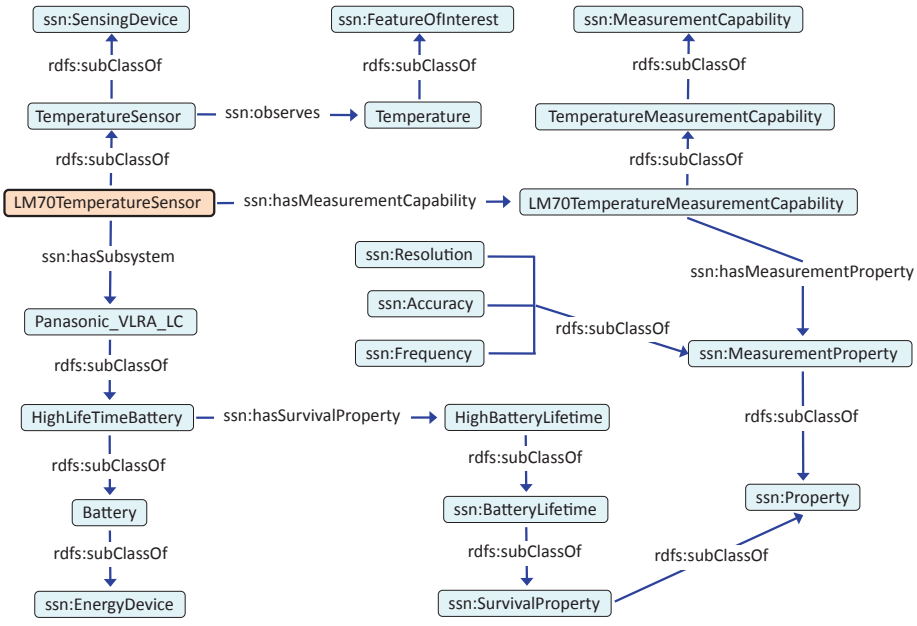

Fig. 3. Temperature sensor modeling

Notice that a sensor could also have more subsystems. In that case, related features are conjunctive concept expressions in the filler of the `hasSubSystem` property. With reference to contextual query attributes in the `GET` request, let us suppose to select devices positioned in **Room C** with a maximum distance of **25m** from the reference location P and a threshold discovery score of **65%**. The sink acts as a CoAP gateway w.r.t. deployed sensors. It executes a pre-processing step to exclude components outside the user-specified range. Hereafter the concept expressions for some of the sensor instances inside the measurement area in Figure 2 are summarized.

($\mathbf{S_1}$)**TSic306TemperatureSensor** $\sqsubseteq TemperatureSensor \sqcap \exists\, hasMeasurementCapability \sqcap \forall\, hasMeasurementCapability.(Tsic306TemperatureMeasurementCapability) \sqcap \exists\, hasSubSystem \sqcap \forall\, hasSubSystem.(EnixEnergies\_RS\_689)$.

Tsic306TemperatureMeasurementCapability $\equiv \exists\, hasMeasurementProperty \sqcap \forall\, hasMeasurementProperty.(MediumResolution \sqcap HighAccuracy \sqcap MediumFrequency \sqcap LowMeasurementRange \sqcap MediumPrecision)$.

EnixEnergies_RS_689 $\sqsubseteq Battery \sqcap \forall\, hasSurvivalProperty.(LowBatteryLifetime)$.

($\mathbf{S_2}$)**LM70TemperatureSensor** $\sqsubseteq TemperatureSensor \sqcap \exists\, hasMeasurementCapability \sqcap \forall\, hasMeasurementCapability.(LM70TemperatureMeasurementCapability) \sqcap \exists\, hasSubSystem \sqcap \forall\, hasSubSystem.(Panasonic\_VRLA\_LC)$.

LM70TemperatureMeasurementCapability $\equiv \exists\, hasMeasurementProperty \sqcap \forall\, hasMeasurementProperty.(LowResolution \sqcap LowAccuracy \sqcap MediumFrequency \sqcap HighMeasurementRange)$.

Panasonic_VRLA_LC $\sqsubseteq Battery \sqcap \forall\, hasSurvivalProperty.(HighBatteryLifetime)$.

($\mathbf{S_3}$)**SE95TemperatureSensor** $\sqsubseteq TemperatureSensor \sqcap \exists\, hasMeasurementCapability \sqcap \forall\, hasMeasurementCapability.(SE95TemperatureMeasurementCapability) \sqcap \exists\, hasSubSystem \sqcap \forall\, hasSubSystem.(Philips\_FR\_6LB)$.

SE95TemperatureMeasurementCapability $\equiv \exists\, hasMeasurementProperty \sqcap \forall\, hasMeasurementProperty.(HighResolution \sqcap HighAccuracy \sqcap HighFrequency \sqcap HighMeasurementRange)$.

Philips_FR_6LB $\sqsubseteq Battery \sqcap \forall\, hasSurvivalProperty.(MediumBatteryLifetime)$.

($\mathbf{S_4}$)**MX6GasSensor** $\sqsubseteq GasSensor \sqcap \exists\, hasMeasurementCapability \sqcap \forall\, hasMeasurementCapability.(MX6GasMeasurementCapability) \sqcap \forall\, hasSubSystem.(Panasonic\_VRLA\_LC) \sqcap \exists\, hasSubSystem.$

MX6GasMeasurementCapability $\equiv \forall\, hasMeasurementProperty.(HighResolution \sqcap MediumAccuracy \sqcap HighFrequency \sqcap MediumPrecision) \sqcap \exists\, hasMeasurementProperty.$

For each device annotation, the sink node apply CAP resolution w.r.t. R_1 in order to evidence requested capabilities missing in the device structure. For example, S_2 has a different battery lifetime w.r.t. the request, therefore it is a nearly full match. On the contrary, S_3 refers to a different type of sensor and does not match the required resolution, accuracy and frequency constraints.

$\mathbf{H_{(R_1,S_2)}} \equiv \forall\, hasSubSystem.(\forall\, hasSurvivalProperty.(MediumBatteryLifetime))$.

$\mathbf{H_{(R_1,S_3)}} \equiv \forall\, hasMeasurementCapability.(\forall\, hasMeasurementProperty.(MediumFrequency \sqcap LowAccuracy \sqcap LowResolution))$.

Afterwards, the sink replies to the client request with the list of suitable sensors in relevance order. The arrangement score is computed via the following utility function:

$$f(R,S) = 100 * \left[1 - \frac{s\_match(R,S)}{s\_match(R,\top)} * \left(1 + \frac{distance(R,S)}{d}\right)\right]$$

where s_match measures the CAP-induced semantic distance between a request R and resource description S; this value is normalized dividing by the semantic distance between R and the universal concept –*Top* or *Thing*– which depends only on axioms in the ontology. Geographical distance –normalized by user-specified maximum range d– is combined as weighting factor. The top results

ID	URI	Distance	Semantic Rank	Final Rank
S_2	`/sink/LM70TemperatureSensor`	20.53 m	0.050	90.89 %
S_3	`/sink/SE95TemperatureSensor`	23.20 m	0.075	85.53 %
S_1	`/sink/TSicTemperatureSensor`	13.70 m	0.125	80.64 %
S_4	`/sink/MX6GasSensor`	9.60 m	0.225	68.85 %

Table 1. Discovery outcomes in case of fire risk prevention

Fire event	Fire risk	Ignition	Propagation	Flash over	Conflagration
temp $(^{\circ}C)$	$40 \div 60$	$60 \div 75$	> 75	> 120	> 120
$\frac{\Delta \mathbf{temp}}{\Delta \mathbf{t}}$	> 0	> 1	> 2	$5 \div 7$	> 7

Table 2. Example of fire hazard event detection criteria

of discovery are reported in Table 1. Sensor $S2$ has the best rank because its description is very similar to the request, in fact it has a semantic distance of 0.050. Nevertheless, its final score is about 90% because it is 20m far from the reference location P.

Now the application can select sensors to query/observe; subsequently it can apply mining procedures on retrieved data –as explained in Section 3.3– and even detect risk events. For example, let us suppose that temperature sensor S_2, having a $f = 0.5Hz$ sampling rate, produces the following data point series in Celsius degrees: $(22.3, 22.5, 25.5, 29.4, 38.7, 49.3, 60.4, 75.5)$. Let us also suppose the fire prevention application adopts an observing window $T = 8s$. Then the following data mining steps are executed:
– Data are divided in $N = fT = 4$-sample blocks: $B_1 = (22.3, 22.5, 25.5, 29.5)$; $B_2 = (38.7, 49.3, 60.4, 75.5)$.
– Average, variance and standard deviation are computed: $\mu_1 = 24.9$; $\sigma_1^2 = 8.3$; $\sigma_1 = 3.3$; $\mu_2 = 56.0$; $\sigma_2^2 = 187.9$; $\sigma_2 = 15.8$.
– Incremental ratio is $\frac{\Delta f}{\Delta t} = \frac{\mu_t - \mu_{t-1}}{T}$ therefore $\frac{\Delta temp}{\Delta t} = \frac{56.0-24.9}{8} = 3.9$.
– Based on studies and laws on fire risk prevention, we designed a classifier able to detect one of the five common fire stages, reported in Table 2. In the example, the classifier detects that *fire propagation* is occurring in the environment. It is useful to point out that the example just used the average temperature for the sake of clarity; in the case study, temperature variance, relative humidity, CO and CO_2 concentrations are also taken into account.

The event description then becomes a query for the actuator discovery phase. In this way, the application can find all devices able to prevent a given dangerous event or perform recovery procedures. Each actuator can be modeled defining, in addition to operating specification, also the context description that, if verified, should lead to an activation. Different kinds and stages of fire events have been modeled mainly through temperature, extension, propagation speed and toxicity parameters. For example, detection of a propagation event can be characterized by high temperature, moderate extension, low propagation speed and moderate toxicity. In such case, actuator request and fire suppressor devices can be described as:

ID	URI	Distance	Semantic Rank	Final Rank
A_2	/sink/FireSuppressionTypeB	11.01 m	0.000	100.00 %
A_1	/sink/FireSuppressionTypeA	5.41 m	0.150	81.75 %

Table 3. Actuator discovery results

ID	URI	Distance	Semantic Rank	Final Rank
S_3	/sink/SE95TemperatureSensor	23.20 m	0.050	90.35 %
S_1	/sink/TSicTemperatureSensor	13.70 m	0.075	88.38 %
S_2	/sink/LM70TemperatureSensor	20.53 m	0.125	77.23 %
S_4	/sink/MX6GasSensor	9.60 m	0.200	72.31 %

Table 4. Sensor discovery results in case of HVAC control

$(\mathbf{R_2})\mathbf{Actuator\_Request} \equiv \exists\ hasTemperature \sqcap \forall\ hasTemperature.(HighTemperature) \sqcap \exists\ hasToxicity \sqcap \forall\ hasToxicity.(ModerateToxicity) \sqcap \exists\ hasExtension \sqcap \forall\ hasExtension.(GrowingExtension) \sqcap \exists\ hasSpeed \sqcap \forall\ hasSpeed.(LowSpeed)$.

$(\mathbf{A_1})\mathbf{FireSuppressorTypeA} \sqsubseteq FireSuppression \sqcap \exists\ hasToxicity \sqcap \forall\ hasToxicity.(HighToxicity) \sqcap \exists\ hasExtension \sqcap \forall\ hasExtension.(WideExtension) \sqcap \exists\ hasSpeed \sqcap \forall\ hasSpeed.(HighSpeed) \sqcap \exists\ hasTemperature \sqcap \forall\ hasTemperature(HighTemperature)$.

$(\mathbf{A_2})\mathbf{FireSuppressorTypeB} \sqsubseteq FireSuppression \sqcap \exists\ hasToxicity \sqcap \forall\ hasToxicity.(ModerateToxicity) \sqcap \exists\ hasExtension \sqcap \forall\ hasExtension.(GrowingExtension) \sqcap \exists\ hasSpeed \sqcap \forall\ hasSpeed.(LowSpeed) \sqcap \exists\ hasTemperature \sqcap \forall\ hasTemperature.(HighTemperature)$.

Results of the actuator discovery phase are reported in Table 3. A_2 completely satisfies the request and consequently produces an overall score of 100%. Comparing the two fire suppressor actuators, it can be seen A_1 presents a lower final rank due to its different specification in terms of detected temperature, toxicity and extension.

The second example we propose aims to show how other applications can query a sink node when searching for devices suitable for specific purposes. The reference scenario refers to an application aiming to control the HVAC system of Room C. Even very small variations of temperature and humidity should be detected in order to improve user comfort: hence devices with finer operating specifications will be required now as evident in the request reported in what follows:

$(\mathbf{R_3})\mathbf{AirConditioning\_Request} \equiv TemperatureSensor \sqcap \exists\ hasMeasurementCapability \sqcap \forall\ hasMeasurementCapability.(\exists\ hasMeasurementProperty \sqcap \forall\ hasMeasurementProperty.(LowMeasurementRange \sqcap MediumFrequency \sqcap HighAccuracy \sqcap HighResolution)) \sqcap \exists\ hasSubSystem \sqcap \forall\ hasSubSystem.(Battery \sqcap \forall\ hasSurvivalProperty.(HighBatteryLifetime))$.

Table 4 reports on the top five results for R_3. Unlike the first scenario, sensor S_3 has the highest score thanks to the lowest semantic distance. Sensor S_1 is also a good candidate, with a similar overall score: it is closer to the reference location P but it has a slightly higher semantic rank than S_3.

5 Conclusion

The paper proposed a novel SSN framework, supporting logic-based matchmaking of meaningful and semantically rich event/device/resource annotations via

simple and backward-compatible CoAP extensions. Peculiarities of the proposed solution have been outlined w.r.t. a case study showing its benefits.

Future work includes a thorough experimental campaign on a large testbed, in order to accurately evaluate performance issues, the integration of novel non-standard inferences (such as Concept Contraction and Concept Covering), to make semantic matchmaking even more detailed as well as the extension of underlying logic toward $\mathcal{EL}/\mathcal{EL}^{++}$ for increasing allowed expressiveness of resource and request descriptions.

Acknowledgments

The authors acknowledge partial support of EU ECTP 2012-14 project "GAIA: Generalized Automatic exchange of port Information Area".

References

1. Ni, L., Zhu, Y., Ma, J., Li, M., Luo, Q., Liu, Y., Cheung, S., Yang, Q. In: Semantic Sensor Net: An Extensible Framework. Volume 3619 of Lecture Notes in Computer Science. Springer (2005) 1144–1153
2. Avancha, S., Patel, C., Joshi, A.: Ontology-driven adaptive sensor networks. In: First Annual International Conference on Mobile and Ubiquitous Systems, Networking and Services. (2004) 194–202
3. Ke, W., Ayyash, S., Little, T.: Semantic internetworking of sensor systems. In: Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference on, IEEE (2004) 484–492
4. Compton, M., Neuhaus, H., Taylor, K., Tran, K.: Reasoning about sensors and compositions. Proc. Semantic Sensor Networks (2009) 33
5. Jiang, G., Chung, W., Cybenko, G.: Semantic agent technologies for tactical sensor networks. In: 2003 SPIE Conference on AeroSense. (2003) 21–25
6. Sheth, A., Henson, C., Sahoo, S.: Semantic Sensor Web. Internet Computing, IEEE **12**(4) (2008) 78–83
7. Guinard, D., Trifa, V.: Towards the Web of Things: Web Mashups for Embedded Devices. In: Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain. (2009)
8. Compton, M., Barnaghi, P., Bermudez, L., Garcia-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., et al.: The SSN ontology of the W3C semantic sensor network incubator group. Web Semantics: Science, Services and Agents on the World Wide Web (2012) in print.
9. Colucci, S., Di Noia, T., Pinto, A., Ragone, A., Ruta, M., Tinelli, E.: A Non-Monotonic Approach to Semantic Matchmaking and Request Refinement in E-Marketplaces. International Journal of Electronic Commerce **12**(2) (2007) 127–154
10. Russomanno, D.J., Kothari, C.R., Thomas, O.A.: Building a Sensor Ontology: A Practical Approach Leveraging ISO and OGC Models. In: The 2005 International Conference on Artificial Intelligence. (2005) 637–643
11. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. International Journal on Semantic Web and Information Systems (IJSWIS) **5**(3) (2009) 1–22

12. Patni, H., Henson, C., Sheth, A.: Linked Sensor Data. In: Collaborative Technologies and Systems (CTS), 2010 International Symposium on, IEEE (2010) 362–370
13. Page, K., De Roure, D., Martinez, K., Sadler, J., Kit, O.: Linked Sensor Data: RESTfully serving RDF and GML. Proceedings of the 2nd International Workshop on Semantic Sensor Networks (2009) 49–63
14. Bormann, C., Castellani, A., Shelby, Z.: Coap: An application protocol for billions of tiny internet nodes. Internet Computing, IEEE **16**(2) (2012) 62–67
15. Barnaghi, P., Presser, M., Moessner, K.: Publishing Linked Sensor Data. In: Proceedings of the 3rd International Workshop on Semantic Sensor Networks. (2010)
16. Pfisterer, D., Romer, K., Bimschas, D., Kleine, O., Mietz, R., Truong, C., Hasemann, H., Pagel, M., Hauswirth, M., Karnstedt, M., et al.: SPITFIRE: Toward a Semantic Web of Things. Communications Magazine, IEEE **49**(11) (2011) 40–48
17. Taylor, K., Leidinger, L.: Ontology-driven complex event processing in heterogeneous sensor networks. The Semanic Web: Research and Applications (2011) 285–299
18. Ruta, M., Di Noia, T., Di Sciascio, E., Donini, F.: Semantic-Enhanced Bluetooth Discovery Protocol for M-Commerce Applications. International Journal of Web and Grid Services **2**(4) (2006) 424–452
19. Ruta, M., Scioscia, F., Di Sciascio, E.: Mobile Semantic-based Matchmaking: a fuzzy DL approach. The Semantic Web: Research and Applications (2010) 16–30
20. Ruta, M., Scioscia, F., Di Noia, T., Di Sciascio, E.: A hybrid ZigBee/Bluetooth approach to mobile semantic grids. Computer Systems Science and Engineering **25**(3) (May 2010) 235–249
21. De Virgilio, R., Di Sciascio, E., Ruta, M., Scioscia, F., Torlone, R.: Semantic-based RFID Data Management. In: Unique Radio Innovation for the 21st Century: Building Scalable and Global RFID Networks. Springer (2011) 111–142
22. Mahoney, M.: Adaptive Weighing of Context Models for Lossless Data Compression. Technical report, Florida Tech. Technical Report CS-2005-16, 2005
23. Liefke, H., Suciu, D.: Xmill: an efficient compressor for xml data. SIGMOD Rec. **29**(2) (2000) 153–164
24. Scioscia, F., Ruta, M.: Building a Semantic Web of Things: issues and perspectives in information compression. In: Semantic Web Information Management (SWIM'09). In Proceedings of the 3rd IEEE International Conference on Semantic Computing (ICSC 2009), IEEE Computer Society (2009) 589–594
25. Baader, F., Calvanese, D., Mc Guinness, D., Nardi, D., Patel-Schneider, P.: The Description Logic Handbook. Cambridge University Press (2002)
26. Ruta, M., Di Sciascio, E., Scioscia, F.: Concept Abduction and Contraction in Semantic-based P2P Environments. Web Intelligence and Agent Systems **9**(3) (2011) 179–207
27. Ruta, M., Scioscia, F., Di Sciascio, E., Gramegna, F., Loseto, G.: Mini-ME: the Mini Matchmaking Engine. In Horrocks, I., Yatskevich, M., Jimenez-Ruiz, E., eds.: OWL Reasoner Evaluation Workshop (ORE 2012). Volume 858 of CEUR Workshop Proceedings., CEUR-WS (2012) 52–63
28. Kovatsch, M., Mayer, S., Ostermaier, B.: Moving Application Logic from the Firmware to the Cloud: Towards the Thin Server Architecture for the Internet of Things. In: Proceedings of the 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS 2012). (July 2012)
29. Kovatsch, M.: Demo Abstract: Human-CoAP Interaction with Copper. In: Proceedings of the 7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2011). (June 2011)