

Pub/Sub and Semantic Annotation Enablers for Future Internet in the PPP EU Project Test-Bed

Boris Moltchanov¹, Oscar Rodríguez Rocha²

¹Telecom Italia, via G. Reiss Romoli, 274, 10148-Turin, Italy
boris.moltchanov@telecomitalia.it

²Politecnico di Torino, Corso Duca degli Abruzzi, 24 - 10129 Turin, Italy
oscar.rodriguezrocha@polito.it

Abstract

Internet has significantly grown during the last decade; also many novel services appeared, generating huge incomes and fame worldwide. This is the era of new Internet services over the top, without requiring specific changes or specific functionalities from the “old” version of Internet. This, on one side, does not pose any limitations for the creation of new services able to build and offer on top of the large Internet technological foundation. On the other side, this way of “do it (everything) by yourself” has its drawbacks, such as slow creation process, long time-to-market, deep knowledge of many technologies on top of Internet, no standardization, no interoperability, solutions created by different people, and many other typical limitations of free but complex individual development. All those aforementioned classic Internet characteristics lead to a strong fragmentation of technology and services. The Future Internet initiative instantiation by European Commission in form of Private-Public Partnership (PPP) Program has the objective to populate Internet with common conceptual components (Generic Enablers or GE) enabling faster and interoperable service creation. FI-WARE is the technological core foundation project of the PPP program that provides specifications and reference implementations of the most common generic enablers and deploys them on its test-bed for public and evaluation usage. The Publish/Subscribe and Semantic Annotation GEs that the project has provided and installed in the FI-WARE test-bed are demonstrated in this paper.

Introduction

The work presented in this paper, describes the process of creation and deployment of two GEs in the PPP FI-WARE test-bed serving for evaluation and adaptation of two new conceptually new components (enablers) within the Future Internet program funded by EU commission under PPP pro-

gram. The first initial technology and thoughts are introduced in the Section 2, describing when selection process of the components started and how it has been chosen with some technical particularities at a very high level. On Section 3, a detailed description of the Public/Subscribe GE is presented while on Section 4 a brief overview of the Semantic Annotation GE is given. The test-bed of this enabler is described on Section 5.

The Conclusions and the roadmap of these two GEs are given together with the future work within the EU funded initiative (Section 6), underlying the relevance of this work for a modern future internet and information enabled Society.

II. PPP FI-WARE Start-up

In 2010, the EU Commission founded the Future Internet Public Private Partner initiative¹². It resulted into a number of Use Case Projects (UCPs) and a Future Internet Core Platform (FI-WARE project³) embracing all the Generic Enablers (GE) commonly used by any UCP. One of the most required GEs identified within FI-WARE, is the Publish/Subscribe GE. We have chosen the OMA’s NGSI open standard⁴ after a careful analysis of the existing industrial open standards and also by taking into consideration the already existing solutions provided by the FI-WARE partners. During the selection process, the priority has been given to the practically implemented, existing and used solutions, which are rather simple and able to work with heterogeneous devices in different application domains. OMA’s

¹Thanks to EU PPP Initiative for funding.

² <http://www.internet-of-things.eu>

³ <http://www.fi-ware.eu>

⁴ See the Open Mobile Alliance (OMA) Next Generation Services Interface (NGSI) Specification http://www.openmobilealliance.org/Technical/release_program/docs/NGSI/V1_0-20101207-C/OMA-TS-NGSI_Context_Management-V1_0-20100803-C.pdf

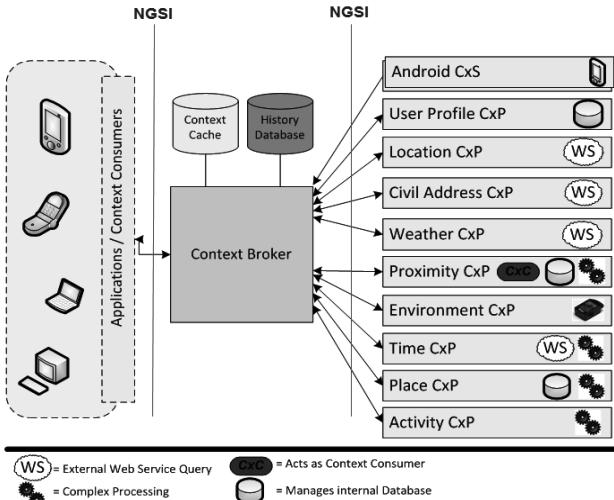


Figure 1. OMA's NGSI data representation model

NGSI open standard allows to retrieve any type of information, including context data and events (represented as schematically shown in the Figure 1), from their respective providers in different modes: on-request and subscription-based.

Another prominent GE allowing “plain” text enrichment with semantically meaningful data is the Semantic Annotation. It has been noted that many modern systems (search, meta-data tagging, etc.) are not leveraging on the semantic data related to the their data (search results, tags and so on respectively) therefore a system allowing to gather and link the semantic data, as meta-data, to the main stream data is needed in Future Internet. This component was introduced into overall Data/Context chapter architecture as Semantic Annotation GE.

The Publish/Subscribe GE

Overview

This GE, enables the publication of context information by entities, referred as Context Producers, so that published context information becomes available to other entities, referred as Context Consumers, which are interested in processing the published context information. Applications or even other GEs in the FI-WARE platform may play the role of Context Producers, Context Consumers or both. Events in FI-WARE-based systems refer to something that has happened, or is contemplated as having happened. As such, they provide context information that can be handled by applications or FI-WARE GEs.

The Context Broker GE supports two-ways communication: push and pull towards both the Context Producer and the Context Consumer. It does mean that a Context Producer with a minimal or very simple logic may continuously push

the context information into the Context Broker, when the information is available or due to the internal logic of the Context Producer, and Context Broker on its side can request the context information to more complex (server) Context Producers, when it is needed to the Context Broker. In a similar way also the Context Consumers can pull the context information from the Context Broker retrieved the information in on-request mode, while the Context Broker can push the information to a Context Consumer interested for that in a subscribed mode.

A fundamental principle of the Context Broker GE is to achieve a total decoupling between Context Producers and Context Consumers. This means that Context Producers publish data without knowing which Context Consumers will consume published data; therefore they do not need to be connected to them. On the other hand, Context Consumers consume data of their interest, without this meaning they know which Context Producer has published a particular event: they are just interested in the event itself but not in who generated it. As a result, this GE it is an excellent bridge enabling external applications to manage events related to the Internet of the Things in a simpler way hiding the complexity of gathering measures from IoT⁵ resources (sensors) that might be distributed or involving multiple low-level communication protocols.

Context elements

In compliance with the OMA NGSI specification,⁶ the Context Information in FI-WARE is represented through generic data structures referred as Context Elements. A Context Element refers to information that is produced, generated, collected or observed that may be relevant for processing, carrying out further analysis and knowledge extraction. It has associated a value defined as consisting of a sequence of one or more <name, type, value> triples referred as data element attributes. FI-WARE will support a set of built-in basic data types as well as the possibility to define structured data types similarly to how they are defined in most programming languages.

A Context Element typically provides relevant information of a particular entity, being a physical thing or part of an application. Finally, there is meta-data (also referred as semantic data) linked to attributes in a data element. However, its existence is optional.

Context elements are associated with (Figure 2):

- An *EntityId* and *EntityType*, uniquely identifying the entity to which context data refers.
- A sequence of one or more data element attributes (<name, type, value> triples)
- Optional meta-data linked to attributes (also <name, type, value> triples)

⁵ <http://www.internet-of-things.eu>

⁶ <http://www.fi-ware.eu>

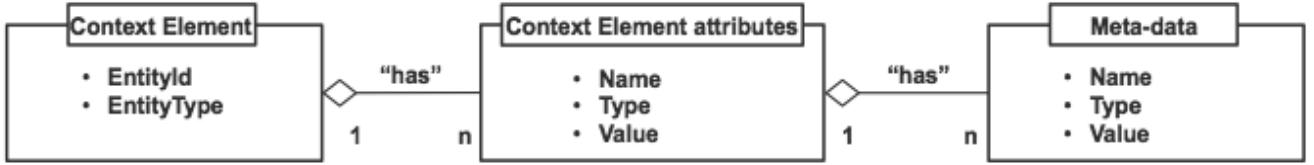


Figure 2. Context Elements

A cornerstone concept in FI-WARE is that Context Elements are not bound to any specific representation formalism. As an example, they can be represented as:

- An XML document (SensorML, ContextML, or whatever)
- A binary buffer being transferred
- As an entry in RDBMS table (or a number of entries in different tables),
- As a number of entries in a RDF Repository
- As entries in a noSQLdatabase.

A key advantage of this conceptual model for context elements is its compliance with IoT formats (SensorML) while enabling further extensions to them.

GE Architecture

Context Broker

It's the main component of the architecture. It works as a handler and aggregator of context data and as an interface between the components in the architecture. Primarily, the CB has to control the context flow among all attached components; in order to do that, the CB must to know every Context Provider in the architecture; this feature is done through an announcement process.

Context Provider

Provides context information in synchronous mode; it means that a Context Consumer or even the Context Broker can invoke the CP in order to acquire context information. A CP provides context data only further to a specific invocation; it never sends data to another platform component in asynchronous mode. Moreover, a CP can produce new context information inferred from the computation of input parameters; hence it is responsible for reasoning of high-level context and for sensor data fusion. Every CP registers its availability and capabilities by sending appropriate announcements to the CB and exposes interfaces to provide context information to the CB and to Context Consumers.

Context Source

A *Context Source* (CS) updates context information, about one or more context scopes, in asynchronous mode. A CS sends context information according to its internal logic and never due to an invocation from another middleware component and does not expose the same interfaces of CP to the CB and to Context Consumers.

Compared to the pull based CP-CB communication, the communication between CS and CB is in push mode.

Context Consumer

A *Context Consumer* (CC) is an entity that uses context data. A CC can retrieve context information sending a request to the CB or invoking directly a CP over a specific interface. Another way for the CC to obtain information is using implicit request to the CB, it means that the CC subscribes to a specific context update event and the CB notifies it when events occur.

Entity

Every exchange of context data here is referred to a specific entity, which can be in its order a complex group of more than one entity. An entity is the subject (e.g. user or group of users), which context data refer to, and it is composed of two parts: a type and an identifier. Every Context Provider supports one or more entity type and this information are published to the Context Broker during an announcement process described later.

A type is an object that categorizes a set of entities; for example entity types are:

- *username* – for human users;
- *imei* – for mobile devices;
- *mobile* – GSM phone number for mobile users;
- *SIP uri* – for SIP accounts;
- *groupid* – for groups of other entities;

The entity identifier specifies a particular item in a set of entities belonging to the same type. Every human user of the context management platform could be related to many entities in addition to the obvious type *username*, it means that a process that provides identity resolution is necessary. Considering for example a CP that provides geographical cell based location for mobile devices; if the location information is obtained from the computation of parameters provided by mobile devices, this CP supports entity type *mobile*. When the CB receives a *getContext* request about location of a human user, therefore with entity type *username*, the CB could not invoke the provider previously described because it does not support this entity type, but if the user has a mobile device; information about his location is probably available in the system. If the CB could retrieve all entities related to this user, it could invoke the provider using, if it is possible, right entities. This feature could be provided using a detailed data-

base collecting all information about users; it means that the CB could refer to this DB in order to retrieve all entities connected to a specific user. In this way the example described previously could work because, when the CB receives the request, it invokes the database and retrieves the entity of type mobile related to the user; afterwards, the CB could invoke the location provider using the obtained entity and could send response with location data to the requester.

Context scopes

Any context information set within the Context Management Framework is defined as a “scope”, which is a set of closely related context parameters. Every context parameter has a name and belongs to only one scope. Using *scope* as context exchange unit is very useful because parameters in that scope are always requested, updated, provided and stored at the same time; it means that creation and update data within a scope are always atomic and that context data in a scope are always consistent. Scopes themselves can be atomic or aggregated, as union of different atomic context scopes.

Main features and functionalities

Context caching

Any context information (scope) received by the Context Broker (from a Context Source or as a result of a request to a Context Provider) is stored in a context cache. If another Context Consumer requests the same scope to the Context Broker, it can be retrieved from the cache, if it is not expired, without need to invoke the same Context Provider again and therefore speeding up the process of context delivery.

Context validity

Every scope that is exchanged is tagged with its timestamp and expiry time. The expiry time tag states the validity of the scope. After this time, the information is considered not to be valid any more, and should be deleted. The setting of the expiration time is in charge of the Context Source or Context Provider that generates the context information and the Context Broker can only change it to synchronize to its clock.

When the Context Broker is asked for a scope, it first looks for it in its cache. If the information is found, the expiry time is checked. If the expiration time is reached, the Context Broker removes it from the context cache and requests it from respective Context Provider again.

Context history

Every context scope exchanged between the Context Broker and Context Providers or Context Sources is logged in the context history. Differently is for the context cache, which stores only currently valid information. The context history makes the past context information of an entity available, without reference to current validity. Context reasoning techniques could be applied to the context history in order to correlate contexts and deduce further context information, e.g. about situations, user intentions (sub-goals) and goals.

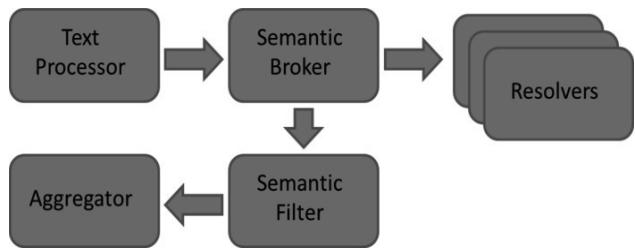


Figure 3. Semantic Annotation GE Architecture

Semantic Annotation GE

The Semantic Annotation GE aims at performing named entity recognition and semantic annotation from a given text. This enabler is based on the Automatic Semantic Annotator described in (Rocha et al. 2012).

The Semantic Annotation GE is shown in Figure 3. In order to accomplish the semantic annotation task, it is composed by a *Text Processor* module, which processes the given text, and identifies the source language. Then, a morphological analysis is performed using FreeLing⁷ configured with the identified language. From this analysis, proper nouns lemmas are extracted while other part-of-speech is discarded. At this time, non-numeric proper nouns lemmas with a score of at least 0.2 are preserved and merged with plain tags to compute a well-defined list of unique (multi) words. At this stage, the module uses term frequency to further process the title and to extract other potential relevant words.

The next step involves the *Semantic Brokering* module assisted by a set of resolvers that perform full-text or term-based analysis based on the previous output. Such resolvers are aimed at providing candidate semantic concepts referring to Linked Data as well as additional related information if available. Resolvers may be domain or language-specific, or general purpose. For term-based analysis, each word of the previously computed list is individually processed to identify a list of candidate Linked Data resources to match with. A set of predefined services, such as DBpedia⁸, Sindice⁹ and Evri¹⁰ are invoked in parallel.

The *Semantic Filtering* module processes candidate Linked Data resources received by the broker and performs a disambiguation based on the DBPedia score and the string similarity between each surface form and its corresponding list of candidates, based on the Jaro Winkler distance. This function aims at maximizing both values to identify the “preferred” candidate. In this process, after several empirical

⁷ <http://nlp.lsi.upc.edu/freeling/>

⁸ <http://www.dbpedia.org>

⁹ <http://www.sindice.com>

¹⁰ <http://www.evri.com>

tests, candidates with distance lower than 0.8 are discarded at this stage, unless their DBpedia score is the maximum. Automatic annotation is performed using the “preferred” candidate identified during this step.

Project Test-Bed

To ensure the proper functioning of the enablers presented above (and some other that make up the set of enablers of the project), and constantly monitor their performance, a test-bed has been created in the Project. The test-bed contains a total of 8 instances of the Pub/Sub GE (each one provides both the NGSI and the ContextML/CQL interfaces. A dedicated instance for the reference implementation of this enabler is also available with the two interfaces.¹¹¹² These two instances are mainly serving to internal FI-WARE testing and integration purposes, while the UCPs’ instances are customized and serving exclusively to their respective purposes. The test-bed is set and running in a high-available fault-tolerant configuration therefore is able to serve high intensive traffic in a pre-production environment. In the same test-bed served by many servers FI-WARE has installed a lot of different GEs, among the two described in this paper, therefore more complex and combined services, logics and configuration are easily possible being provided many GEs with the same FI-WARE NGSI standard interface. This makes the test-bed much more valuable and useful.

Conclusions and Future Work

The Publish/Subscribe and Semantic Annotation GEs are installed and running in the test-bed and 7 Use Case Projects already have their own dedicated instances and used them for their internal purposes, giving to us a very valuable feedback about the functionality and usage of the components. The PPP Phase II projects are already starting to take off and those also will use the test-bed with the GEs created in FI-WARE. FI-WARE provided not only unique one instance of the Publish/Subscribe GE but couple of them, one created by Telecom Italia (originally CAP platform) and another one by Telefónica (as a part of the SAMSON platform).

The roadmap of the GE is shown in the wiki web-site¹³; and the features, which will be supported by the GEs are described in the wiki pages.¹⁴

All these functionalities and features are taken from the tracker of the FI-WARE project, which collects the require-

ments and request coming from the UCPs and other 3rd parties and then after approval of the FI-WARE Architectural Board (AB) included into the development roadmap.

The future work we still have to accomplish is mainly focused on the following main topics:

- Completely fulfill the features and functionality of the Publish/Subscribe NGSI interface;
- Integrate the semantic data handling with non-semantic current version of the Publish/Subscribe GE;
- Develop and integrate the mechanisms allowing to the Semantic Annotation GE to attach new databases for the semantic data, handling different application semantic domains with different logic or reasoning, which shall also be attachable and configurable and give explicit indications about sources of the information used for the result to the requesting part.

All these features will be integrated during this last year of the project execution and reflected in the GE roadmap referenced above.

Reference

Rocha, Oscar Rodríguez; Criminisi, Carmen; Mondin, Fabio; and Goix, Laurent-Walter. 2012. LODifying personal content sharing. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops (EDBT-ICDT '12)*

¹¹ See Pub/Sub GE Reference implementation with the NGSI Interface. <http://pubsub.lab.fi-ware.eu/ngsicbapi/>

¹² See Pub/Sub GE Reference implementation with the ContextML/CQL Interface. <http://pubsub.lab.fi-ware.eu/CB>

¹³ See the FI-WARE Project Wiki, http://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php Releases_and_Sprints_numbering,_with_mapping_to_calendar_dates

¹⁴ See the FI-WARE Project GE Wiki, http://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php Roadmap_of_Data/Context_Management