A Public-Private Partnerships Model Based on OneM2M and OSGi Enabling Smart City Solutions and Innovative Ageing Services

Paolo Lillo(✉), Luca Mainetti, and Luigi Patrono

Department of Innovation Engineering, University of Salento, Lecce, Italy
{paolo.lillo,luca.mainetti,luigi.patrono}@unisalento.it

Abstract. The smart cities promise to offer innovative services to citizens in order to improve the level of life quality but currently, the integration among different ecosystems of data are still lacking. It is noteworthy that data, produced and consumed by public and private institutions, and citizens, may be a precious resource if abstraction and effectiveness are guaranteed in the interface mechanism between data-producers and data-consumers. This work proposes a coupling model, based on standards, technologies, and methodologies, able to make easy and effective the distribution, access, and use of data between services providers and citizens. The system architecture has been designed and developed by using the OSGi technology exploiting a support supplied by Docker, a platform able to ensure greater freedom in the modularization of software platforms oriented to micro-services. Furthermore, for higher levels, an approach based on the OneM2M standard has been adopted in order to obtain a middle layer useful to large-scale coordination of aspects related to the gathering, discovery, security and distribution of data and services. An use case has been defined and summarized in order to clearly show potential benefits of the proposed system for all stakeholders.

Keywords: Smart cities · oneM2M · OSGi · Docker · Micro-services

1 Introduction

A very challenging issue related to smart cities is related to the integration of heterogeneous data and services in order to guarantee an efficient and effective support in the daily activities of citizen, especially for the elderly people.

Currently, who provides a service for a smart-city often makes available more or less 'raw' and 'open' data, stored on more or less 'raw' supports (e.g., CSV files or relational databases). Furthermore, the access to such data by the consumer requires the knowledge of logical and semantic aspects, the implementation of articulated polling mechanisms in order to detect data updates, the implementation of filtering, aggregation and reasoning processes of accessed data due to application needs. Other times, the service is provided in a fully enclosed manner, through the development of the complete cycle of collection, formatting, data storage and distribution through a proprietary front-end. On the contrary, a 'harmonious' development of services to citizens would certainly

A. Longo et al. (Eds.): IISSC 2017/CN4IoT 2017, LNICST 189, pp. 49–57, 2018.
https://doi.org/10.1007/978-3-319-67636-4_6

be facilitated by the availability of open, semantic data, accessible by third parties on-demand through the adoption of a modularized system supporting the full decoupling of the components related to the production and the consumption of data.

In such a context, some public and private operators would provide a service of gathering and delivery of raw/semantic data, assigning to the system the task of managing issues about marshalling, reasoning and delivery of data to other public and private operators, in the role of consumers, on their preferred channels; adaptation and supplying operate symmetrically with respect to the core system by providing respectively data abstraction and data concretization.

The interaction between systems and external agents must be designed by minimizing the coupling between components (manufacturer, system, consumer) and respecting the principles of the "reactive manifesto" [1].

The efforts of the oneM2M Global Initiative [2, 3], an international project involving eight leading standard ICT bodies (ARIB, ATIS, CCSA, ETSI, TIA, TSDSI, TTA, TTC) are oriented in the direction of the development of specifications able to simplify and harmonize the integration of systems on issues related to the IoT and the M2M communication. This happens in a highly fragmented context, full of methodologies based on a plethora of protocols and proprietary solutions that make very difficult the discovery and the access to data and common services by applications. A lot of works in the scientific literature are converging on the oneM2M specifications, strengthening infrastructural [4, 5] or semantic [6] aspects, but to the best of our knowledge, issues related to the decoupled and reactive interaction between applications and system have not been resolved yet.

The goal of this work is to stimulate the development of applications and services by public and private 'third parties', making easier the access, in 'asynchronous' and 'reactive' mode, involving various channels (e.g., REST, streams, brokers, etc.), not only to the raw data but exploiting abstractions based on semantics and reasoning.

This paper proposes a modularized architectural model composed of the following levels: (i) platform (Docker), (ii) middleware orchestration (OneM2M), and (iii) micro-services (OSGi) [7, 8]. Furthermore, the proposed model is therefore applied in a project aimed at behavioral monitoring of elderly by means of data gathered using IoT technologies whose name is City4Age [9].

The rest of the paper is organized as follows. Section 2 describes the proposed system architecture. Details about middleware orchestration by adopting OneM2M standard are reported in Sect. 3, while an micro-services implementation of the proposed system, based on OSGi, is described in Sect. 4. Section 5 reports a short discussion of a use case the uses the proposed system. Concluding remarks are drawn in Sect. 6.

2 Proposed System Architecture

The proposed architecture aims at decoupling producer and consumer of data through the introduction of the concept of Data Producer/Consumer Channels (respectively DP Channels and DC Channels) among which the system acts as a "smart coupler". Figure 1 shows the Level 1 DFD (Data Flow Diagram) with a high level of abstraction,

highlighting the components participating in the process of production, gathering, parsing, processing, adaptation, delivery and consumption of information.

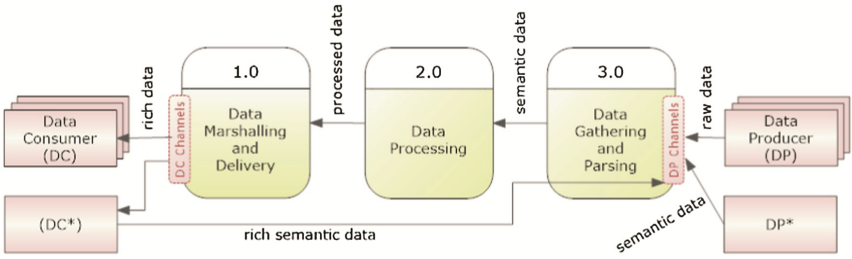

Fig. 1. Level 1 DFD: overall architecture

The specific DP Channel supplied by the producer may be of different type from that required by the consumer.

Producers and consumers are connected to the system through multi-channel adapters able to support the various levels of intelligence, technology and performance of related agents. For example, a DP (DP* in Fig. 1) can provide a stream of semantic data, ready to feed the knowledge base (embedded in the process 2.0 in Fig. 1, named "Data Processing") with a minimum adaptation effort. Another provider can provide access to its own data in "pull" mode by equipping the system with credentials in order to access its DBMS (Data Base Management System) via a public URL. Another can provide raw data by uploading a CSV file in "push" mode using a REST API supplied by the system or alternatively in "pull" mode, providing a URI of the CSV file which updates can be periodically monitored by the system.

On the other hand, the Data Consumer (DC), aiming at offering services to citizens (but also - as in the case of the consumer DC* in Fig. 1 - the implementation of further reasoning processes or enrichment of the information), has the freedom to access the data by choosing the preferred mode via DC Channels related to specific RESTful API, push notifications, stream, brokering protocols (e.g., MQTT, AMQP, XMPP).

The proposed architecture has the objective to 'normalize' the access mechanism to the system, providing external agents with standardized, modern and responsive ways to gather and provide data, then giving responsibility to the system about most of activities related to issues about coupling, processing and distribution.

2.1 Data Processing and Challenges

The proposed system implements a mechanism of data adaptation from the producer to the consumer foreseeing, at the same time, a model able to guarantee data processing through the use of semantic and linguistic tools.

At a higher level of detail, Fig. 2 shows in a Level 2 DFD a possible development of the Data Processing, steering it towards a highly declarative approach that supports high levels of system automation.

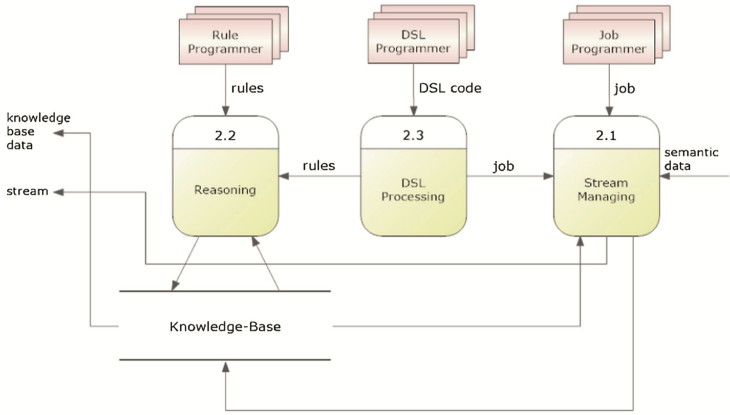

Fig. 2. Level 2 DFD: Details of the Data Processing block

In practice there are well-established solutions for the aspects of the representation and the semantic reasoning: the implementation of the Reasoning block (Process 2.2) can therefore be reasonably reduced to an activity of integration, in the proposed system, of well-known OWL (Ontology Web Language) technologies and open-sources java tools such as OWL API [10, 11]; the conclusions inferred from reasoning will be available to business logic in the form of events.

Conversely, the implementation of the DSL Processing block (Process 2.3) proposes an interesting challenge on the ability to use 'ad hoc' grammars (and related compilers/interpreters) in order to support, declaratively and as close as possible to the varius application domains, the interaction between agents and system about issues of (i) negotiation of the requested/offered services and (ii) submission of 'jobs' to be executed in the system. A very interesting approach to the question is provided by the Xtext framework [12].

Note that the process 2.3 does not add paradigmatic elements (rules and job) but incorporates them by introducing a further abstraction.

3 Middleware Orchestration

The interconnection and exchange of data and services among heterogeneous and distributed systems are inevitable and it would behoove to use the standards in a manner as shared as possible, exposing the functionalities to the agents in the most accessible way.

OneM2M achieves its targets of decoupling and integration by means of a model based on the concept of "resource" and related CRUD + N functionalities (Create, Retrieve, Update, Delete and Notify) accessible both through blocking-requests as through synchronous/asynchronous non-blocking-requests.

The oneM2M specification defines a network topology based on the following node types: Infrastructure Node (IN), Middle Node (MN), Application Service Node (ASN), Application Dedicated Node (ADN), and Non-oneM2M Node (NoDN).

Inside a network of a specific organization, services are exposed by one or more Common Service Entities (CSE) to the Application entities via "Mca Reference points"; the interaction between services deployed on distinct internal nodes (Asn, MN) is managed by the "Mcc Reference points" while, the Infrastructure Nodes, by means of Mcc' Reference Points, mediate the interaction between distinct Service Providers, enabling the distribution of services shared on a large-scale.

The communication mechanism is abstracted and conveyed using a protocol layer below implemented via HTTP (Hyper Text Transfer Protocol)/COAP (Constrained Application Protocol)/MQTT (MQ Telemetry Transport).

This paper introduces inside the architecture of OneM2M, through its own paradigm, some entities, resources and CSE (Common Service Entity) aimed to a flexible and responsive coupling between DP and DC, assisting the developers in implementing reactive programming models.

Figure 3 shows as new Common Service Functions (CSF) are included within the CSE (Common Service Entity) related to the system architecture proposed in this paper.

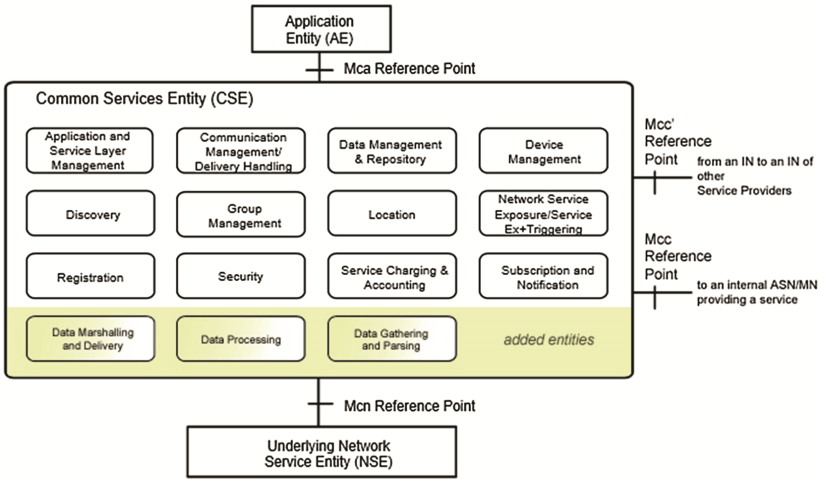

Fig. 3. New common service functions

According to its own type, each node maybe contain specific entity types among those represented in Fig. 3. The proposed architecture fully is able to satisfy the requirements of modularity and scalability enabling integration of different structures (Providers) through IN nodes that provide the interaction among the infrastructural services via "Reference Points" of type Mcc'.

The interaction between the new CSF and their interaction with the outside world is mediated by the exchange of a new set of oneM2M Resources [3], introduced in order to support, in a oneM2M context, the functional paradigm introduced by the system proposed in this work. Only some of the most significant Resources are reported:

- DPChannel: description of the data-channel proposed to the system.
- DPChannelRequest: request to create a referenced DPChannel.

- DPChannelResponse: outcome of a DPChannelRequest and detail of the access parameters to the related DPChannel.
- DCChannel: description of the data-channel requested to the system.
- DCChannelRequest: request to access a referenced DCChannel.
- DCChannelResponse: outcome of a DCChannelRequest and detail of the access parameters to the related DCChannel.

4 OSGi Micro-Services Implementation

The goal of decoupling is to be achieved not only in terms of the implementation aspects but also in terms of the independence of design/development teams, leaving them free to design, implement, test and install system components and services, eliminating or minimizing the interference with other teams and operators but always respecting a shared application model.

With those expectations, a PaaS (Platform as a Service) model based on Docker [11], well known technology based on the concept of 'container' as a way for isolation of application components and for the isolation of the components from underlying infrastructure, has been adopted.

The 'containerization' system based on Docker perfectly matches with a modularization technology known as OSGi (Open Service Gateway initiative) [7].

The OSGi specifications make it possible (i) the modularization of packaging, release and deployment of software components (aka bundles), and (ii) a 'dynamic' modularization at run-time based on local and remote sharing of micro-services.

By using an integrated development environment (IDE) such as Eclipse, enhanced with BndTools [14] plugin for support of OSGi specifications, the design/development team can build its continuous-delivery environment by splitting itself into subgroups, individually dedicated to the implementation of several modular components (OSGi bundles).

The development process concludes each iteration with an automatic process that, starting with the generation of a single JAR file (which contains: OSGi implementation, microservices, configurations and resources), continues producing the Docker 'image' (the container) of the entire application component (including all dependencies required for its execution) and finally ends with the deployment of the container in a Docker environment.

Once installed, the container is fully integrated with the system by registering its offered services and by 'hooking' its requested services through the local registration service and the discovery mechanisms provided by implementations of the Remote Services specifications defined in the enterprise section of OSGi specifications [8].

As reported in [15], a rather natural approach to the interworking between oneM2M and OSGi is the implementation of the oneM2M Resources by means of OSGi services by exposing a hierarchy of Java interfaces with root-interface as reported, below, mapping its methods on the attributes [15], in the Table 9.6.1.3.1-1, *resourceType*, *resourceID*, *resourceName*, *parentID*, *creationTime*, *lastModifiedTime*, *labels*, universal to all resource types:

```
public interface oneM2MResource {
    public int getResourceType();
    public String getrResourceID();
    public void setResourceName(String resourceName);
    public String getParentID();
    public Date getCreateionTime();
    public Date getLastModifiedTime();
    public String[] getLabels();
}
```

5 Use Case Discussion: City4Age Project

With the aim to validate the system proposed in this work we applied the model to the intervention mechanism of the City4Age project [16] in the context of the pilot in the Lecce Municipality.

"City4Age - Elderly-friendly city services for active and healthy ageing", a project co-funded by the Horizon 2020 Programme of the European Commission, supports smart cities to empower social/health services facing MCI and frailty of the elderly population. In particular, the City4Age project aims to utilize digital technologies at home and in the city to monitor behaviors of elderly people with the goal of properly intervene on them in order to delay the onset and progression of MCI and frailty.

As a result of their behavioral patterns analysis, participants will receive interventions aimed at improving behaviors known to affect risk of onset or worsening of MCI and frailty. There are two types of interventions: (i) informative intervention characterized by general information related to the same city, and (ii) specific intervention related to personal risks.

Furthermore, the Municipality of Lecce offers to citizenship, as open-data [17], information accessible mainly through CSV text files. Emulating the work of a municipal employee responsible for the publication of information on cultural events offered by the same city, we provided a structured CSV format for that purpose.

The proposed system, instantiated for City4Age case study, has been set to prepare a DPChannel resource ('resource' in terms of oneM2M paradigm) sending a description of the provided data-source (DPChannel resource) through: (i) an URI, pointing the CSV file, (ii) a semantic description of fields (the comma-separated values) and (iii) a semantic description of the channel.

The system responded by creating a DPChannel resource (with a referencing ID in the DPChannel response) and automatically set up a polling mechanism with access to the CSV file every 30 s (default) and consequent updating of a data base.

Acting in the role of operators for City4Age, we therefore asked the system to create a second DPChannel provided in the form of a REST API to be invoked for the push (from the outside) of a set of data related to recognition of a behavior of an older person (e.g. excessive physical inactivity); also in this case we have accompanied the request with the following description: (i) a semantic description of the input parameters and (ii) a semantic description of the channel.

Then the system has been ordered to create two DCChannel resources: the first to receive the continuous stream of behavioral data and the second to perform, via REST API, filtered requests relating to scheduled cultural events.

Finally, we have implemented and ran an application (outside but connected to the system) that collects data from the stream (containing, among others, the elderly and the data type of the detected behavior), via REST queries the system on appropriate events to chance and sends the elder's phone a PUSH notification via Amazon SNS containing the suggested advice.

The adopted validation approach has shown the effectiveness of the proposed architecture allowing external agents to seamlessly integrate with the system maintaining a high degree of decoupling.

6 Conclusions

With the aim of facilitating interaction between producers and consumers of data for the development of services for the smart-city this article has proposed a modular architecture based on technologies (OSGi and Docker) and standard (oneM2M) aimed to support the development of reactive applications based on micro-services. The proposed model has been applied to the intervention mechanism of a case study (i.e., City4Age project) focused on services for elderly people showing potential benefits, in term of effectiveness, of the approach based on the decoupling between the system and the specific requirements of access by producers and consumers of data.

Acknowledgment. The City4Age project has received funding from European Union's Horizon 2020 research and innovation programme under grant agreement No. 689731.

References

1. http://www.reactivemanifesto.org/
2. http://www.onem2m.org/
3. oneM2M-TS-0001 "Functional Architecture", August 2016. http://www.onem2m.org/technical/published-documents
4. Swetina, J., Lu, G., Jacobs, P., Ennesser, F., Song, J.: Toward a standardized common M2M service layer platform: introduction to oneM2M. IEEE Wireless Commun. **21**(3), 20–26 (2014)
5. Glaab, M., Fuhrmann, W., Wietzke, J., Ghita, B.: Toward enhanced data exchange capabilities for the oneM2M service platform. IEEE Commun. Mag. **53**(12), 42–50 (2015)
6. Kanti Datta, S., Gyrard, A., Bonnet, C., Boudaoud, K.: oneM2M architecture based user centric IoT application development. In: 3rd International Conference on Future Internet of Things and Cloud, 24–26 August 2015
7. The OSGi Alliance. https://www.osgi.org/
8. OSGi Release 6. https://www.osgi.org/developer/downloads/release-6/
9. Paolini, P., Di Blas, N., Copelli, S., Mercalli, F.: City4Age: Smart cities for health prevention. In: IEEE International Smart Cities Conference (ISC2), 12–15 September 2016, Trento (Italy) (2016)

10. http://owlapi.sourceforge.net/
11. Horridge, M., Bechhofer, S.: The OWL API: a Java API for OWL Ontologies. Semant. Web J. **2**(1), 11–21 (2011). Special Issue on Semantic Web Tools and Systems
12. http://www.eclipse.org/Xtext/
13. https://www.docker.com/
14. bndtools.org/
15. HUAWEI: Interworking between oneM2M and OSGi. ftp://ftp.onem2m.org/Meetings/ARC/20160516_ARC23_Seoul/ARC-2016-026-Possible_Solution_of_Interworking_between_oneM2M_and_OSGi.ppt
16. Mainetti, L., Patrono, L., Rametta, P.: Capturing behavioral changes of elderly people through unobtruisive sensing technologies. In: 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 22–24 September 2016, Split (Croatia) (2016)
17. OpenData Lecce Municipality. http://dati.comune.lecce.it/