An Avatar Architecture for the Web of Things

Michael Mrissa, Lionel Médini, Jean-Paul Jamont, Nicolas Le Sommer and Jérôme Laplace

***Abstract*—The Web of Things (WoT) extends the Internet of Things considering that each physical object can be accessed and controlled using Web-based languages and protocols.**

In this paper, we summarize ongoing work promoting the concept of avatar as a new virtual abstraction to extend physical objects on the Web. An avatar is an extensible and distributed runtime environment endowed with an autonomous behaviour. Avatars rely on Web languages, protocols and reason about semantic annotations to dynamically drive connected objects, exploit their capabilities and expose user-understandable functionalities as Web services. Avatars are also able to collaborate together in order to achieve complex tasks.

I. Introduction

The future Internet has been envisioned as an Internet of Things (IoT) in which billions of heterogeneous things or objects will be connected to the Internet using wired or wireless interfaces. The pervasive presence around us of a variety of objects will have a high impact on several aspects of everyday-life and behavior of people. These effects will be visible in various domains such as home automation, e-health, industrial manufacturing, logistics and intelligent transportation [1].

The "Web of Things" (WoT) extends the Internet of Things so that physical object can be accessed and controlled using Web standards. To do so, objects are expected to expose logical interfaces through Web services, to describe Web contents and services using semantic Web languages and annotations, and to communicate together through standard protocols in order to provide software interoperability between objects.

Although the number and types of connected objects increases quickly, the WoT is not yet a reality, as several issues must be addressed in order to seamlessly interconnect physical objects, and to make these objects accessible on the Web. Indeed, objects are heterogeneous and are rarely able to communicate with each other as they do not share the same communication protocols, same business model representations (aka interfaces) or abstraction levels [2]. Moreover, objects are usually designed to process basic requests using their sensors and actuators. By relying on Web standard protocols and technologies and providing end-users with semantically-described services, it will be possible to achieve complex tasks that cope with end-users' needs, even if such tasks require cooperation between several heterogeneous objects.

In this paper, we introduce a new kind of software artifact called "avatar". Avatars provide a virtual extension to physical objects. Physical objects can therefore be coupled with avatars, to form cyber-physical objects that are compatible with WoT constraints and infrastructures. Avatars are implemented using a distributed software platform that can be fully deployed on powerful objets, or distributed over resource-constrained objects and a cloud infrastructure. This platform is designed and developed in the ASAWoO project to address the research challenges of the WoT.

Section II overviews related work and summarizes research challenges. Section III details the different contributions introduced by the avatar platform and show how they fit the challenges of the WoT. Section IV gives an overview of the architecture of the avatar platform. Section V presents our prototype and illustrates the feasibility of our work with a use case scenario. Section VI discusses our ongoing results and gives some guidelines for future work.

II. Research Challenges and Related Work

The WoT imposes new challenges regarding the representation of heterogeneous physical objects on the Web (e.g., robots, sensors, camera, mobile phones, connected-watches), their interoperability and their possible collaboration. Indeed, the functional and physical capabilities (i.e., processing, acting, sensing and storage) of objects can be radically different. Moreover, some objects are fixed and can be connected to the Internet using wired links, while others are mobile and can suffer from connectivity disruptions due to their mobility and the short communication range of their wireless interface.

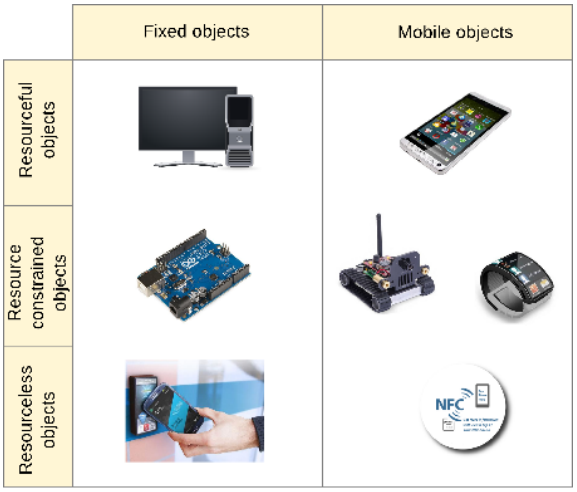

Figure 1. Classification of connected objects.

From our point of view, connected objects can be classified in 6 different categories, depending if they are fixed or mobile and depending on their physical capabilities. Resourceful objects embed a WoT platform that hosts all the services

Michael Mrissa, Lionel Médini are members of the LIRIS Laboratory, Université de Lyon, France, e-mail: {michael.mrissa,lionel.medini}@univ-lyon1.fr

Jean-Paul Jamont is member of the LCIS Laboratory, Université de Grenoble Alpes, France, e-mail: jean-paul.jamont@lcis.grenoble-inp.fr

Nicolas Le Sommer is member of the IRISA Laboratory, Université de Bretagne Sud, France, e-mail: nicolas.le-sommer@univ-ubs.fr

Jérôme Laplace is head of the company "Génération Robots", France, e-mail: jl@generationrobots.com

they provide. Installation of these objects is often simple since they are standalone and do not require additional infrastructure elements to run. Resource-constrained objects cannot embed the whole WoT software platform due to restricted resources, but it is possible to link them to distant hosts that can embed missing parts of the platform. Resourceless Objects are passive objects, detected using unique identifiers such as QR codes or RFID tags. They do not have any computation, storage or memory capability. They can be extended with software deployed on the cloud or on the local network gateway.

The lack of standard specification for developing WoT applications highlights the need for a WoT software platform that extends physical objects on the Web. Based on existing works, we hereafter list a set of design requirements for software platforms to address the main research challenges imposed by the WoT. Such a platform should be based on Web standards, to cope with scalability issues and to able to automatically discover and interact with heterogeneous physical objects [2] (R1: interoperability). It should adapt its behavior and structure to its environment at runtime (R2: Live reactivity), such as the CityPulse[1] platform or the work of the INCOME project [3]. To do so, it should be able to estimate the global cost of physical actions in terms of device usage, computation and networking and determine if an object can perform this action (R3: resource management), as well as to support connectivity disruptions occurring between several mobile objects and between mobile objects and network access points (R4: disconnection tolerance). As expressed in the Webinos[2] project, the platform must be reliable and secure so that objects and applications are harmless and avoid privacy issues (R5: safety). Instead of completely delegating computation tasks to cloud-based infrastructures (see IFTTT[3]), we believe that, as stated in [4], a WoT platform would gain performance by identifying the most suitable location to execute each code module and deploying these modules on the object processing unit or on a cloud infrastructure (R6: delegation). At a higher level, a WoT platform should provide applications that make sense and are useful for the users (R7: user-understandable services) [5] and allow to a set of objects to exhibit a collective behavior [6] (R8: collaboration), as seen in the SensorMeasurement[4] framework. Consequently, a major challenge resides in the ability of the WoT platform to evaluate the intents of the other objects/platforms in order to identify the tasks it is likely to be involved in or it must initiate. Finally, a WoT platform should also favor the emergence of an open market of software components and applications dedicated to connected objects that rely on Web standards. Such a WoT marketplace should permit developers and industrial companies to distribute software applications and components and provide end-users with software pieces allowing them to implement different functionalities into their objects in order to perform various tasks. The Compose[5] project targets such an objective by standardizing WoT marketable applications. We herein propose a WoT platform that takes advantages of existing works and satisfies all these requirements. In the rest of this paper, these requirements are referred to using a reminder of the form “Rx”.

[1]http://www.ict-citypulse.eu/
[2]http://webinos.org/
[3]https://ifttt.com/
[4]http://sensormeasurement.appspot.com/
[5]http://www.compose-project.eu/

III. Contribution

The ASAWoO project aims at extending physical objects by proposing virtual representations of these objects, so that an object and its representation both form a "Web-based cyber-physical object" that defines and embodies high-level behaviors for physical appliances. To do this, we propose a software artifact called an avatar that is dedicated to a particular physical object and represents the virtual part of its corresponding cyber-physical object. The concept of avatar aims at providing hardware vendors, software developers and end-users with a comprehensible abstraction that makes physical objects accessible on the Web and that extends their status and capabilities (processing, acting, sensing, etc.) into homogeneous, user-understandable functionalities. An avatar is built on a software platform that addresses most of the challenges imposed by the WoT and fits the requirements presented in the previous section. It relies on several features described in the following.

A. Semantic description of capabilities and functionalities

Semantically described capabilities and functionalities are a key and original feature of our proposition [7]. In our WoT platform, we describe them using the OWL semantic Web language[6], in order to be able to process and reason about these descriptions using standard semantic Web tools (e.g. triple stores and SPARQL query engines). An avatar discovers the capabilities of an object using introspection techniques and then infers its functionalities using a common ontology. This ontology keeps trace of the different combinations of capabilities required to achieve each functionality, therefore allowing mapping the object layer (i.e. capabilities) with the application layer (i.e. functionalities) in a declarative and loosely coupled manner (R1). This way, a functionality, a service and a WoT application can be performed in different ways depending on the current environment and available objects. Reasoning about semantic descriptions of the capabilities and functionalities also help inferring complex functionalities involving sub-functionalities, which allows defining high-level representations that better match the users’ needs (R8) than the low-level capabilities (e.g. a user can tell a robot to move forward to another room, instead of piloting each of its wheels individually). Semantic descriptions of capabilities and functionalities also provide intelligent resource management facilities (R3) that are used to meet several of the other challenges, as explained in the following sections.

B. Context-Aware Adaptation

Objects can have heterogeneous capabilities and can be used in various environments. Therefore, an avatar must be able to run on different types of objects, regardless of their

[6]http://www.w3.org/TR/owl2-overview/ for details.

type and manufacturer. We address this challenge (R2) by designing a multi-level, semantic context model, as well as its processing engine. This model can be processed at different abstraction levels and must respond to requests regarding four adaptation goals: code deployment (find out where to deploy the different application modules (R6)); avatar-object communication (choose the most appropriate communication protocols and schemes for a given task (R4)); functionality exposition (decide whether an object has enough resources to perform a task or not (R3) and if the task is safe (R5)); collaborative behavior (take part in negotiations to achieve collaborative functionalities (R8)). In order to achieve these goals, this multi-level context model relies on QoS data, as well as high-level information from both the user's and object profiles and from external Web services. For deciding how to compose these contextual information pieces, making them available to use and keeping them operational, we build from the literature [8], as well as from several projects, such as the semantic, multi-scale adaptation model developed in the INCOME project [3]. Our context engine will be both compatible with the high-level features defined in the project and the contextual and QoS data provided by the objects or other resources. Technically, we plan to implement the context engine as a service continuous substitution mechanism, such as the one described in [9].

C. Disruption tolerant service provisioning

Both the free movements of mobile devices (e.g., robots or handheld devices carried by people) and the short communication range of wireless interfaces, such as Wi-Fi, accompanied with the radio interferences induce some frequent and unpredictable disruptions in the communication links. These disruptions can also result from the volatility of devices, which are frequently switched off due to their limited power resource.

To cope with these issues, routing protocols devised for networks that suffer from frequent and unpredictable disruptions, such as delay tolerant and opportunistic networks, implement the "store, carry and forward" general principle [10]. When two devices cannot communicate directly because they are not in the transmission range of each other, these protocols make it possible to exploit other mobile nodes as intermediate relays that can carry a copy of a message when they move and forward it afterwards to other nodes so that it eventually reaches its destination. The provision of application services with this kind of opportunistic and asynchronous communications has been addressed so far only by a few number of research works [11], [12]. The avatar platform implements a disruption-tolerant protocol suite that supports the discovery and the invocation of REST services either with COAP or HTTP. This protocol suite implements both a centralized and a distributed service discovery, and unicast and anycast service invocation models (R4).

D. Autonomous behavior and collaboration

Because it does not have all the knowledge and the skills to reach the associate goals, an avatar must be able to exhibit collective behaviors [6]. Intent recognition [13] in artificial systems appears with the Plan, Activity, and Intent Recognition at the 2007 National Conference on Artificial Intelligence. In our context, this recognition enables the interaction situation identification by avatars. An interaction situation is a set of behaviors resulting from the gathering of agents that act to satisfy their objectives while taking into account their limited resources as well as their individual skills. For example, an avatar can be *indifferent* to other avatars with similar objectives, in the situation where there is not competition for specific resources or skills, *cooperative* with respect to other avatars with compatible objectives that can help compensating for missing resources or skills, or *antagonist* to entities with incompatible objectives. We develop an intent evaluation model devised to allow each avatar to assess the interaction situation from its own partial representation of the world. A common semantic model will provide the underlying background of this contribution. It will be coupled with a network listening protocol that will allow avatars to reason about the skills and objectives searched by other avatars. Moreover, this work includes studying the relevance of a collective, decentralized intent evaluation algorithm that will enable a trusted group of agents to evaluate the intents of another agent. Once the interaction situation has been identified, a collaboration strategy will be devised by the avatars involved in the collective task (R8).

E. Semantic service composition

In the avatar architecture presented in the following, applications describe configurable orchestrations of functionalities. Avatars expose the functionalities they provide as RESTful resources, and as well take the responsibility to expose and run applications. We envision collaborating avatars that manipulate semantically described RESTful resources to drive the execution of applications. However, the automation of discovery and management of calls to resources to fulfill a high level objective driven by an application remains a challenge at the charge of avatars that should be modeled through interactions in their multi-agent systems. We deem appropriate to explore how the operationalization of the interaction protocols used in multi-agents systems into sets of RESTful services that exchange semantically described messages can be realized. Avatars envisioned as agents should be able to communicate about the applications they host and query other avatars to engage collaborative behaviors (R1, R8).

F. An Avatar-based infrastructure for the WoT

In order to support the dynamic deployment of functionalities, services and application on avatars, we have designed a software infrastructure for autonomous avatars. This infrastructure is composed of a cloud environment and of a set of package and ontology repositories. The package repositories can be perceived as a marketplace for the WoT. They include drivers for connected objects, as well as software functionalities that can be deployed on objects according to their capabilities and exposed as services. These repositories also include WoT applications designed for users to control or to interact with their objects (R7). The ontology repositories are

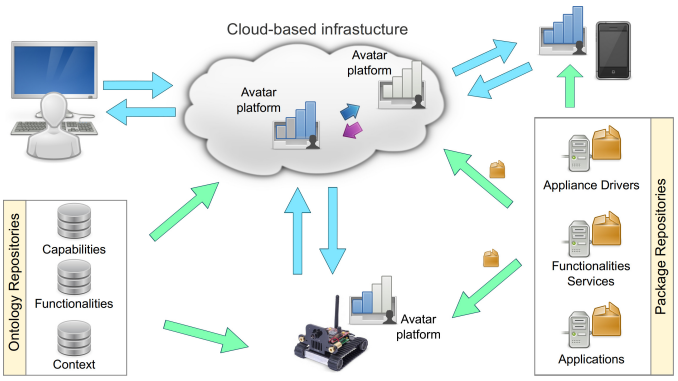

Figure 2. An Avatar-based infrastructure for the WoT.

expected to be queried by avatars in order to reason about the available semantic descriptions (capabilities, functionalities, context data...), to find out what functionalities and application they can deploy automatically, where they can deploy them – on the physical object or in the cloud – and how they must adapt their structure and their behavior according to their execution context. This software infrastructure also allows making avatars accessible on the Web so as to permit end-users accessing their objects using their Web browser. Interoperability between objects is achieved using standard protocols and technologies defined by the W3C for the Web, coupled with a distributed service-oriented mediation infrastructure developed in previous work [14] (R1). For scalability and efficiency purposes, the architecture of the avatar platform matches the REST architectural style. Avatars expose the services they offer as RESTful services, promoting uniform interface, stateless communication and loosely-coupled, late bindings between service clients and providers. Client are transparently bound to the providers of the services they require and are not concerned with service implementation. Communications between avatars are achieved using standards protocols such as HTTP and COAP.

IV. The Avatar Architecture and Lifecycle

A. Overview of the Avatar Architecture

The avatar platform has been designed as an OSGi service-oriented architecture partially relying on the OM2M open-source platform[7], a RESTful implementation of the ETSI M2M standard[8]. Other platforms to support WoT development such as the Lab of Things[9] or AllJoyn[10] either do not follow the REST architectural style, lack platform-independence, or do not follow the ETSI standards. In the avatar, specific services are dedicated to object control while others implement the autonomous, self-adaptive and collaborative behavior of avatars, ensuring interoperability with objects and external software using Web standards. The runtime environment is entirely decoupled from its logical architecture. An avatar can dynamically adapt the distribution of its services to different locations (e.g., object, local network gateway and cloud) in order to improve their execution (R2, R3). The avatar architecture consists of 6 functional modules containing components, described in the following.

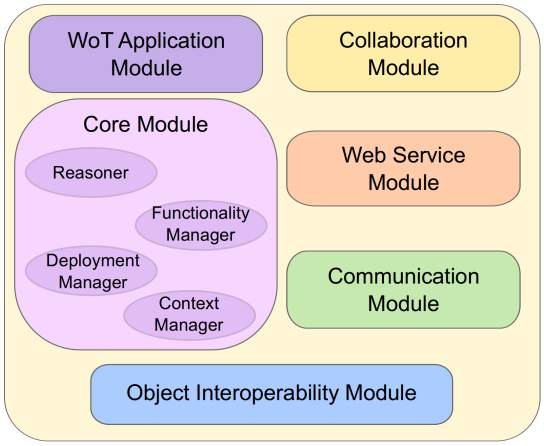

Figure 3. Achitecture of the avatar software platform.

1) Object Interoperability Module: The object interoperability module of an avatar offers a uniform interface to interact with the physical object it is attached to (R1). This module relies on a repository of files and tools to configure the object (i.e. to define the methods that must be used to communicate with it). This module loads and uses the drivers to exchange messages with the object. It performs an introspection of the connected object in order to discover its physical capabilities.

2) Core Module: The core module of the avatar platform includes several components that are reused in different steps of the avatar lifecycle. The main components of this module are respectively a reasoner and a component deployment manager. The reasoner allows to process semantic information pertaining on the capababilities, the functionalities and the execution context of a connected object. The pieces of information provided by the object interoperability module are aggregated with those of the context manager and processed by the reasoner in order to perform semantic multi-level adaptation. These adaptation directives are exploited for instance by the component deployment manager, another component of this module, in order to decide where and when it must deploy the other components of the platform. These directives also help to determine the most appropriate protocol stack for the current communication scheme and contextual conditions. This module also includes a functionality manager, which is responsible for deciding what functionalities can be deployed according to the capabilities of the object and to its current execution context. It also relies on the deployment manager to define where the functionalities must be deployed. This module is essential to respect the (R2), (R3), (R4), (R5) and (R6) requirements.

3) Communication module: The communication module of the avatar platform is responsible for selecting the right network interface and for selecting the right network (Wi-Fi, Bluetooth, Zigbee...) and application protocols (CoAP, HTTP) according to available communication interfaces and

[7]http://eclipse.org/om2m/
[8]http://www.etsi.org/technologies-clusters/technologies/m2m
[9]http://www.lab-of-things.com/
[10]https://www.alljoyn.org/

performance needs (throughput and energy consumption). In order to support connectivity disruptions induced by the mobility of some objects and the short communication range of their wireless interfaces, we have introduced in the communication module of the avatars a disruption tolerant communication protocol suite that relies on the "store, carry and forward" principle and that allows to satisfy requirement (R4).

4) Web service module: The Web service module is designed to expose the functionalities of an avatar as REST services. By this means, other avatars and Web applications can discover and invoke these REST services to interact with the avatar (and the object) to perform a given task. This Web service module also allows an avatar to discover and to invoke remote REST services offered either by avatars or by servers on the Web. This module is thus in charge of implementing the inter-avatar negotiation processes, using a Web service-based communication scheme.

5) WoT Application module: In our avatar architecture, WoT applications expose the high-level behavior of a connected object. They provide users with Web-based interfaces to control and interact with the physical objects. These WoT applications are executed inside a WoT application container that can be partly distributed on the connected object and partly on the Web infrastructure (e.g. a cloud) thanks to the deployment manager. This module satisfies requirement (R7).

6) Collaboration module: Thanks to the core, communication and Web service modules, an avatar is able to identify other avatars and the functionalities they expose. By observing these functionalities and the activity of other avatars, the collaboration module is able to identify if the objectives of the avatar are compatible with those of the other avatars. It can also identify if a conflict with another avatar is likely to occur. According to these interaction situations (antagonism, indifference, cooperation…), negotiation with other avatars can be achieved so as to expose a collaborative functionality as a WoT application. It must be noticed that the location information is an important context property that is taken into account in the collaboration procedure since specific tasks can only be achieved by nearby objects.

V. Scenario and Implementation

We consider the following temperature regulation scenario, freely adapted from [15], to illustrate the operation of our WoT infrastructure. Our user called Bob starts his workday and enters into his office. The following physical devices are involved:

- Bob owns a cell phone that connects to the local wireless network.
- The office room is equipped with a temperature sensor,
- a controllable heater (temperature increaser),
- a controllable air conditioner (temperature decreaser),
- and a window with a controllable motor than can be programmatically opened/closed.

Each of these devices has its own avatar accessible on the local network. More specifically, Bob's cell phone, which has already been connected to the WoT infrastructure, does not get a new avatar instance, but retrieves its latest instance through deserialization. The retrieved avatar hosts a WoT application (WotApp) that performs temperature regulation. The WoTApp has been provided either by the device manufacturer or by a third party. The application has already been configured by Bob to define his preferred room temperature: 21°C. The actual room temperature is 24°C, therefore actions have to be taken to meet the user needs. Let us see how our WoT infrastructure allows these actions to take place without any user intervention.

Upon connection, the avatar of Bob's cell phone broadcasts the network to discover available functionalities. It receives a response from all the other avatars in the room, containing a list of their exposed functionalities that can be accessed as HTTP REST services, along with their associated semantics. The regulation application knows that Bob has expressed the need to have a room temperature of 21°C. Thanks to the reasoning engine, it is also aware that it will need a "temperature sensing" functionality, which is actually exposed on the network by another device, to know the room temperature. The corresponding service is then invoked and the avatar is informed of the current room temperature: 24°C. It is now aware that Bob's need is not fulfilled and initiates a new goal, which is to decrease the temperature to 21°C.

The WoT regulation application has been designed to make use of all devices that can have an impact on the room temperature and also to reason about the outside temperature thanks to a remote service provided by the device manufacturer or a third party. Therefore it first invokes the heater avatar service to make sure that it is turned off. Then, it contacts the external weather forecast Web service, which informs him that the outside temperature is 19°C and that it is a sunny day. As a consequence it decides to open the window instead of starting the air conditioner. To do so, it contacts the window motor avatar and asks it to go to open position, and it invokes the air conditioner service to make sure it is turned off. Reasoning about the different ways to decrease the room temperature is possible with the help of semantic descriptions of functionalities as well as domain specific ontologies developed for this purpose. As well, the relationships between the room temperature, the external temperature and the position of the window are described in RDF terms and exploited in the reasoning process.

Bob found the room too warm and left during the regulation process. However the regulation is not interrupted since it is managed by the avatars of the objects that are still in the room and can communicate together. His temperature regulation WoTApp regularly queries the avatar of the sensor to obtain the current temperature and notifies Bob on its phone when the room has reached the desired temperature, at which point it also closes the window.

VI. Conclusion

The connection between the Web and physical objects is not yet a reality. In this paper, we propose an avatar architecture that enables connecting objects to the Web and improving their skills with additional intelligence. Avatars receive data from the objects they extend and provide reasoning capability

to drive object towards a cleverer behavior, thus naturally improving object intelligence and rising object possibilities to a new level. They form communities and collaborate to realize WoT applications (WoTApps) based on available objects.

Future work includes developing models and protocols to optimize energy consumption during the lifecycle of avatars, to adapt avatar behavior to mobile situations and to dynamically react to environmental changes. Another challenge to explore is the development of WoTApp marketplaces and their application to different domains such as smart home, enterprise or city.

ACKNOWLEDGEMENT

This work is supported by the French ANR (Agence Nationale de la Recherche) grant number <ANR-13-INFR-012>.

REFERENCES

[1] Luigi Atzori, Antonio Iera, and Giacomo Morabito, "The Internet of Things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] Dominique Guinard, Vlad Trifa, Friedemann Mattern, and Erik Wilde, "From the internet of things to the web of things: Resource-oriented architecture and best practices," in *Architecting the Internet of Things.*, Dieter Uckelmann, Mark Harrison, and Florian Michahelles, Eds., pp. 97–129. Springer, 2011.

[3] Jean-Paul Arcangeli, Amel Bouzeghoub, Valérie Camps, Marie-Françoise Canut, Sophie Chabridon, Denis Conan, Thierry Desprats, Romain Laborde, Emmanuel Lavinal, Sébastien Leriche, et al., "Income–multi-scale context management for the internet of things," in *Ambient Intelligence*, pp. 338–347. Springer, 2012.

[4] Matias Cuenca, Marcelo Da Cruz, and Ricardo Morin, "Programming Device Ensembles in the Web of Things," in *W3C Workshop on the Web of Things*, Berlin, Germany, jun 2014.

[5] Stephan Bischof, Athanasios Karapantelakis, Cosmin-Septimiu Nechifor, Amit Sheth, Alessandra Mileo, and Payam Barnaghi, "Semantic Modelling of Smart City Data," in *W3C Workshop on the Web of Things*, Berlin, Germany, jun 2014.

[6] Francisco Cervantes, Michel Occello, Félix Ramos, and Jean-Paul Jamont, "Toward self-adaptive ecosystems of services in dynamic environments," in *Proceedings of the International Conference on Systems Science 2013, ICSS 2013,*. 2014, vol. 240 of *Advances in Intelligent Systems and Computing*, pp. 671–680, Springer.

[7] Michaël Mrissa, Lionel Médini, and Jean-Paul Jamont, "Semantic Discovery and Invocation of Functionalities for the Web of Things," in *IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises*, June 2014.

[8] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos, "Context aware computing for the internet of things: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 1, pp. 414–454, 2014.

[9] Mounir Beggas, Lionel Médini, Frédérique Laforest, and Mohamed Tayeb Laskri, "Towards an ideal service qos in fuzzy logic-based adaptation planning middleware," *Journal of Systems and Software*, vol. 92, pp. 71–81, 2014.

[10] Vinicius F. S. Mota, Felipe D. Cunha, Daniel F. Macedo, José M. S. Nogueira, and Antonio A. F. Loureiro, "Protocols, Mobility Models and Tools in Opportunistic Networks: A Survey," *Computer Communications*, vol. 48, pp. 5–19, March 2014.

[11] Marco Conti, Emanuel Marzini, Davide Mascitti, Andrea Passarella, and Laura Ricci, "Service Selection and Composition in Opportunistic Networks," in *9th International Conference on Wireless Communications and Mobile Computing (IWCMC 2013)*, jul 2013, pp. 1565–1572.

[12] Ali Makke, Nicolas Le Sommer, and Yves Mahéo, "TAO: A Time-Aware Opportunistic Routing Protocol for Service Invocation in Intermittently Connected Networks," in *Eighth International Conference on Wireless and Mobile Communications(ICWMC 2012)*, Dragana Krstic ; Eugen Borcoci, Ed., Venise, Italy, June 2012, pp. 118–123, Xpert Publishing Services.

[13] Gita Sukthankar, Robert P. Goldman, Christopher Geib, David V. Pynadath, and Hung Bui, *Plan, Activity, and Intent Recognition: Theory and Practice*, Morgan Kaufmann, 2014.

[14] Michael Mrissa, Mohamed Sellami, Pierre De Vettor, Djamal Benslimane, and Bruno Defude, "A decentralized mediation-as-a-service architecture for service composition," in *WETICE*, Sumitra Reddy and Mohamed Jmaiel, Eds. 2013, pp. 80–85, IEEE.

[15] Simon Mayer, Dominique Guinard, and Vlad Trifa, "Searching in a web-based infrastructure for smart things," in *3rd IEEE International Conference on the Internet of Things, IOT 2012, Wuxi, Jiangsu Province, China, October 24-26, 2012*. 2012, pp. 119–126, IEEE.