A more human-centric Internet of Things with temporal and spatial context

Marco A. Calderon, Saul E. Delgadillo, J. Antonio Garcia-Macias

Abstract

As the Internet of Things evolves, attention has been paid to adding intelligence capabilities, some of them via semantic technologies. However, many of these developments are still too device-centric and more needs to be done to make the IoT more people-centric. One important aspect in this direction is supporting spatial constructs that are meaningful to humans. We present the OpenThings platform as a basis to provide meaning via semantic technologies and demonstrate how it can be used to build environments that take into account a human-centered view of temporal and spatial context. We exemplify this by building a prototype for a system that supports proxemic interactions and time awareness.

Key words: internet of things; human-centric; spatial context; temporal context; proxemics

1. Introduction

According to Atzori et al. there are several visions or approaches to the Internet of Things (IoT). While attention was first mainly on the “things” and “Internet”-oriented visions, it was clear that these approaches were not sufficient to have an IoT that could support context-awareness, ambient intelligence and other concepts related to truly intelligent systems. Therefore, more attention has been paid recently to the “semantic”-oriented vision; with this approach, the use of semantic technologies can help in modeling objects and information, adding reasoning capabilities and providing semantic communication and execution.

Even with the semantic-oriented vision, most IoT developments have taken a device-centric approach, where smart objects communicate with others and sometimes there are human users taking advantage of them. We are interested in also being able to provide a people-centric approach where systems can be proactive and anticipate human needs. A prime example of contextual variables that have meaning to humans and that can thus be exploited for a human-centered IoT are space and time. Temporal and spatial configurations constantly guide and constrain our thought and action. Previous works have explored the use of spatial meanings within the realm of pervasive systems via proxemic interactions; we revisit these types of interactions and also add temporal context to IoT environments.
In this paper we argue that time and space are two very important contextual variables to incorporate to make a more human-centric IoT; we then present OpenThings (section 1), our platform to facilitate the integration of semantic technologies into IoT environments; in section 2 we show how OpenThings can be used to produce IoT environments that take into account spatial configuration by using proxemic interactions and also incorporate a human-centric view of time. We then proceed to give an example of use by presenting an scenario where all the concepts discussed are applied 3.1. We conclude with some final remarks and outline some future work.

2. The OpenThings platform

We developed OpenThings, a distributed platform containing the tools and services to facilitate the integration of semantic technologies into IoT environments. Its main objective is to distribute semantic information among entities inside it, so it can be used to improve physical integration and spontaneous interoperation of IoT applications. Both being desirable characteristics of a ubiquitous software as described by Kindberg and Fox. OpenThings has been built over UbiSOA, a publicly-available platform comprised of many different services, resources and tools to allow the creation of ubiquitous applications. Instances of such services are localization, environmental primary context providers (wireless sensor networks), identification context (RFID devices), and others. The access to those services is done through standard Web services (via REST but also SOAP is possible) and Web feeds (Atom and RSS feeds). An important advantage of Web services and feeds is that developers can use the same tools and languages they normally use for developing applications. This means, no hardware specifications needed, no need to learn or be tied to a particular language, and no wasting valuable time with these issues. For lack of space, we let interested readers learn about the technical details of UbiSOA in previous works and will focus here on what OpenThings adds.

The OpenThings platform is built over four classes of entities: device services, virtual services, virtual environments, and applications.

Device services are the basis of the OpenThings platform, they encapsulate every IoT device to be used on the platform. Although IoT devices are vastly heterogeneous, using a combination of devices with different capabilities, each device could be embedded on a semantically described web service using HTTP or CoAP. Those services can be announced on a local environment using DNS-SD and make use of SSE or CoAP observe protocols to send events of device resources.

Virtual services are entities that augment device services functionality; used as proxies can improve scalability on constrained devices, or act as priority use controls to allow actuators be used by different applications. Other functionalities can be added via virtual services, for example: an MQTT virtual service can be used to bridge devices that use such protocol outside the platform.

Virtual environments are especial cases of virtual services, they store contextual data of near services (see Fig 1). The contextual data stored should be linked to a physical or conceptual location like a physical building or a work group, that allows the virtual environment to be used as a registry of devices linked to real world environments. Using hierarchically organized virtual environments the whole OpenThings environment can be navigable on a human meaningful structure. In addition, the description of two different contexts for the same device can be stored on different

Fig. 1. Virtual environments containing smart objects (left) and a Linked Data navigation from a Virtual Environment (right).
virtual environments, for example: a cup production and pricing data can be stored on a store virtual environment, but a recipe virtual environment can have data related to its use.

Finally, applications are the entities that use IoT device resources to bring high level services to final users. Applications are free to use every resource via HTTP or CoAP, however a more controlled environment is recommended. Applications could query virtual environments data and use semantic reasoning to find virtual services acting like proxies or other applications using their devices of interest, thus increasing its possibility of successfully running on that environment. Following that recommendation, all applications running should register themselves on the virtual environment of convenience.

The platform focuses on semantic descriptions of devices linked together by a human-meaningful structure, on which we encourage the self-configuration of devices and registered interactions by using semantic descriptions on registry-like services. Although HTTP and CoAP protocols bring flexibility to use different semantic languages, we strongly recommend the use of semantic web standards to allow the use of tools developed for the semantic web or the linked data initiative.

3. Spatio-temporal context for the IoT

We argue that the current state of IoT systems is still too things-centric and that more should be done in order to provide a more human-centric approach; we are not saying that these approaches are mutually exclusive, as the things-centric approach would be appropriate for industrial IoT, for instance, where machines need to communicate and probably no human-in-the-loop is present. However, in situations where context-awareness and proactive systems are in order, it is not only important to have smart objects, but also the spatio-temporal configuration of these objects, as that can carry meaning to humans.

Humans use space to regulate the interaction with their environment, be it with persons or objects. The surrounding spatial configuration carries meaning to human activity. The anthropologist Edward Hall coined the term proxemics to define the study of how people use inter-personal distance to understand and mediate their interaction with others, according to cultural factors. He defined four proxemic zones to help understand the interactions between people: intimate distance (0 to 18 inches); personal distance (18 inches to 4 feet); social distance (4 to 10 feet); and public distance (10 feet to infinity). In the realm of ubiquitous computing and ambient systems, Greenberg argues that “just as people expect increasing engagement and intimacy as they approach others, so should they naturally expect increasing connectivity and interaction possibilities as they bring their devices in close proximity to one another and to other things in the ecology.” In ubicomp, proxemics concern not only people but entities in general, i.e., it concerns the interactions between people, digital devices, and non-digital things. Therefore, Greenberg takes Hall’s dimensions as a starting point but then proposes five proxemics dimensions more adequate for ubicomp (see Fig. 2):

- Distance. The distance between entities can be discrete or continuous. As Hall described proxemic zones, others have defined discrete zones between entities as well as their implications.
- Orientation. Distance is sometimes not enough and more nuanced definitions, such as orientation, play an important role. Orientation can be continuous (e.g., the angle of an object relative to another), or discrete (e.g.,
Fig. 3. Our three ontologies for physical location, identity and preferences, and c) proxemic zones beta.

in front, slightly in front, looking away from, etc.). Orientation makes sense only if the reference entity has a “front face”.

- **Movement.** Captures the distance and orientation of an entity over time. It is possible, for instance, to infer velocity of movement and interpret it as a gesture that carries meaning.
- **Identity.** It describes an entity, ranging from a detailed description (i.e., exact identity and attributes) to a general description that allows to differentiate an entity from others.
- **Location.** Describes the physical context in which the entities reside (e.g., a particular room and its characteristics). Location is important, as the meaning applied to the four other inter-entity measures may depend on the contextual location.

### 3.1. Semantics for space and time

Using OpenThings as the platform, we developed a virtual service that allows to store, consult, update and perceive changes in an ontology related to spatial and temporal configurations. For the representation of time we used the Time Ontology in OWL proposed by the W3C\(^6\). For the spatial configurations, we used the proxemic dimensions proposed by Greenberg et al.\(^5\); based on these dimensions we created three ontologies to indicate a) the physical location, b) identity and preferences, and c) space according to proxemics; Figure 3 shows how these ontologies are visualized using the OntoGraf tool of the Protégé 5.0 software. These three context ontologies incorporate the properties to connect them semantically in order to provide meaningful information that humans understand in a computational model.

The proxemics ontology represents four proxemic dimensions used in this work: distance (with its four proxemic zones), orientation, identity of the person, and stance, this last used to represent if the person is sitting or standing. It also has a locatedIn property to represent physical locations from the places ontology (see Fig. 3); along with the preferences and its hasPreference property describe how a person wishes the environment to behave. To represent the persons we used the existing FOAF (Friend Of A Friend) ontology\(^9\), so developers wanting to represent time and space should jointly use the classes and properties of the corresponding ontologies.

Within the platform, a semantic Jena reasoner (GenericRuleReasoner) uses rules to infer proxemic zones based on the relative distance between two individuals (or an individual and an entity). It also allows to infer the location of an object or person based on proxemic zones, e.g., a person is near an object residing within a room, therefore the person is also within the same room. The inference engine allows the system to not have hard coded behaviors, as these are based on the ontologies. As such, changes are made at a semantic level, in the description of objects, user preferences, etc., that are contained in the Knowledge Base.

For these events to take place, and for the platform to be aware of changes in the environment, several key components are needed to enable a communication channel between them. These components are:

- **The virtual service TripleStoreServer.** An special instance of an OpenThings virtual service, with the purpose of obtaining the physical location of a person.
- **Devices.** Used to represent identity and measure proxemic distances relative to other nearby objects. An application for the Android platform was developed, called SmartNavi, in order to communicate different
OpenThings entities via WiFi; Bluetooth Low Energy is used to calculate approximate distances of smart objects within communication range.

- **Beacons.** Represent physical objects and locations to help devices connect with the appropriate virtual environment. The device uses these beacons with an active scanning of BLE messages that allow to know an approximate distance between them, and also with others within the scanning range.

- **Virtual environment with an integrated context application.** Besides representing objects corresponding to the virtual environment, a semantic reasoner and a knowledge base are integrated to infer and update changes in proxemic dimensions; the virtual environment is notified to update the state of the corresponding smart objects.

We would like to note that other projects\(^\text{10}\) have dealt with proxemic interactions in IoT environments but, to the best of our knowledge, none has also incorporated the temporal along with the spatial dimension to provide enhanced contextual awareness into IoT systems. Another distinctive characteristic of our proposal is that since we approach spatio-temporal awareness via semantic technologies, richer context descriptions and interactions can be made possible, along with semantic reasoning.

4. Proof of concept

In order to exemplify how human-centric IoT systems can be built, taking into account spatio-temporal dimensions, we outline a typical scenario of context-aware systems:

Paul is an architect who has adopted OpenThings as the basis for automation and customization of his smart office. He has defined custom profiles for several situations. For instance, when he arrives, if adequate natural light is detected, the window blinds open up, the electric lights are turned off; also, his workspace is set-up by automatically turning on the computer, although with the monitor off if he is not in front of it, to save energy. So, in a typical workday, as he enters the building door his ID is detected, and up at his office all the previously mentioned setup is performed. That way, as he arrives at the office, he takes his seat but looks away from the computer and instead towards a wall smartboard where all the pending activities are posted; therefore, even if he is within a close distance to the computer, the monitor is still off as the direction indicates no intention of interaction. Once he turns toward the computer, the monitor turns on and he starts working on it. Whenever Paul leaves his desk during the day, the monitor turns off but the computer remains on. It is until the day of the workday, when Paul leaves the office, when his computer automatically shuts down. If one day Paul happens to go quickly to his office to grab a document on a weekend, the office and workspace procedure will not take place, as the profile works only on weekdays.

As stated, the previous scenario is typically found in the ubicomp literature where proactive and context-aware systems are in place to anticipate the users’ needs. However, implementing such scenarios in the realm of Internet of Things systems is not currently a straightforward endeavor, mainly because, as we have argued, the current state of the art (and state of the practice) has advanced a lot in the “things” and “Internet”-oriented visions, but not as much in making the IoT people-centric. It is exactly in this direction that our work focuses on, and we now show how a scenario such as the previous one, where human-relevant contextual elements such as distance (proxemic zones), orientation (to indicate intention), location, time, identity and movement can be used together to construct systems that put the person at the center.

4.1. Scenario implementation

Let us now detail how the platform would make possible the previous scenario. To know the identity of a person and the relative distance between him and a smart object, a device (e.g., smartphone) running the SmartNavi application is used. Before performing the distance update, the device must know to whom it must send the updates within its space. This is where the TripleStoreServer comes to offer the name of the virtual environment (published via DNS-SD) based on the ID of a smart object within range, as well as the distance to it. When a smart object is detected nearby, the device takes the ID and sends it through a RESTful message as a reply. The device now knows to which virtual environment it should connect and transmits the changes of proxemic dimensions relative to the physical entities.

The virtual environment receives the connection from the device and updates the knowledge base to indicate a location change. If there is any change in the environment, for instance, when Paul is in the corridor and opens the
door to enter his office, the smart objects inside are within range; the device then sends a message with the unique id of one of the smart objects found to the TripleStoreServer in order to get the name of the virtual environment representing the office. When it gets the reply, the device connects to the virtual environment sending the id and the approximate distances relative to the smart objects. Therefore, the location is updated (from corridor to office) as well as the relative distance to objects in the environment.

At this point the semantic reasoner comes into play. Each time the knowledge base is updated, the reasoner looks for the relevant changes between objects, in order to determine changes in the spatial configuration around the person (distance, orientation, location), as well as the time when this is taking place (date, time). As seen in the scenario, as Paul is approaching his office when the time context variable indicates it is a workday, then the virtual environment waits for changes in distances notified by the contextual application and Paul’s preferences profile for its office and workspace. This way, the virtual environment receives changes in distances and takes the preferences to adequate the environment as indicated, which in this case is to raise the window blinds (as enough natural light is sensed), to turn off artificial lights, and to turn on Paul’s computer. It is worth noting that if one of the conditions is not met, for instance the time context variable indicates it is not a workday, then even if the other conditions are met, no changes in the environment take place.

To customize and adequate the environment, the orders should be sent to a HubServer service, which concentrates all messages and is in charge of sending the notifications to the smart objects, so they can take the necessary actions to adapt to Paul’s preferences.

5. Conclusions and future work

Our aim with this work is to make the IoT more human-centric. With this in mind, we incorporated context elements that carry meaning to people, such as time and space; the latter was based on the proxemic dimensions for ubicomp defined by Greenberg\textsuperscript{5}. For both, temporal and spatial dimensions, we defined ontologies to describe knowledge about them; these are then used to make rules to implement into semantic reasoners and consequently define context-aware and proactive behaviors. This allowed us to make some proof of concepts that show that it is feasible to put people at the center in IoT systems.

Although we developed several ontologies and then used them with a semantic reasoner, such reasoner is rule based only. We think other types of reasoners should be explored to allow more power and flexibility, but also to decrease reasoning times. Along these lines, a schema of actions or recipes based on an event-condition-action model could be implemented; this could be easily interpreted by reasoners and would make the system more modular and understandable by developers not familiar with semantic technologies.

We made some proof of concepts with several contextual variables, but some important ones were not incorporated. For instance movement, by combining orientation and distance, which would allow to infer changes in relative
distances between entities, or intentions as a person moves toward another person or an object. There are some aspects that were out of the scope of our current work, but that deserve great attention, for instance security and privacy.

Acknowledgments

This work has been partially funded by the Mexican National Science and Technology Council (CONACyT).

References