Top-Down Delivery of IoT-based Applications for Seniors Behavior Change Capturing Exploiting a Model-Driven Approach

Adriana Caione, Alessandro Fiore, Luca Mainetti, Luigi Manco, and Roberto Vergallo

Original scientific paper

***Abstract*—Developing Internet of Things (IoT) requires expertise and considerable skills in different fields in order to cover all the involved heterogeneous technologies, communication formats and protocols. Developers and experts ask for new solutions that speed up the prototyping of IoT applications. One of these solutions is Web of Topics (WoX) middleware, a model-driven Cloud platform that aims to ease IoT applications developing, introducing a strong semantic abstraction of the IoT concepts. In WoX, almost all the IoT entities and concepts are limited to the concept of Topic, i.e. an entity containing the value of a feature of interest that we intend to detect. The local counterpart of WoX is L-WoX (Local-Web of Topics), which manages local instances of features of interest, allowing mobile applications to collaborate among them, offering and receiving data to/from smart objects, and enabling the communication with WoX Cloud platform. The presented study leverages WoX approach for showing an experience in rapid design and prototyping of an ambient assisted living system that detects the movements of elderly persons in their home, acquiring data through sensors in an unobtrusive way. Moreover, the paper shows that the chosen model-driven solution is very suitable in a top-down approach, starting from users requirements: the created system simplifies the user-centered design of IoT applications, adopting a full top-down approach from user required to the technological solution.**

***Keywords*—IoT; WoX; middleware; smart environment; behavior analysis.**

I. INTRODUCTION

Innovations in computer technology and communications that have occurred during the last few years, along with the miniaturization of hardware components such as microprocessors and memories, have initiated an era in which every object can potentially be smart. Until a few years ago, no one would have imagined about looking at his own watch to check his heartbeats or to send messages to somebody, or to control by remote what kind of food is missing in the fridge in that moment. Every object is connected and can communicate with the world. This evolution is called Internet of Things (IoT).

In this scenario, every smart object can proactively do something useful for users, but these connected objects are many, different and heterogeneous. So we could have a situation in which there are few developers with the skills needed for implementing applications. As a consequence, those applications could be implemented very slowly. In addition, because of the different technologies and protocols used by the smart objects, we could also have some problems during the creation or the maintenance of the applications.

What we need is an abstraction level between the physical layer, closer to the technologies and protocols, and the application layer. This level has to be robust and easy to manage, in order to prototype applications rapidly and without changing anything in the application architecture when the smart object's technology differs. To satisfy these needs, we have to look for a model-driven architecture, in order to be agreed among a large number of stakeholders, and topic-based, because of the event-driven nature of IoT.

We found all these features in a solution called Web of Topics (WoX) [1][2][3] and in its local counterpart called Local Web of Topics (L-WoX). WoX is a Cloud-based platform, by which smart objects can communicate with applications and vice versa, simply subscribing to a topic in read or write mode using the WoX APIs. A topic is a combination of a feature of interest and a specific location. So, the topic gets the meaning of the value of a certain feature in a certain location (e.g. the *temperature* of the *living room*). Smartphone applications must have a perception of what is the topic of interest, in order to subscribe to it. Also a set of WoX adaptors is present for each IoT technology, both physical (e.g. RFID, BLE, WSN, KNX, etc.) and virtual (social networks, virtual environments, etc.), which uses the values incoming from smart objects to update specific topics.

In this paper, we describe an experience in rapid design and prototyping of a mobile-based IoT solution through the use of the WoX platform. The case study deals with an Ambient Assisted Living (AAL) system prototype created in the context of the City4Age European project [4][5], aiming at monitoring elderly people behaviors. Such a solution can be seen as the starting point for an unobtrusive data collection technique that can be processed through artificial intelligence algorithms, and

Manuscript received January 10, 2018; revised January 30, 2018. Date of publication March 15, 2018.

Authors are with the Department of Innovation Engineering, University of Salento, Via Monteroni 165, 73100, Lecce, Italy.

E-mails: {adriana.caione, alessandro.fiore, luca.mainetti, luigi.manco, roberto.vergallo}@unisalento.it

Digital Object Identifier (DOI): 10.24138/jcomss.v14i1.438

that may be helpful to detect potentially critical situations (frailty, MCI). If compared to other AAL solutions, our system simplifies the user-centered design of IoT applications, adopting a full top-down approach from user required to the technological solution.

This paper extends our previous work [6]. To be more precise, the paper adds, in the Related Work Section, references to contributions to the generation of prototypes for IoT solution; more details are given to the model driven architecture WoX and L-WoX along with to the common data format defined in the City4Age project. Finally the Test and Validation Section is extended including the experimentation architecture and some sample data sent to the central repository of City4Age.

The paper is organized as follows: Section 2 briefly reports on the key related work in the area of e-Health and IoT. Section 3 provides readers with a brief introduction to the WoX and L-WoX platform, and the model on which they are based on. Section 4 demonstrates in detail how our solution works in order to monitor the elderly people behaviors. Finally, Section 5 summarizes our key messages and sketches future research directions.

II. Related Work

In the last years, several IoT solutions for e-Health services have been proposed. We quote the most relevant scientific contributions according to our research work and field.

Many European countries have developed electronic health records management systems with the aim of improving the quality and continuity of care. Patient summary, electronic medical records and electronic prescribing are services that can help making the performance safer and more appropriate for a more effective health care, by providing data and information that are useful to professionals. The Smart Open Services for European Patients (epSOS) project [7] aimed to test the patient summary and the electronic drug prescription on European scale. Subsequently, it has been extended to the definition and experimentation of services designed to allow direct access to the citizens to their patient summary.

Recently, an Italian company named Vidiemme Consulting has started a project that exploits connected living technologies, developed in collaboration with the American VDM Labs. The scope of this project is to improve home care for patients with heart failure diseases thanks to the application of IoT technology and to real time analysis of data coming from medical wearable devices. Those devices are able to draw an accurate and up to date status of the patient's health, by processing data in real time and, if necessary, transforming them into alarms for doctors and the family. In addition, constant monitoring reduces the phenomenon of frequent hospitalizations, because, if critical situations are identified early, they can be traced within normal limits without having to force the patient to run to the emergency rooms, saving money.

Another interesting project is the Service Application Integration (SAI) [8] middleware that has been made up to create a model of care to treat chronic conditions. As known, when some people have chronic diseases, such as chronic obstructive pulmonary diseases, coronary hearth diseases, and so on, an organized care system is needed. The SAI project involves the cooperation of many actors, such as family, doctors, volunteers and so on, that have different responsibilities, aiming to create a service framework that supports information acquisition, integration and sharing in continuous care networks. To obtain this, the SAI middleware is composed by a message bus, which is the application level of the system; an adaptors framework that allows the communication between heterogeneous information systems; and an event processor cluster. The SAI middleware receives incoming messages from devices to enable appropriate control actions, and to support the implementation of the health plan.

The Hydra Middleware project [9] uses modular service oriented architecture and integrates distributed software systems, in order to make possible the communication between heterogeneous systems. These kinds of services are platform independent, therefore they are cut off from the logic and specifics of the supporting system. Among all the devices that attempt to connect to Hydra middleware, we can distinguish Hydra-enabled devices, and non Hydra-enabled devices, which are connected via proxy to enabled devices. A developer does not care about what kind of devices she/he is facing with: she/he simply sees the network as a collection of Hydra Devices. These devices are heterogeneous and use different ways to communicate. With Hydra middleware, the developer has just two tasks: integrating non Hydra-enabled devices and connecting Hydra-enabled devices to a network. Hydra has already been applied in e-health to permit the integration of different devices in one solution. In particular the middleware aims to support medicals care of patients at home.

In addition to the above quoted related work there is also an increasing attention towards the generation of prototypes for IoT solution. For instance, the following ones are IoT Cloud platforms very popular among developers and new adopters.

Google Fit [10] allows users/developers to manage their fitness data and to develop fitness apps using a specific framework. The system is able to store data from a variety of devices and mobile apps.

Xively [11] aims to help developers and companies to turn physical sensors into software sensors, and connect them to Xively's IoT cloud platform quickly and simply. In fact, this system provides a Web-based applications that is able to rapidly connect IoT devices to its Cloud in order to collect data from them.

CarrIoTs [12] is a platform that enables M2M communications. The main advantage of CarrIoTs is that it supports network level scalability. Users can put triggers on various stages of the data processing cycle to push data to an external system.

Paraimpu [13] is a social-aware IoT middleware that allows consumers to add, use, share, and interconnect their RESTful IoT services whether physical or virtual. Things are mapped to either the abstract concept of sensors or actuators in Paraimpu. The key advantage of Paraimpu over other IoT middleware is

the ability for consumers to reuse and share IoT services with others in their social network.

Moreover we report some example of IoT service based middleware.

Global Sensor Network (GSN) [14] is an IoT solution, the feature of which is the virtual sensor abstraction, by which users/developers can specify XML-based deployment descriptors for deploying a sensor. The architecture of this middleware can host multiple virtual sensors and the container provides functionalities for lifecycle management of the sensors, which includes persistency, security, notification, resource pooling, and event processing.

In [15] authors report on Mobile Sensor Data Processing Engine (MOSDEN), a plug-in-based IoT middleware for mobile devices, that allows to collect and process sensor data without programming efforts, and integrate plugins allowing MOSDEN to communicate with sensor hardware. Its architecture also supports sensing as a service model. Moreover, MOSDEN is developed in such a way that it is interoperable with other cloud-based middleware solutions such as GSN.

A noteworthy work is described in [16] in which the authors propose a development toolkit based on a model-driven approach, called IoTLink. It allows developers to realize mashup applications through a graphical domain-specific language that can be easily configured to create an IoT application. IoTLink hides the complexity of communicating with devices and services on the Internet and abstracts them as virtual objects accessible through different communication technologies.

As a last example of service oriented middleware, we mention SOCRADES [17] that abstracts physical things as services using devices profile. SOCRADES simplifies the management of underlying devices or things for enterprise application especially in the industrial automation. It's an extension of two previous projects [18][19].

The emergence of model-driven approach and mashup development give benefits in terms of rapid prototyping. However, there are few works done both in the industrial and academic fields, due to the heterogeneity of IoT nodes and sources. Moreover, the above works are more focused on the technological aspects rather than the design of user-centric applications. The advantage of the solution we propose here is directly connected to the model-driven approach due to a user-centered design. The user, the elderly person in our case study, has guided since the beginning the design process in order to easily develop unobtrusive scenarios. The middleware WoX on which the proposed solution is built, helps the rapid prototyping of IoT solution with a model driven approach that simplifies and reduces the gap between technology and the IoT stakeholders: developers, device manufacturers, business entities, end users. This through a language based on the concept of topic that gives the idea of what is the object of interest (feature) and where it is located (location).

III. WoX and L-WoX model-driven architecture

The personal data capturing system needs a sensing middleware, which provides access to the underlying technologies, hiding their intrinsic heterogeneity and complexity. Furthermore, in order to make the architecture highly scalable, the structure of the middleware should be modular, so that new technologies can be easily integrated in the system.

To make possible this kind of approach, the middleware should be equipped with appropriate software modules, called adapters, which are able to communicate with the sensing technologies according to the respective standards and protocols.

A software framework able to fulfill all the above requirements is the WoX (Web of Topics) IoT platform.

WoX is a model-driven approach for the IoT, with the aim to minimize the gap between people (end users, developers) and technology. WoX abstracts the complexity of the IoT hardware and communication protocols. In WoX, the main concept is the *Topic*, that is the value of a certain *Feature* of interest (taken from a discrete taxonomy, e.g. presence, temperature, or even high-level concepts such as the crowd presence or an excessive stay in a place) in a certain URI-identified *Location*. More rigorously, a feature is any characteristic or entity of the environment that can be perceivable, definable, measurable and/or controllable.

WoX brings two main advantages. The first one is that not only concrete things but also virtual entities can be easily wired up to develop innovative scenarios. WoX concepts are close to the user: anyone can design and deploy custom scenarios. The second advantage is that WoX accelerates the development of applications, by managing the communication toward the heterogeneous IoT things and hiding the respective protocol details. WoX refers to both IoT hardware nodes and IoT applications generically as IoT entities. In WoX, the "sensor" and "actuator" concepts are abstracted the main uniform concept that is the Topic.

The WoX Topics are grouped in three domains: the WoX Cloud, the Local WoX (L-WoX, e.g. deployed on mobile devices), and eMbedded WoX (M-WoX, e.g. deployed on embedded devices). Every entity talks about WoX Topics and can perform the update of the value of a Topic along with the share of its knowledge. According to the scenarios, every entity can decide if forwarding Topic updates to the parent entities or not. As a consequence, WoX entities at the edge of the architecture talk about few specific Topics, while entities at the core have the master knowledge. This interaction is performed through APIs that are different according to the entity types: mobile apps use iOS/Android APIs, embedded devices use C/C++ APIs, software in general uses Java/Python/.NET APIs.

So, the WoX model requires a robust ICT architecture able to face with the high number of IoT entities and Topics, the heterogeneity of the IoT technologies and the intense exchange of messages.

The best-fit architectural design pattern is the publish-subscribe (pub/sub) pattern. Pub/sub is an enterprise integration pattern where senders of messages, called publishers, do not program the messages to be sent directly to specific receivers, called subscribers. Instead, published messages are characterized into classes, without knowledge of subscribers' identity. Similarly, subscribers express interest in one or more classes, and only receive messages that are of interest, without knowledge of publishers' identity.

The Topic class has two member variables: the topic actual value and preferred value. The actual value contains the most recent value for the feature in the location. The preferred value

is used to send and receive requests about the desired topic value.

The pub/sub architecture sits on top of a component that acts like a middleware toward physical technologies in order to guarantee abstraction and transparency. The component is called Hardware Abstraction Layer (HAL) and the whole architecture, shown in fig. 1, is composed of:

- **The Environment Level**: it comprises the physical layer as well as any virtual environment that can generate events. Social Networks chats can be source of events too.
- **The middleware core (HAL)**: it is responsible of querying/piloting the Environment Level and packing event reports for the upper layers. It includes adaptors for communicating with IoT technologies (both physical and virtual) and REST Web services able to receive or send data from or to technologies.
- **The WoX Capturing Application**: it is the architectural level implementing the WoX model. It instantiates the topics, updates them, takes care of the topic map, and make such data available to the end-user apps by a set of REST interfaces.
- **The end-user apps**, running on any kind of device, use the WoX APIs to subscribe to topics.

The *Local WoX* (*L-WoX*) is a subset of the whole WoX architecture running on the personal user device (smartphones and tablets). It replicates the topic-based, model-driven approach of WoX on a local level, and manages the lifecycle of Topic instances available for any WoX-enabled application running on mobile devices. It also guarantees the communication between sensing technologies and client applications.

L-WoX, as soon as it understands that a topic has been updated, notifies the value of the topic to the WoX server, using the WoX APIs. Data are stored on remote servers, and can be accessed by other devices, in which the L-WoX app is installed.

In L-WoX we have the following components:

- The L-WoX service, instanced singleton in the mobile operating system, retaining the topic instances;
- The mobile apps, subscribing to the local topics after binding to the L-WoX service;
- The native sensors APIs, offered by the operating system to access the available sensors;
- A set of WoX adaptors, which use the values incoming from the sensors to update some specific topic;
- A REST client, able to handle the communication with the WoX Cloud.

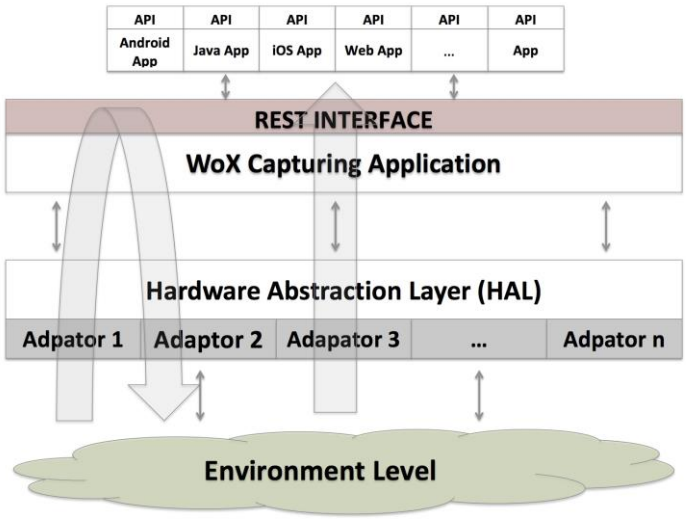

Fig. 1. WoX technical architecture, based on the HAL

IV. The Proposed Solution

In this section we describe the solution we propose in terms of logical architecture, inter-app communication and common date format used for data sharing. We present also the validation made in the scenario of interest.

A. Logical Architecture

The system proposed in this work monitors elder people behaviors, collecting data from a sensor and allowing caregivers and family members to capture in real time useful information, such as whether the elder person is correctly moving or not.

The equipment involved in our case study is composed by:

- A smart wristband, with a sensor tag, that is equipped of a little CPU, by which it is possible to run algorithms that recognize the movement;
- A mobile application (Activity Detection App) connected to the wristband via Bluetooth Low Energy (BLE) technology, that can interpret received data;
- The L-WoX mobile middleware, which receives data from the mobile application installed on the device and forwards them to the WoX server. Both the communications between the Activity Detection App and the L-WoX middleware, and among L-WoX and
- the WoX Cloud are performed invoking APIs.

The designed logical architecture is shown in fig. 2.

The first element of the system is the sensor tag wristband. A little CPU mounted on the wristband interprets raw data incoming from sensors. It runs algorithms that understand some useful features, like the still or moving of an elder person. These features, in the specific location that is the person's position, represent the topic of interest. So the wristband plays the role of source of feature values. The BLE Technology guarantees

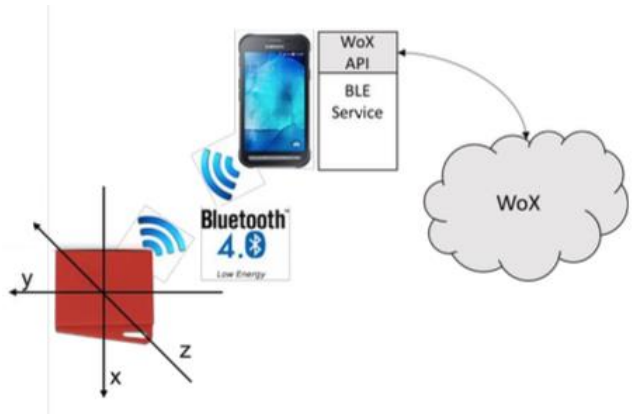

Fig. 2. System logical architecture

the connection between the sensor tag and the mobile application.

The Activity Detection App, in addition to establish and maintain the BLE connection, is able to store some recent data and features as well as to update the values of the still and moving topics to which it is subscribed. This is done in order to better understand the person behaviors, and to send data to the WoX Cloud platform only if actually there is a change in the user status and, as a consequence, in the topic values.

L-WoX receives data incoming from the Activity Detection App and sends them to the remote WoX platform.

It is possible to note that the Activity Detection App plays the role of a bridge between the sensor tag bracelet and the L-WoX middleware, and the role of a filter that sends and updates the subscribed topics using L-WoX APIs, only at certain conditions.

In fig. 3, it is represented the functional scenario. The elderly person is supposed to wear the sensor tag wristband, and to have the Activity Detection App and the L-WoX installed on her/his own smartphone.

Once s/he has these three components working, s/he simply needs to switch on the bracelet and to run the app. The app will make a BLE scan.

Once the Activity Detection App and the smart wristband are coupled, the mobile app starts running a service in background, subscribes to the still and moving topic and automatically receives data incoming from sensor. The app runs algorithms that improve the recognition of movement, basing on data received.

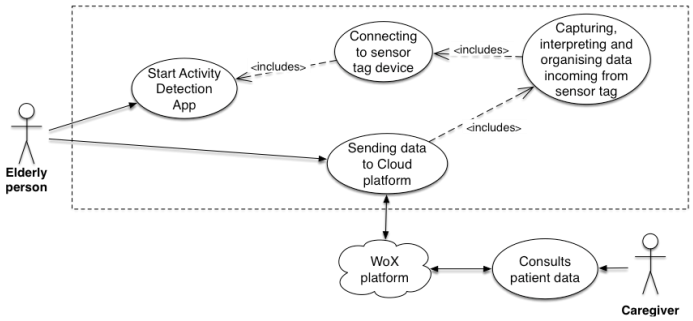

Fig. 3. Use case diagram

When close topic values differ – that means when the elderly person changes her/his status from moving to still or vice versa – the app updates the value of the topic first, and then forwards data to the WoX Cloud. Thanks to the Cloud platform, this information can by analyzed, for example by a geriatrician, a family member or a caregiver, who is using another dedicate app developed for detecting change behaviors.

B. Inter-App Communication

In this section we discuss what enables the communication between the app that monitors the behaviors of elderly people and the dashboards for geriatricians and caregivers.

Two main steps characterize this communication:

- the Inter Process Communication (IPC), that identifies the dialog between two different applications installed on the same device;
- the communication between the L-WoX application and the WoX server.

Here some important steps in order to better understand the logical behaviors:

- At the starting of the Activity Detection App, using L-WoX APIs, the *SensorTagAction* Topic is created and subscribed;
- The Activity Detection App retrieves sensor data and needs to update the *SensorTagAction* Topic, so it calls the method *setTopicActualValue()* of APIs, thanks to which it is possible to update the current value of the Topic;
- The L-WoX app, as soon as it understands that a topic has been updated, sends the value of the Topic to the WoX server, using the WoX APIs;
- Finally, data are stored on remote server, and can be accessed by other devices, on which L-WoX app is installed, or by dashboards.

TABLE I
LEA DESCRIPTIONS

Property	***Description***	***Type***
Action	It is the name of the action executed by a user	URN String
Location	The location where the action is executed	URN String
Payload	- User performing the action - Sensor tag device name - Sensor tag device MAC address	String
Timestamp	The timestamp of the action	String
Rating	A value defining the uncertainty of the inferred action (1.0: max certainty, 0.0: unaffordable)	Float [0..1]
Extra	It can wrap information that are not mandatory but are useful for the current analysis.	JSON
Secret	It contains a token needed to grant security.	String

C. LEA (Low-Level Elementary Action)

All the data gathered from the sensing infrastructure must be collected and validated by the City4Age Platform. This information is then used to derive elderly people's behaviors and then relate them to some risk indicators (RI).

In order to address issues related to heterogeneous data sources, low level technologies, semantic interpretation and so on, City4Age has defined the notion of *Low-level Elementary Actions (LEAs)*.

A LEA is the finest grain atomic information used to detect behavior of elderly people. It relates to start/stop events of user basic actions and contains additional information about time and position of the action that is being taken. All this information is enveloped in the defined Common Data Format and sent to the upper layer of the City4Age Platform.

Each detecting solution needs to comply with this format. In Table 1, all the attributes of LEA objects are described.

LEAs can be grouped in the following macro-categories:

- *Body state*: for tracking user states about motility, like standing, moving, walking, etc.;
- *Indoor home monitoring*: for tracking user movements inside his/her home environment;
- *Presence in indoor places in the city*: for tracking user movements inside monitored places in the city;
- *Presence in outdoor places in the city*: for tracking user movements in outdoor places in the city;
- *Smartphone usage*: for collecting data about the usage of smartphone for calling;
- *Usage of home appliances*: for collecting data about the usage of home appliances, like fridge, TV, washing machines, etc.
- *Interaction with transportation*: for tracking the interaction of user with public transportation systems;
- *Ambient parameters*: for providing additional information about ambient parameters of the place where a given action is executed.

City4Age has defined also the *Measures:* daily indicators that make sense from a geriatric point of view in order to assess changes of behavior. Examples of *Measures* could be: the number of steps (daily), the average speed (daily), the number of walks (daily), the pace of movement (daily), the distance covered (daily) at fast pace, etc.

Based on these daily measures, further indicators can be computed in order to define a risk profile of each elderly person on a monthly base. These indicators are related to the *Geriatric Factors (GEFs)* and *Geriatric Sub-factors (GESs).* A classification of some of them is illustrated in Table 2.

TABLE II
CLASSIFICATION OF GEFS AND GESS

GEF – Geriatric Factors	***GES – Geriatric Sub-factors***
Motility	Walking Climbing stairs Still/Moving Moving across rooms Gait balance
Physical Activity	Physical Activity
Basic Activities of Daily Living	Bathing and showering Dressing Self-feeding Personal hygiene and grooming Toilet hygiene Going out
Instrumental Activities of Daily Living	Ability to cook food Housekeeping Laundry Phone usage New media communication (Skype, Messenger, Facebook, WhatsApp) Shopping Transportation Finance management Medication

D. Testing and Validation

The aim of this Section is to test and validate the solution we propose by illustrating a simple use case focused on the detection of the still/moving state. In general, the user motility refers to the ability of the user to perform activities, such as walking, running, moving, but also stay still, sleeping, etc.

Fig. 4 shows the reference solution based on wearable wristband and a smartphone, acting as a gateway for data gathering and forwarding, for detecting user motility.

Fig. 4. Reference solution for user motility detection

The above mentioned use case is part of a more generic scenario, in the context of the City4Age project, whose high-level architecture is depicted in fig. 5. Data are collected from different ways that are: sensors, external systems, APPs and direct observation.

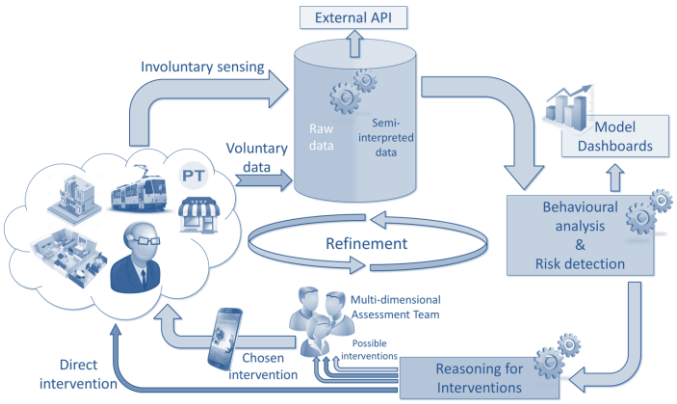

Fig. 5. City4Age high-level architecture

Collected data are gathered in the smartphone that plays a central role in this architecture, because it acts as a gateway for data transmission and as a terminal for interventions.

The raw captured data are processed to obtain LEAs and Measures to be sent and stored in a central repository. Here it is possible to perform complex behavioral analysis and risk detection algorithms.

For testing purposes we use a virtual machine with Ubuntu 16.04 operative system with 2 CPU and 4 GB RAM. The Activity Detection App is installed on a LG G3 smartphone. The test environments involve 16 mobile devices.

When the app is executed, it performs a scanning of the BLE wristband and, once detected, the app starts collecting data.

The elder updates a specific topic moving through the sensor tag wristband s/he is wearing on. The sensor tag sends to the elder's smartphone a BLE message (referring to the correct feature of the topic).

The L-WoX service installed on the smartphone is then able to detect the BLE sensor tag and forward the local topic information to the Cloud along with to any local mobile app subscribed to the considered topic.

Fig. 6 summarizes the elder's status detected from the mobile app.

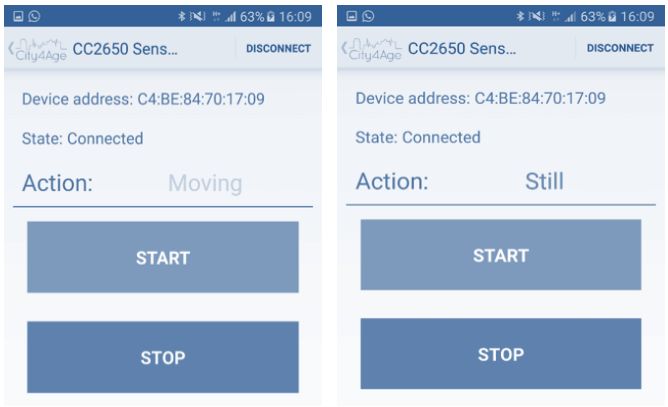

Fig. 6. Person's status detected from the mobile app

From the two screenshots of Fig. 6 it is possible to observe:

- The state of the connection with the sensor tag;
- The elder's action or status detected (Moving or Still);
- The Start and Stop button to, respectively, start and stop the data gathering and forwarding to the cloud.

When the Start button is pressed, the app starts collecting data.

Fig.7 shows the formatted payload of the LEAs transferred to the central repository for start and stop moving actions. Computing the difference between the timestamp of the START_MOVING action and the timestamp of the STOP_MOVING action it is possible to calculate the duration of this "session" of body state along with the STILL_TIME value can be computed and sent as a Measure.

The application of the model-driven WoX and L-WoX approach to build the above-described prototype has demonstrated us that the middleware is definitely able to support a rapid prototyping development. The provided abstractions guarantees time and effort saving if compared to the use of conventional approach where developers are required to deal with heterogeneous communication paradigms and protocols, as well as physical and virtual technologies. WoX and L-WoX are able to encapsulate the technical details and automate some implementation tasks. The L-WoX middleware guarantees the communication between sensing technologies and client applications, receiving data from mobile applications and forwarding them to the Cloud.

```
02-09 16:12:13.401 15018-15018/com.lwox.core D/BleIdentityReceiver:
Topic ricevuto
02-09 16:12:13.401 15018-15018/com.lwox.core D/Topic:
{"action":"eu:c4a:action:START_MOVING","extra":{"pilot":""},"locatio
n":"it:puglia:lecce:testlocation","payload":{"mDeviceAddress":"C4:BE
:84:70:17:09","mDeviceName":"CC2650
SensorTag","user":"eu:c4a:pilot:lecce:user:USER_01"},"rating":0.4,"s
ecret":"token","timestamp":"2017-02-09 16:12:07.181"}
```

Topic 1 - START_MOVING

```
02-09 16:10:48.831 15018-15018/com.lwox.core D/BleIdentityReceiver:
Topic ricevuto
02-09 16:10:48.831 15018-15018/com.lwox.core D/Topic:
{"action":"eu:c4a:action:STOP_MOVING","extra":{"pilot":""},"location
":"it:puglia:lecce:testlocation","payload":{"mDeviceAddress":"C4:BE:
84:70:17:09","mDeviceName":"CC2650
SensorTag","user":"eu:c4a:pilot:lecce:user:USER_01"},"rating":0.4,"s
ecret":"token","timestamp":"2017-02-09 16:10:45.424"}
```

Topic 2 - STOP_MOVING

Fig. 7. LEAs received from the WoX platform

V. Conclusion and Future Work

In this paper, we have presented an experience in rapid prototyping of an IoT mobile-based solution for the monitoring of elderly people behaviors. The case study has involved the use of a sensor tag wristband that periodically sends data to a smartphone application through BLE protocol. It relies on the L-WoX middleware that enables the communication with the WoX cloud platform.

Despite of the related work, the chosen model-driven solution is very suitable when a top-down approach is needed, starting from users requirements. The WoX model and related platform simplify the work and reduce the gap between technology and the IoT stakeholders through the adoption of an intuitive metaphor that is the concept of Topics of interest for the specific domain. Just like humans, IoT nodes and applications can interact and exchange data about topics. Furthermore the WoX model aggregates data and events providing ready-to-use information and concepts.

As future work, in a short time, the developed system will be installed in the elders' habitations and devices for the pilot of the city of Lecce in Italy. The pilot case sees a number of 24 home buildings and 32 elderly.

Furthermore, in relation with the application that monitors the elders' behaviors it would be possible to improve the security and privacy, the automation of connection to a well-known sensor tag, the stability of the routine that collects data along with to allow the support to other action detections, like the use of a lift, or climbing stairs, etc.

Acknowledgment

This work partially fulfills the research objectives of the City4Age project (Elderly-friendly City services for active and healthy ageing) that has received funding from the European Union's Horizon 2020 research and innovation program under the grant agreement No 68973. We also want to thanks Raffaele Prudenzano for his great collaboration in the design and implementation of the presented system prototype.

References

[1] A. Caione, A. Fiore, L. Mainetti, L. Manco, and R. Vergallo, "WoX: Model-Driven Development of Web of Things Applications" in Managing the Web of Things, B. B. Quan Z. Sheng, Yongrui Qin, Lina Yao, Ed. Elsevier, 2017, pp. 357–387.

[2] L. Mainetti, L.Manco, L Patrono, I. Sergi, and R. Vergallo, "Web of Topics: An IoT-aware Model-driven Designing Approach" WF-IoT 2015, IEEE World Forum on Internet of Things. Milan, Italy, Dec. 14-16, 2015, p. 46-51, ISBN: 978-150900365-5, Piscataway, NJ, USA, IEEE, doi: 10.1109/WF-IoT.2015.7389025.

[3] L. Mainetti, L. Manco, L. Patrono, A. Secco, I. Sergi, and R. Vergallo, "An Ambient Assisted Living System for Elderly Assistance Applications". PIMRC 2016, 27th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Valencia, Spain, Sep. 4-7, 2016, p. 2480-2485, ISBN 978-1-5090-3253-2, Piscataway, NJ, USA, IEEE.

[4] P. Paolini, N. Di Blas, S. Copelli, and F. Mercalli, "City4Age: Smart cities for health prevention." In: IEEE International Smart Cities Conference (ISC2), 2016, 12-15 Sept. 2016, Trento.
[5] L. Mainetti, L. Patrono, and P. Rametta, "Capturing Behavioral Changes of Elderly people through Unobtruisive Sensing Technologies." In: 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2016, September 22-4.
[6] A.Caione, A. Fiore, L. Mainetti, R. Vergallo, L. Manco, "Rapid Prototyping Internet of Things Solutions Through a Model-Driven Approach: A Case 99 Study in AAL". SPLITECH 2017, 2nd International Multidisciplinary Conference on Computer and Energy Science, Split, Croatia, Jul 12-14, 2017, p. 99-105.
[7] epSOS, 2017. Retrieved from http://www.epsos.eu/ on April, 4 2017.
[8] F. Paganelli, D. Parlanti, and D. Giuli, "A Service-Oriented Framework for Distributed Heterogeneous Data and System Integration for Continuous Care Networks" CCNC 2010 7th IEEE (2010).
[9] M. Jahn, F. Pramudianto, and A.-A Al-Akkad, "Hydra middleware for developing pervasive systems: A case study in the eHealth domain." In: International Workshop on Distributed Computing in Ambient Environments, 2009, Paderborn, Germany, 15-18 Sep.
[10] Google Fit. (2015.) [Online]. Available: https://developers. google.com/fit/.
[11] *Xively*. (2014.) [Online]. Available: http://xively.com.
[12] Carriots [Online]. Available: https://www.carriots.com/.
[13] K. Aberer, M. Hauswirth, and A. Salehi, "A middleware for fast and flexible sensor network deployment," in Proc. 32nd Int. Conf. Very Large Data Bases, Seoul, South Korea, 2006, pp. 1199–1202.
[14] Global Sensor Networks. (2004.) [Online]. Available: http://lsir.epfl.ch/research/current/gsn/.
[15] C. Perera, P. P. Jayaraman, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "MOSDEN: An Internet of Things middleware for resource constrained mobile devices," in *Proc. 47th Hawaii Int. Conf. Syst. Sci.*, 2014, pp. 1053–1062.
[16] F. Pramudianto, C.A. Kamienski, E. Souto, F. Borelli,L.L. Gomes, D. Sadok, and M. Jarke, "IoT Link: An Internet of Things Prototyping Toolkit" In *Ubiquitous Intelligence and Computing, 2014 IEEE 11th Intl Conf on and IEEE 11th Intl Conf on and Autonomic and Trusted Computing, and IEEE 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UTC-ATC-ScalCom)*, 2009,pp. 1-9.
[17] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, "Interacting with the SOA-based Internet of Things: Discovery, query, selection, and on-demand provisioning of web services," *IEEE Trans. Serv. Comput.*, vol. 3, no. 3, pp. 223–235, Jul. 2010.
[18] H. Bohn, A. Bobek, and F. Golatowski, "SIRENA-Service infrastructure for real-time embedded networked devices: A service oriented framework for different domains," in *Proc. Int. Conf. Netw./Int. Conf. Syst. Int. Conf. Mobile Commun. Learn. Technol. (ICN/ICONS/MCL)*, Apr. 2006, p. 43
[19] S. de Deugd, R. Carroll, K. Kelly, B. Millett, and J. Ricker, "SODA: Service oriented device architecture," *IEEE Pervasive Comput.*, vol. 5, no. 3, pp. 94–96, Jul. 2006.

Adriana Caione is a post-doc research fellow in Information Engineering, on Internet of Things (IoT) and data analysis and processing, at the Department of Innovation Engineering, University of Salento. She is also co-founder of the VidyaSoft srl, start-up of the University of Salento.

Alessandro Fiore graduated in Computer Engineering in January 2011, at University of Salento and he earned the Ph.D. at the same University in 2017 with the thesis "Designing and prototyping middleware for IoT model-driven mobile system". His research interests regard mobile system architectures, software engineering and IoT technologies applied in particular contexts such as E-Health and E-Learning. He is also co-founder of the VidyaSoft srl, start-up of the University of Salento.

Luca Mainetti is an Associate Professor of Software Engineering at the University of Salento. His research interests include web engineering, software engineering, and model-driven IoT engineering. He is a scientific coordinator of the GSA Lab - Graphics and Software Architectures Lab at the Department of Innovation Engineering, University of Salento, and a co-founder of the VidyaSoft srl start-up of the University of Salento.

Luigi Manco graduated in Computer Engineering in October 2012, at University of Salento, with a thesis concerning image-guided micro-invasive surgery systems, after a 6-mounths internship at the Vicomtech-IK4 Spanish research centre. He earned the Ph.D. at the same University in 2017 with the thesis "A Semantic Internet-of-Things Platform Based on Multi-Agent Architecture", after a year-long collaboration with DEIB at Polytechnic of Milano. His research topics follow out two main streams: Agile development methodologies and Software Engineering metrics analysis and semantic-based Multi-Agent Systems for Smart Environments.

Roberto Vergallo is a post-doc in Information Engineering at University of Salento (Italy). He graduated cum laude in Computer Engineering at University of Salento in October 2010, and earned the Ph.D. in 2015. Since 2007 he has been working on several research projects at the same University involving the design and development of middleware for IoT architectures. He is a contributor for the Fosstrak open source RFID platform. He is the inventor of WoX, a Cloud platform for the Internet of Things. In 2013 he worked as research visitor at the SMERC Lab (Smart Grid Energy Research Center), University of California Los Angeles (UCLA). In 2013 he received a special award from BMW. He is co-founder of VidyaSoft srl, a spin-off company of Salento University.