Design and Implementation of a Platform for Wearable/Mobile Smart Environments

Fabio Sartori and Riccardo Melen

***Abstract*—Modern smartphones are a rich and pervasive source of information about the environment. Being equipped with position sensors, movement sensors, barometers, thermometers, etc., they can feed smart applications with a huge amount of data about themselves and the surroundings. Moreover, they can collect data from various kinds of interconnected wearable devices. However, it is hard to exit from a purely local view of the smartphone capabilities and to be able to treat all those information sources as components of a scalable platform, enabling the rapid and effective development of applications in fields such as e-health, smart environments, and smart city. We have designed an architecture, namely, Wearable environment acquisition and representation infrastructure, which allows seeing each sensor as a source of information, which can be dynamically tied to a distributed application. This is made possible by an App, hosted by each smartphone, which can be interrogated about the device capabilities. The concept is demonstrated by the development of an e-health application supporting personalized recovery/training programs. The advantages of this solution for the production process of Internet of Things software consist in a faster application development and in the resulting code being more robust and easily portable. The App can also provide information about the willingness of the owner to contribute a certain amount of data by periodically publishing sensor measurements. In this way, it will be possible to configure smart city applications on the fly, providing, for instance, traffic density information or road bump recognition or noise pollution indications.**

***Index Terms*—Android, application configuration, smart city, smart environments, wearable/mobile sensor information.**

I. Introduction

As INTRODUCED in [1], design and development of smart systems requires cooperation of different technical disciplines. The authors highlight how the recent technological and societal developments promoted by mobile communication systems have drastically changed the approach to standardization landscape in the Information and communications technology sector, enabling the generation of many and potentially competing standards.

This situation is particularly critical for smart environments. As pointed out in [2], *smart environment is what everyone likes to be in these days. Since we live in an automated society, a smart environment would embody this trend by linking computers, smartphones, and Internet into our everyday activity*. The development of smart environments has launched the challenge to inject "smartness" into more "traditional" technologies [1], with many potential ramifications in society (e.g., healthcare), being based on the design and implementation of "smart systems combining cognitive functions with sensing, actuation, data communication, and energy management in an integrated way [...] What separates a smart system from a system that is purely reactive is the knowledge base, which ranges from a set of parameters for a feedback loop to embedded databases and algorithms." [3].

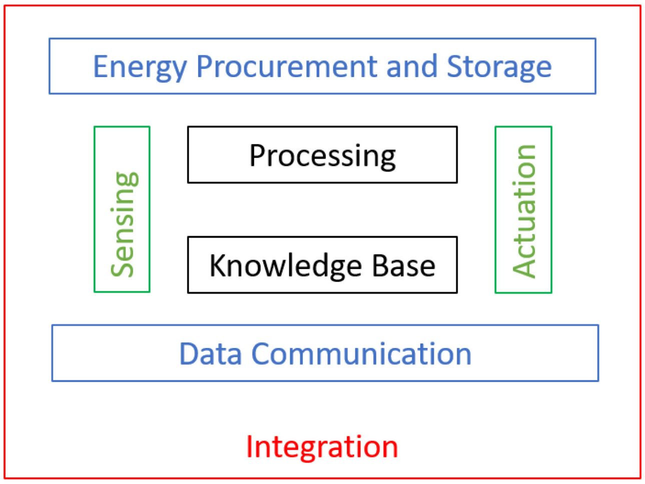


Fig. 1. Draft architecture of a smart system, as proposed in [3].

Fig. 1 presents the typical architecture of a smart system. Different levels can be identified: *actuation* and *sensing* devices define the level of *data acquisition*, aimed at acquiring data from the environment around in a reliable way; *storage* defines the level of *data representation*, where suitable strategies must be implemented to store and share data previously acquired in an effective and efficient way; *processing* and *knowledge base* define the *reasoning level*, where applications must be able to implement (possibly heterogeneous) decision-making strategies exploiting data previously acquired and represented. This article focuses on smart systems exploiting *wearable devices* as data acquisition resources and *wearable expert systems* (WESs) as reasoning applications.

Manuscript received January 13, 2021; revised February 12, 2021 and February 20, 2021; accepted February 23, 2021. Review of this manuscript was arranged by Department Editor T. Daim. *(Corresponding author: Fabio Sartori.)*

The authors are with the Department of Informatics, Systems and Communication, University of Milano-Bicocca, 20126 Milano, Italy (e-mail: fabio.sartori@unimib.it; riccardo.melen@unimib.it).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TEM.2021.3062786.

Digital Object Identifier 10.1109/TEM.2021.3062786

The terms *wearable technology*, *wearable devices*, and *wearables* refer to electronic devices or computers that are incorporated in items of clothing and accessories, which can be worn comfortably on the body. Wearable technology usually provides sensory and scanning features not seen in mobile and laptop devices, such as biofeedback and tracking of physiological functions. Thus, wearables could be profitably used in a number of smart applications, being important sources of data for reasoning. In particular, wearables could be exploited in the design and implementation of a new breed of expert systems.

In a previous paper [4], WESs have been introduced as an innovative paradigm to develop knowledge-based systems capable to modify deliberations dynamically, according to the temporal evolution of data acquired from wearable devices. In that paper, we faced the design and implementation of WESs from the knowledge engineering perspective, focusing on the conceptual model necessary to develop a complete rule-based system from a bottom-up analysis of the relations among the data collected in the environment. In other words, data were assumed to be perfectly reliable and always available, since the final goal was to test the capability of the WES framework to develop a correct reasoning process depending on a variable set of observations.

In this article, we intend to show how the WESs can be connected to real-world wearable devices in order to build effective mobile applications. To this aim, it is important to notice that the data feeding the system are characterized by several parameters defining their suitability to applications. Such parameters (resolution, accuracy, availability, stability, etc.) are all related to the kind of sensor and device detecting them. For this reason, we have extended the WES notion in order to give a complete view of modern mobile applications, defining a wider concept we dubbed *wearable environment*.

This article describes the Wearable Environment Acquisition and Representation InfrasTructure (WEAR-IT), a set of application programming interfaces (APIs) building a bridge between data collected by wearables and Android applications. The aim of WEAR-IT is the development of an intermediate layer between applications, WESs in particular, and data sources. This software layer provides an homogeneous view of a given knowledge domain, where data are acquired from multiple wearable devices and the applications are free to choose the data sources best suited to their needs. This API layer allows the dynamic connection of data sources to distributed application.

The overall goal of WEAR-IT is to improve the production process of Internet of Things (IoT) software, providing the programmer with APIs with a well-defined semantics, which hide the specific implementation details of sensors and wearable devices. In this way, the application development can be made faster, and the resulting code is more robust and easily portable.

An App,[1] accessing the WEAR-IT set of APIs, can be hosted on a large number of smartphones of different manufacturers with different sensors and connected wearables. The App can be interrogated about the device capabilities and about the willingness of the owner to contribute a certain amount of data by periodically publishing sensor measurements.

[1]In this article, we use consistently the term App when referring to the software component installed on the endpoint (smartphone).

The first application of WEAR-IT, described in this article, pertains to the e-health domain. As a matter of fact, e-health is one of the most interesting and promising fields of application of wearable technologies. Wearable devices have radically changed the approaches to healthcare systems design. They satisfy at best the portability and lightweight requirements, ranked in telemedicine surveys as the most important in the patient's perspective [5]. Smart wearable body sensors, devoted to continuously monitoring human's physiological activities and actions, have been demonstrated to be accurate enough to be clinically useful in the treatment of patients. Despite this, they are still underutilized in the healthcare domain [6]. An interesting point of view about this issue is proposed in [7], where four main causes are identified in the Strengths, Weaknesses, Opportunities, Threats (SWOT) analysis made by the authors: in particular, the *lack of interoperability and standardization* in detecting and storing acquired data is critical. The WEAR-IT APIs are able to provide this interoperable and standardized view of the environment and of the data, which can be collected.

Quite naturally, an early application of WEAR-IT has been developed in the mHealth environment. We are now integrating this tool in a distributed IoT platform, and we are extending it with the features necessary to collect the permits from the users and to guarantee a privacy-compliant data treatment. In this way, it will possible to configure on-the-fly smart city applications, providing, for instance, traffic density information or road bump recognition or noise pollution indications. Even if just a small percentage of citizens agrees to install the App, a cooperative environment with tens of applications and tens of thousands of sensors (and users) can be readily created.

The rest of this article is organized as follows. Section II briefly reviews significant literature about wearable devices and their use. Section III describes the wearable environment notion from the conceptual point of view, focusing on the *applications* model. In Sections IV and V, we explain how the WEAR-IT framework has been designed and implemented to provide effective support to WESs. In Section VI, two case studies are described, to show how WEAR-IT can be adopted as either a stand-alone application or as an application program interface to develop WESs. Section VII compares the WEAR-IT framework with similar tools available for Android OS environments. Finally, Section VIII concludes this article.

II. Related Work

A massive work has been carried out in recent years on the definition, specification, and deployment of IoT infrastructures (see, for instance, several well-known surveys such as [8] and [9]). However, the general problem of realizing open application-neutral platforms did not attract much attention and is far from having found a satisfactory solution.

For instance, the proposals aimed at achieving interoperability at the protocol level privilege approaches based on interworking gateways, overlooking the convenience of a common programming environment [10]. Many works stemming from the Web of

Things model (see, for instance, [11]–[13]) provide solutions to the information semantics problem, but they assume implicitly rather complex, resource-rich, endpoint implementations. Moreover, they are not well suited to situations where the applications may be either centralized or (fully or partially) distributed to the endpoints; in our view, providing a programming model supporting any application distribution strategy would be beneficial.

Finally, limited attention has been given to the issues of searching for the needed sensors, enrolling and configuring them on the fly. The study reported in [14] identifies several IoT discovery mechanisms: two of them are adopted in the implementation of the WEAR-IT APIs described in this article.

As reported in [15], wearable sensors have become very popular in many applications, being useful in providing reliable and accurate information on people's behavior. Most of these applications are related to the medical [16]–[18] and sport and training [19]–[21] domains.

According to [22], *ubiquitous healthcare systems* take advantage of a large number of hardware and software components, including wireless sensor networks (WSNs) [23] and wireless body area networks [24], mobile devices, and wireless cloud services, in order to achieve pervasive availability.

Monitoring activities performed by older adults and individuals with chronic conditions participating in "aging in place" programs has been considered a matter of paramount importance. Accordingly, extensive research efforts have been made to assess the accuracy of wearable sensors in classifying activities of daily living (ADL). The feasibility of using accelerometers to identify the performance of ADL by older adults monitored in the home environment has been shown in [25]. A prototype system using in-shoe pressure and acceleration sensor has been proposed to classify activities such as *sitting*, *standing*, and *walking* [26]. Patients affected by degenerative disorders, such as Parkinson's disease, are an important target: Giansanti *et al.* [27] present an accelerometer-based device designed for step counting in patients with Parkinson's disease. Wearable sensors were used in [28] to control the rehabilitation of patients at home after abdominal surgery.

The important facilitating role of wearable technologies for the promotion of a healthy lifestyle has been stressed in [29]. Wearable technologies have been used to monitor physical activities in obese individuals and to facilitate the implementation of clinical interventions based on encouraging an active and healthy lifestyle [30], [31]. Monitoring physiological data can improve the diagnosis and treatment, for instance, of cardiovascular diseases [32].

In this article, we want to overcome the limitations of the architecture of the applications above, which is based on WSNs communicating with fixed stations, where data are collected and managed. Our approach is very similar to the principles of *wireless sensor networks with mobile elements (WSNMEs)* proposed in [33]. In WSNMEs, special support nodes are introduced in addition to sensor nodes and information sinks: these nodes act as intermediate data collectors or mobile gateways.

As reported in [34], mobility in WSNs is useful for several reasons, such as increased reliability and reduced cost of data transmission, connectivity benefits, and energy efficiency. The main difference between WSNME and traditional body sensor networks is the introduction of this intermediate layer, where data from sensors are collected, making them available to interested users. In our model, interested users are WESs, and the intermediate level is implemented by smartphones communicating with an IoT platform; this approach leads to the following definition of a wearable environment.

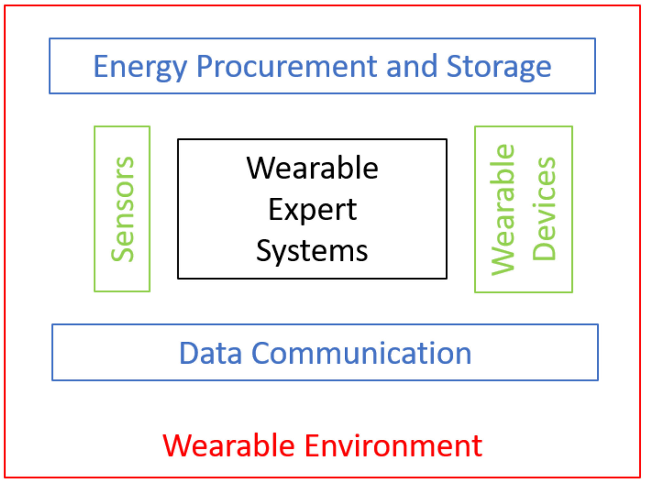


Fig. 2. Wearable environment definition mapped onto the smart system architecture.

III. Conceptual Model of Wearable Environment

A *wearable environment* is a triple

$$W = \{A, MD, S\}$$

where

1) $A = \{a_1, a_2, \ldots, a_n\}$ is a set of *applications*, possibly interconnected;
2) $MD = \{wd_1, wd_2, \ldots, wd_m\} \cup \{sp_1, sp_2, \ldots, sp_k\}$ is a set of *wearable devices* and *smartphones*, possibly interconnected;
3) $S = \{s_1, s_2, \ldots, s_t\}$ is a set of *sensors*.

Fig. 2 maps the definition of wearable environment on the smart system definition and architecture. In a sense, a wearable environment allows us to integrate into a unique conceptual and computational framework the *data acquisition*, *data representation*, and *reasoning* functionalities of a smart system.

In principle, a wearable environment can support every kind of wearable device, like, for instance:

1) *smart-glasses* providing a link to the Internet through a wearable display screen; they overlay data on the user field of vision, allowing him/her to capture pictures and videos by means of integrated cameras controlled by voice and touch;
2) *smart-headphones* allowing the wearer to answer calls by tapping the earpiece; they provide the user with voice-activated dialing functions and motion detectors to perceive when they are worn, to be ready to get commands;
3) *smart-wristbands* and *smart-watches*, which can record physiological parameters such as heartbeat rate (HBR),

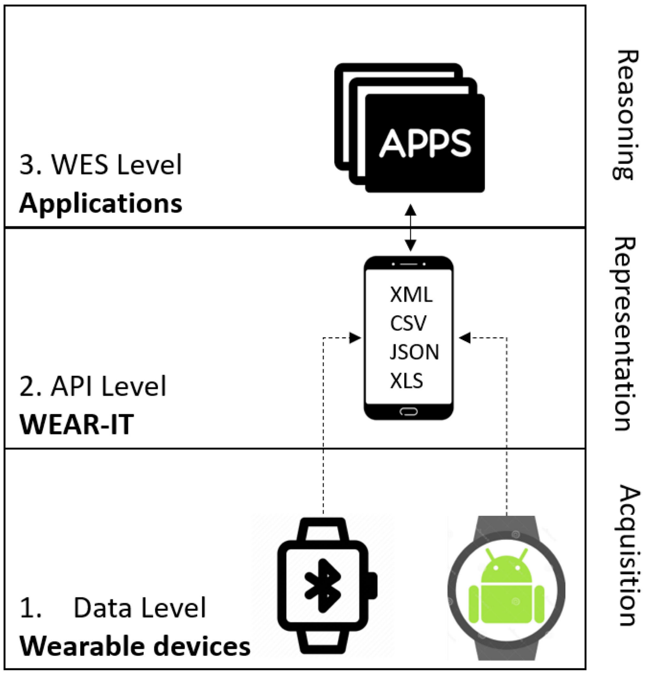


Fig. 3. Sketch of a wearable environment.

calories burnt during the day, and so on; they are usually equipped with movement tracking functions that keep trace of distance walked and the amount of time active;

4) *smart-belt clips*, which can be worn to detect sleep quality and the number of times the wearer wakes.

In a wearable environment, the interconnection among all these devices is mediated by a (physically close) *smartphone*, which plays two roles, being both a data-generating device and a data collection hub for other wearables. For a large-scale application, the wearable environment would include several smartphones, each acting as the hub for a group of wearables: in the following, we shall describe the simplest case of a single smartphone whenever the extension to a larger environment is straightforward.

As shown in Fig. 3, a wearable environment can be seen as a three-tier architecture, where *acquisition*, *representation*, and *reasoning* levels are identified.

Applications at the WES level exploit heterogeneous information and knowledge, coming from different sources, to implement decision-making processes and reasoning strategies. In our approach, the reasoning level of a wearable environment has been thought as an open collection of hybrid knowledge-based systems, composed of *Bayesian networks* (BNs) [35] and/or *production rules*. The WES knowledge base typically changes over time: the observed system and its reference environment evolve through a series of macroscopic states, each one characterized by a specific set of relevant rules. Moving from one state to another, the meaning and importance of some events can change drastically; therefore, the applicable inferences, as described by the rule set, must be modified accordingly.

This is the reason why BNs have been integrated with production rules: they allow us to realize a knowledge-based system, which takes care of variability over time, identifying, for each significant time frame, the set of rules that better fit with the current macroscopic state. Further details about the theoretical and computational framework behind the reasoning level of wearable environments are out of the scope of this article: they can be found in [36] and [37].

Fig. 4 shows the abstraction adopted in the WES model to represent a problem in a time-dependent domain. The state of the system is a collection of descriptors, namely

$$Q/A - MODE = \{Q/A, M, O, DE\}$$

where:

1) *Q/A* is a set of *questions/answers*, which are configuration values inserted manually by the users of the system, e.g., in the e-health context, the patient or the family doctor or, in an open environment, the user of a smartphone granting some permissions to read its local sensor data;
2) *M* is a set of *measurable variables*, whose values are detected by wearable devices, either directly or through computations based on the raw data, e.g., in the e-health context, such variables could be the *HBR* detected by a smart-bracelet or *the amount of physical activity* (PA) accomplished by the user, determined on the basis of the HBR;
3) *O* is a set of variables describing the effect of *outside* events on the system state;
4) *DE* is the set of *deliberations*, which is the list of possible actions to be accomplished in the particular system state according to the implemented decision-making strategy.

The Q/A-MODE, represented in the gray box in the figure, can be related to a set G of *goals*.

A mathematical model has been developed, which represents the evolution of the system state in relation to the decisions taken according to a specific strategy. The model adopted is a dynamic decision network (DDN), a sequence of simple BNs, each representing the situation at a specific time frame (e.g., one week for the case study reported in Section VI-B). The main reason behind the choice of dynamic Bayesian networks (DBNs) (DDNs are a kind of DBN, which embodies decision variables) is their proven effectiveness in modeling evolving scenarios, including the human behavior in several fields such as driving [38], studying [39], and affective states [40].

Depending on the particular aspect of the domain of a given application, the Q/A-MODE, as well as the G set, must be properly defined. This means that some components of the model could be empty in a specific application domain. Moreover, as shown in Fig. 4, the different components of the Q/A-MODE, as well as the external variables G, are bound by unidirectional relations. These relations depict the possibility that the system states can be considered at a specific point in time (i.e., the *present time slice* in Fig. 4) or within a time series of correlated events, where the state at time t directly influences the future behavior of the system at time $t + i$, with $i \in [1, N]$. Accordingly, different decision-making strategies must be considered: in the first case, simple production rules can be adopted; in the second case, DDNs are required to take care of temporal constraints among the different states.

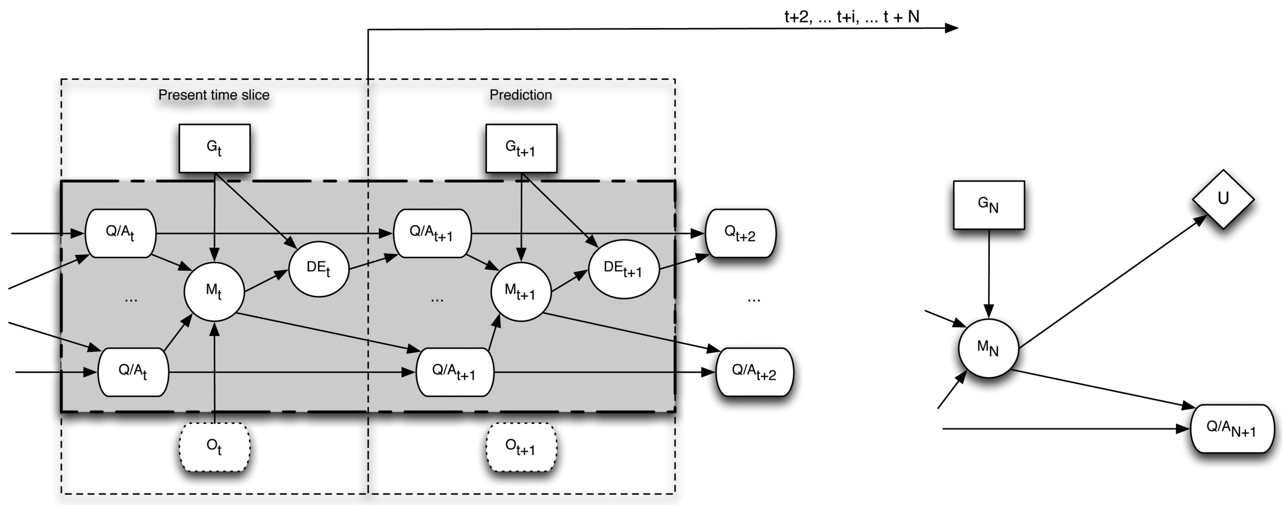


Fig. 4. Sketch of the general Bayesian model of a WES.

IV. WEAR-IT APIs

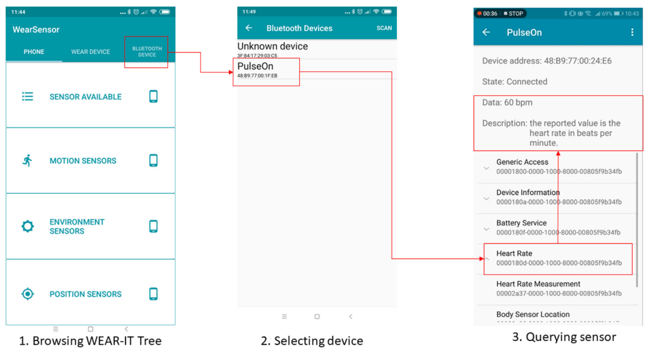
The reasoning level defined in the wearable environment is based, as said above, on the Q/A-MODE model. The main characteristic of the Q/A-MODE is that it embodies both qualitative and quantitative measures within a unique conceptual framework. Quantitative data are time dependent, being collected by the wearable devices included in the wearable environment; of course, the sampling frequency depends on the system domain.

WEAR-IT implements the API level, where the smartphone is the heart of the whole architecture and acts as an intermediate data collector between sensors and applications. A *wearable environment acquisition level* is a nonempty collection of wearable devices, each one equipped with a set of sensors for the detection of raw data from the environment they are acting in. When a new device is included in the wearable environment, its sensors are browsed and queried to detect data: devices are paired with the smartphone and clustered according to the nature of their pairing mechanism. In this way, the wearable environment is organized as a tree, where the root is the smartphone, the leaves are the available sensors, and one or more intermediate levels represent the classification of devices. In the current implementation of WEAR-IT, *Android Wear* and *Bluetooth Low Energy* (BLE) devices are considered. Both Android Wear and BLE are abstracted by the WEAR-IT APIs into the same programming model; therefore, applications can be developed independently from the characteristics of the underlying devices. Since the smartphone itself is provided with many sensors, it also belongs to the acquisition level. A wearable environment must be capable to detect when new wearables enter its range of influence, as well as when they leave it. The API of WEAR-IT provides operations to identify and use *smartphone sensors* and to allow interaction between *wearable devices* (wd) and *smartphone* (sp) and between *applications* (a) and *sensors* (s).

The WEAR-IT APIs are summarized in Table I. First of all, given a smartphone sp, the $sensors(sp)$ operation returns the list of all the sensors detected on sp; the $select(s\_list, s\_type)$ returns a reference to a sensor of type s_type from list s_list; the $classify(sensors(sp))$ operation returns the tree structure of sensors detected on sp, classifying them according to *motion*, *environment*, and *position* labels (see Fig. 5). Moreover, $scan(sp)$ allows us to browse the wearable environment looking for wearables potentially extending it, and $connect(wd, sp)$ allows us to pair sp and one or more wd detected from $scan(sp)$. Note that the implementation of the $scan(sp)$ operation follows exactly the "searching around me" model defined in [14]. The semantics of $sensors(wd)$, $select(s\_list, s\_type)$ (applied to wd sensors), and $classify(sensors(wd))$ is the same as above. Finally, given a sensor s and an application a, the $play(s)$ and $stop(s)$ operations, respectively, allow us to start and terminate the detection of raw data from sensor s, while $store(k, s, a)$ enables the application a to get data from sensor s between two consecutive $play$ and $stop$ operations and save them in a datastore record with key k (see Fig. 5). In the present implementation, the parameter a is redundant because data can

TABLE I
WEAR-IT APIs

API	**Operation**
API_{sp}	sensors(sp)
	select(s_list, s_type)
	classify(sensors(sp))
	scan(sp)
	connect(wd,sp))
API_{wd}	sensors(wd)
	select(s_list, s_type)
	classify(sensors(wd))
API_{as}	play(s)
	stop(s)
	store(k,s,a)



Device address: 48:B9:77:00:24:E6

State: Connected

Data: 60 bpm

Description: the reported value is the heart rate in beats per minute.

Generic Access
00001800-0000-1000-8000-00805f9b34fb

Device Information
0000180a-0000-1000-8000-00805f9b34fb

Battery Service
0000180f-0000-1000-8000-00805f9b34fb

Heart Rate
0000180d-0000-1000-8000-00805f9b34fb

Heart Rate Measurement
00002a37-0000-1000-8000-00805f9b34fb

Body Sensor Location

Fig. 5. Example of devices and sensor selection from a wearable environment in WEAR-IT.

Algorithm 1: WEAR-IT API.

```
Require: $sp$
Ensure: $s, wd, data_s$
  $\{wd_i\} = scan(sp), i \in [1 \ldots k]$
  connect to a device $wd$ suitable for the application
  $connect(wd, sp)$
  identify the sensors in the wearable environment
  $\{s_j\} = sensors(sp) \cup sensors(wd), j \in [1 \ldots m]$
  select the needed sensor of type T
  $s = select(\{s_j\}, T)$
  $play(s)$
  measurement takes place
  $data_s = stop(s)$
  Return $data_s, wd$
```

only be saved within the application, which calls a $store(fp, s)$ primitive, where fp is a file pointer.

V. Implementation of WEAR-IT

WEAR-IT has been developed for *Android* OS: the main reason for this choice is the world-wide diffusion of this operating system on modern smartphones; moreover, many wearable devices, fully compatible with the $AndroidWear$ interface, are available at low costs on the market. Android OS provides a collection of primitives through which the sensors of a smartphone, or an Android Wear device, can be queried. However, the aim of WEAR-IT is the development of a middleware to define a wearable environment, where wearables are not necessarily Android Wear compliant. For this reason, we have considered other methods of interconnections among wearable devices, focusing on $BluetoothLowEnergy$ technology, while other possibilities are the subject of future works (see Section VIII).

Algorithm 1 shows a sketch of the usage of the WEAR-IT APIs. As shown in Fig. 2, the smartphone sp is the only input necessary. Through the *scan* operation, the smartphone is able to determine the set of other available wearable devices, showing them to the user, which can be either a human being (if the graphical user interface is employed) or a software application. The user is then enabled to choose one specific wearable device wd, by means of the *connect* primitive; wd can then return the list of available sensors, and the user can select one of them exploiting the *select* operation. Finally, the *play* and *stop* functions are invoked between two distinct time instants t_1 and t_n, in order to detect raw data from the sensor for the desired period of time. If $t_1 = t_n$, the $data_s$ will contain a single value.

The following subsections describe how the operations of the WEAR-IT API are implemented when wd is an Android Wear device and when wd is a BLE device.

A. Android Wear

As described in Section IV, a sensor must be identified before using it. For this reason, suitable primitives are provided that exploit Android mechanisms for querying sensors, both *hardware* (i.e., concrete electronic components mounted on wearables, capable to measure a physical variable) and *software* (i.e., implementations that simulate the behavior of an hardware sensor, using data detected by other kinds of hardware sensors). The most interesting feature of these mechanisms is that they are completely transparent to the wearable device and/or applications using data: Android is able to hide the virtual nature of software sensors. Table II summarizes the main sensors managed by Android. Given that each wearable device has its own set of sensors, it is really difficult that a device is equipped with all of them. The main sensors are *accelerometer, magnetometer*, and *gyroscope*, and other software sensors based on them can be derived.

B. Bluetooth Low Energy

The devices that do not support Android Wear must be managed differently: for example, to employ the BLE interconnection, the *generic attribute* (GATT) profile has been exploited. The GATT profile adopts the *attribute* (ATT) protocol, which allows a device to read attributes deployed by another device. Each BLE device can be both a client and a server. The ATT specifies how data are coded, but the GATT profile is necessary to access the resources provided. A GATT profile defines a hierarchical data structure that aims at connecting BLE devices. The GATT profile describes a connection context, being composed of one or more *services*. Each service is organized as a collection of *characteristics* or relationships with other services. A characteristic is described by its *typology*, a *value*, a *set of properties*, which denote the operations on the characteristic, and a collection of *permissions*. Moreover, one or more descriptors can be included to describe the characteristic value.

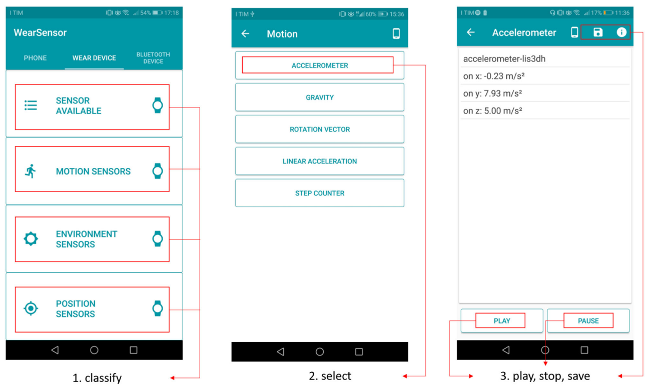
A GATT profile is established through a typical client–server procedure, developed in three steps:

1) *searching for BLE devices*;
2) *initializing the client*;
3) *initializing the server*.

The GATT client sends a request to the GATT server, which stores data transported on the ATT protocol. Then, the GATT server responds and checks for given events occurrences, notifying asynchronously the client when needed. Thus, data provided by available sensors are incapsulated in the profile–service–characteristic structure, with each component identified by a

TABLE II
ANDROID SENSORS

Sensor	**Type**	**Description**	**Common Uses**
TYPE_ACCELEROMETER	Hardware	Measures the acceleration force in m/s^2 that is applied to a device along the x, y, z axes, including the force of gravity	Motion detection (shake, tilt, an so on)
TYPE_AMBIENT_TEMPERATURE	Hardware	Measures the ambient room temperature in degrees Celsius	Monitoring air temperature
TYPE_GRAVITY	Software or Hardware	Measures the force of gravity in m/s^2 that is applied to a device along the x, y, z axes	Orientation and motion detection
TYPE_GYROSCOPE	Hardware	Measures the device rate of rotation in rad/s around each of the x, y, z axes	Rotation detection (spin, turn, and so on
TYPE_LIGHT	Hardware	Measures the ambient light level in lx	Controlling screen brighteness
TYPE_LINEAR_ACCELERATION	Software or Hardware	Measures the acceleration force in m/s^2 that is applied to a device along the x, y, z axes, excluding the force of gravity	Monitoring acceleration along a single axis
TYPE_MAGNETIC_FIELD	Hardware	Measure the ambient geomagnetic field the x, y, z axes, in μT	Creating a compass
TYPE_ORIENTATION	Software	Measures degrees of rotation of a device around the x, y, z axes	Determining device position
TYPE_PRESSURE	Hardware	Measures the ambient air pressure in hPa or mbar	Monitoring air pressure changes
TYPE_PROXIMITY	Hardware	Measures the proximity of an object in cm related to the view screen of a device	Phone position during a call
TYPE_RELATIVE_HUMIDITY	Hardware	Measures the relative ambient humidity in percent	Monitoring dewpoint, absolute and relative humidity
TYPE_ROTATION_VECTOR	Software or Hardware	Measures the orientation of a device by providing the three elements of the device's rotation vector	Motion and rotation detection
TYPE_TEMPERATURE	Hardware	Measures the temperature of the device in degrees Celsius	Monitoring temperature



accelerometer-lis3dh
on x: -0.23 m/s²
on y: 7.93 m/s²
on z: 5.00 m/s²

Fig. 6. Screenshot of WEAR-IT GUI.

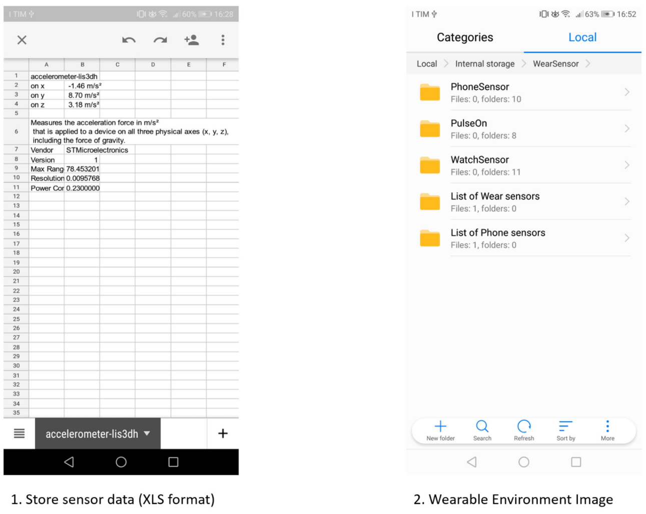
universally unique identifier (UUID), which makes it possible to recover information and data transmitted by devices.

C. *WEAR-IT Usage as a Stand-Alone App*

At launch, the application allows us to choose the device to connect with; this device belongs to one of the WEAR-IT$_{wd}$ subsets. Fig. 6 shows the developed graphical user interface; the result of *classify* primitive is shown in part 1. The wd sensors are grouped as follows.

1) *Sensor available* provides the list of all sensors mounted on the wearable device, alphabetically ordered; this function is useful to have a quick view about all the possible data an application at the reasoning level can exploit from the current device; the list of sensors can be exported to be used by applications.
2) The sensors (e.g., accelerometer) belonging to *motion* category can be exploited by applications interested in the analysis of the user movement, like, e.g., recommender systems for training.
3) The sensors (e.g., light) belonging to the *environment* category can be interesting for applications suggesting actions to take in response to changes in the wearable environment context, like, e.g., personalized entertainment.
4) The sensors (e.g., orientation) belonging to *position* category can be queried by applications interested in the analysis of the user geographic position, like, e.g., systems for suggesting places to eat.

The *select* primitive usage is shown in part 2 of Fig. 6; accessing a category returned by *classify*, it is possible to watch the list of clustered sensors. Then, one of them can be chosen by clicking on its name. This operation brings up the interface in



	A	B	C	D	E	F
1	accelerometer-lis3dh					
2	on x	-1.46 m/s²				
3	on y	8.70 m/s²				
4	on z	3.18 m/s²				
5						
6	Measures the acceleration force in m/s² that is applied to a device on all three physical axes (x, y, z), including the force of gravity.					
7	Vendor	STMicroelectronics				
8	Version	1				
9	Max Rang	78.453201				
10	Resolution	0.0095768				
11	Power Cor	0.2300000				

1. Store sensor data (XLS format)

PhoneSensor
Files: 0, folders: 10

PulseOn
Files: 0, folders: 8

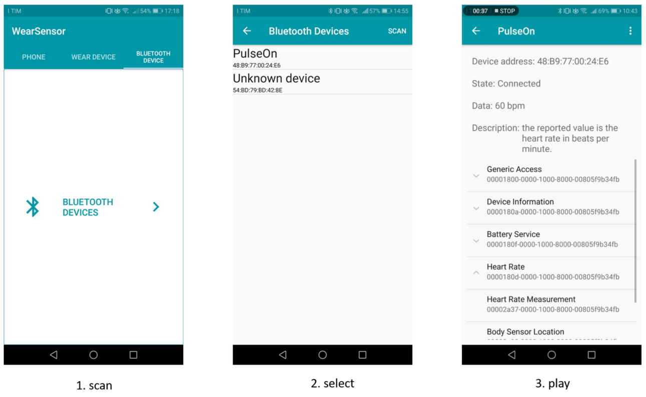
WatchSensor
Files: 0, folders: 11

List of Wear sensors
Files: 1, folders: 0

List of Phone sensors
Files: 1, folders: 0

2. Wearable Environment Image

Fig. 7. Storing sensor data in a persistent way within the wearable environment image on phone disk.



1. scan

PulseOn
48:B9:77:00:24:E6

Unknown device
54:BD:79:BD:42:8E

2. select

Device address: 48:B9:77:00:24:E6

State: Connected

Data: 60 bpm

Description: the reported value is the heart rate in beats per minute.

Generic Access
00001800-0000-1000-8000-00805f9b34fb

Device Information
0000180a-0000-1000-8000-00805f9b34fb

Battery Service
0000180f-0000-1000-8000-00805f9b34fb

Heart Rate
0000180d-0000-1000-8000-00805f9b34fb

Heart Rate Measurement
00002a37-0000-1000-8000-00805f9b34fb

Body Sensor Location

3. play

Fig. 8. Screenshot of WEAR-IT GUI for BLE devices.

part 3 of Fig. 6, where raw data can be acquired and permanently stored by means of *play*, *stop*, and *store* functions.

Fig. 7 shows the result of the *store* operation: on the left, data from an accelerometer are saved in XLS format (other options are, currently, JSON, XML, and CSV). On the right, an image of the wearable environment is produced on the smartphone disk: each wearable device belonging to it is included into a directory structure. In the figure sample, *PulseOn* is a BLE watch, *WatchSensor* is the directory related to an LG AndroidWear smartwatch, and *PhoneSensor* contains sensors data form the current smartphone.

Fig. 8 depicts the results of *scan*, *select*, and *play* primitives for BLE wearables. The difference between this kind of devices and WearOS ones is that BLE devices could be recognized or not according to the implementation of the GATT profile. The current implementation of WEAR-IT is able to recognize only BLE devices that implement the standard libraries of the ATT protocol (e.g., PulseOn in the figure). Therefore, many Bluetooth devices may not be recognized because the GATT profile is not implemented or is implemented in a nonstandard fashion. Moreover, the *select* function returns all the data associated to the GATT profile, like the *service name* (e.g., *Heart Rate*) and the related characteristics (e.g., *Heart Rate Measurement* and *Body Sensor Location*) and their UUIDs.

D. Integration in the Kaa Platform

In order to have the possibility to deploy centralized applications capable of interacting with sensors on a fairly large scale, we integrated the WEAR-IT APIs into the Kaa platform.

Kaa is an open-source platform, which provides a communication mechanism between a cluster of servers and an arbitrary number of endpoints, along with an important set of auxiliary services including client (i.e., endpoint) authentication, registration, and security (optionally with complete encryption of transmitted data). The centralized Kaa cluster implements load balancing, configuration management/storage (based on a relational DB), and data persistency (employing a noSQL DB). The endpoints support an instance of the Kaa SDK, which implements all the client-side functions and provides a set of APIs for the user applications. In our implementation, a Kaa endpoint coincides with an instance of a wearable environment, with the smartphone hosting the Kaa SDK integrated within our client-side application.

The communication mechanism allows specifying the data formats and the upload rules (fixed frequency, fixed amount of data collected, on-demand, etc.). Moreover, a notification mechanism is provided, which includes both system standard and user-defined notifications: user notifications are the key mechanism that we employ to implement the remote access to the WEAR-IT API.

The remotization of WEAR-IT allows implementing dynamically the communication component of a smart city application. In order to provide a complete support to this kind of applications, we extended our API with the primitives needed to identify and select the endpoints suitable to the specific application needs. The primary extension is a $search(area, N)$ primitive, which, in its simplest form, returns a list of up to N smartphones supporting WEAR-IT presently located within the area boundaries. The present implementation of $search(area, N)$ employs a directory with recent information about the reachable smartphones, i.e., it follows the “searching in directories” model described in [14]. As an example, consider an application that requires a snapshot of the climate conditions (temperature, pressure, and humidity) in a specific area. The implementation could follow the simple schema sketched in Algorithm 2.

VI. Evaluation

In this section, we will present an example to show how the WEAR-IT API can be profitably exploited to develop applications at the reasoning level of a wearable environment. In particular, we describe an e-health application supporting personalized training programs in people at risk from the cardiovascular disease point of view.

Algorithm 2: Distributed Data Collection With WEAR-IT.

```
$\{sp_i\} = search(area, N)$
for all $i$ do
    $\{s_j\} = sensors(sp_i)$
    $st = select(\{s_j\}, TEMPERATURE)$
    if $st <> NULL$ then
        $t = play(st)$
    end if
    $sr = select(\{s_j\}, PRESSURE)$
    if $sr <> NULL$ then
        $p = play(sr)$
    end if
    $sh = select(\{s_j\}, HUMIDITY)$
    if $sh <> NULL$ then
        $h = play(sh)$
    end if
    if $t <> NULL$ then
        $store(sp_i.st, t)$
    end if
    if $p <> NULL$ then
        $store(sp_i.sr, p)$
    end if
    if $h <> NULL$ then
        $store(sp_i.sh, h)$
    end if
    end for
```

The proportion of the population over 60 is growing rapidly, as highlighted in the United Nations' report on *word population aging*.[2] Consequently, the public health will face a significant increase in costs in the next future, primarily related to noncommunicable diseases (NCDs) such as cardiovascular and respiratory malfunctions, cancer, diabetes, and obesity. PA has been identified as a crucial protection factor against NCDs; despite this, large part of the population does not know and/or follows the suggested PA guidelines, continuing to conduct a sedentary life [41]. While traditional behavioral interventions have produced scarce results from the PA promotion point of view [42], the availability of new technologies, in particular wearable technologies, can be exploited to develop new applications to support people in modifying their PA behavior.

Although this case study cannot be properly related to the *smart city* or *IoT* domains, it presents some interesting characteristics to show the potentialities of WEAR-IT in the design and implementation of such applications.

1) Most of potential users are interested in the use of tailored applications, which can be extended on the basis of their goals, according to their attitudes.
2) They could be interested in accomplishing group activities; this means that a wearable environment definition, where many users are linked into a unique conceptual and computational framework and Kaa acts as a collector of information coming from more than one source, could be useful.
3) They should be free to decide which specific wearable device to acquire to feed the application in the proper way; data detected should be saved in an open format rather than a proprietary one, in order to be easily accessible by the applications.

[2][Online]. Available: http://www.un.org/esa/population/publications/WPA2009/WPA2009_WorkingPaper.pdf

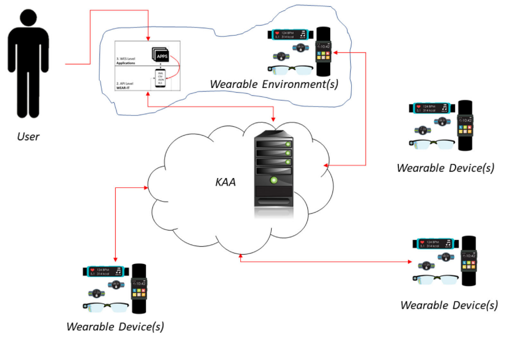


Fig. 9. Information flow in the proposed scenario.

Fig. 9 shows a sketch of the information flows among the user, the wearable environment(s), and the Kaa platform in this scenario, on the basis of what stated in Section V-D.

In our case study, the user exploits an application suggesting him/her how much PA to accomplish during the week; the evaluation is based both on quantitative physical (e.g., how much time do you train during a week?) and on qualitative psychological variables (e.g., how well did you feel during the training session?). From the physical point of view, the *metabolic equivalent of task* (MET) variable is used to estimate the amount and intensity of PA accomplished.

Here, we are interested in exploiting the relationships between MET and HBR, given by the following formula [43]:

$$\mathrm{MET} = 4 * \mathrm{Time}^{\mathrm{MPA}} + 8 * \mathrm{Time}^{\mathrm{IPA}} \quad (1)$$

where $\mathrm{Time}^{\mathrm{MPA}}$ and $\mathrm{Time}^{\mathrm{IPA}}$ are the periods of time the subject is involved in *moderate* and *intense* PA, measured in minutes.

Thus, the wearable environment in Fig. 9 will be configured in order to detect HPR values from one connected wearable device at a given instant, at a given frequency and for a given period of time, as suggested by the application. A PA session is defined *moderate* if the registered HBR values are in the range $[\frac{6*\mathrm{MHR}}{10}, \frac{7*\mathrm{MHR}}{10}]$, with $\mathrm{MHR} = 220 - \mathrm{age}$ is the subject *maximum heart rate*, depending on his/her *age*. A PA session is defined *intense* if the registered HBR values are in the range $(\frac{7*\mathrm{MHR}}{10}, \frac{8*\mathrm{MHR}}{10}]$. Thus, the HBR rate values detected from the wearable devices chosen during the configuration of the wearable environment will be received by Kaa and finally stored in suitable formats to be easily accessed by the application at the reasoning level of the wearable environment.

To take care of psychological well-being, the *self-efficacy* (SE) variable [44] has been considered. SE, also referred as *personal efficacy*, is the extent or strength of one's belief in his/her own ability to complete tasks and reach goals. Psychologists have studied SE from several perspectives, noting

TABLE III
DECISION RULES AND RATIONALE FOR SETTING NEW WEEKLY GOALS

Condition	**Goal for the new training period (G)**	**Rationale for the goal setting strategy based on the relevant literature [45]**
PA goal achieved & $SE \geq 3$	Increase PA goal	Setting a harder goal is challenging but doable for the person, because it is in line with the physical capabilities and supported by strong SE beliefs
PA goal achieved & $SE \leq 3$	Maintain the same PA goal	Maintaining the same goal is a strategy to reinforce the self-efficacy beliefs through the achievement of the same goal, thus trains the person for successive more difficult goals
PA goal missed & $SE \geq 3$	Maintain the same PA goal	Maintaining the same goal is a strategy to avoid disappointing motivations and self-efficacy beliefs, thus provides the person with a further opportunity to achieve a goal corresponding to his/her SE beliefs
PA goal missed & $SE \leq 3$	Decrease PA goal	Setting an easier goal is a strategy to allow the person to become familiar with the behaviour through an easier task and reinforce self-efficacy beliefs through more likely successful experiences

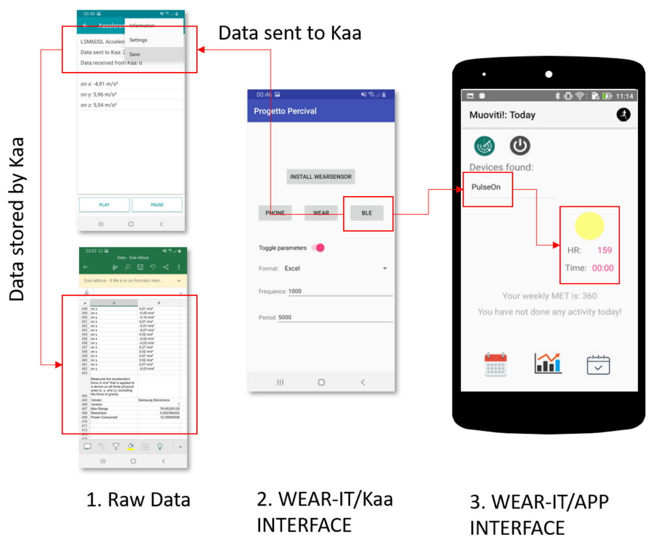


Fig. 10. Three stages of app development in the case study.

various paths in the development of SE; the dynamics of SE, and lack thereof, in many different settings; interactions between SE and self-concept; and habits of attribution that contribute to, or detract from, SE. SE is evaluated through a collection of questions about the positive or negative conduction of training sessions; possible answers are values on a [1...4] scale, and the mean of such values returns the whole SE on a week.

The reasoning process is implemented by a rule-based system to properly link physical and psychological variables to obtain suggestions. Table III summarizes it.

Fig. 10 shows the role of WEAR-IT in the development of the relative WES. The module starts from raw data acquired by wearable devices (e.g., the HBR) to provide suggestions about the amount of PA to accomplish week by week in order to avoid cardiovascular problems (further details about the decisional model in [46]). The API of WEAR-IT has been used by the reasoning level of the wearable environment architecture. The $scan(sp)$ operation returns $wd = \{PulseOn^{TM}\}$ and the $connect(PulseOn^{TM}, sp)$ pairs the wearable device with the smartphone. Then, the $select(sensors(PulseOn^{TM}), HeartRate)$ primitive is invoked by the application, to activate the *Heart Rate* sensor (see part 3 in Fig. 10) exploiting the $sensors(PulseOn^{TM})$ operation. The $play(HeartRate)$ can be activated to start the detection of patient heart rate, till the end of training session, when the $stop(HeartRate)$ is invoked; finally, the $store(xfp, HeartRate)$ operation is applied, to store acquired data in XLS file, which is the starting point of reasoning step in the application.

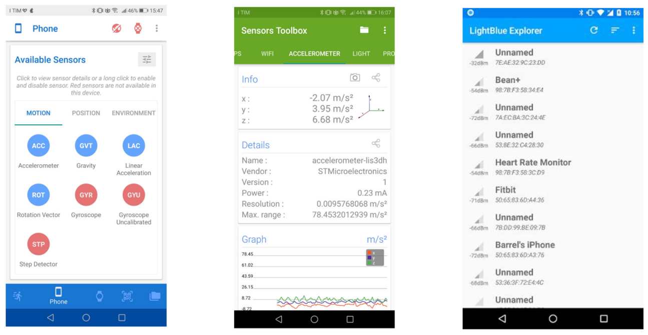
VII. DISCUSSION

A. Comparison With Similar Conceptual Tools

The wearable environment notion is a first attempt to standardize the information cycle among applications heterogeneous from both the reasoning and data acquisition and representation standpoints. As observed by Folmer and Jakobs [1], standardization of smart systems is a complex activity, characterized by multidisciplinarity and diversity over, at least, two dimensions: *domains* and *stakeholders*.

In our view, these dimensions are expressed by applications at reasoning level: applications implement different decision support systems, related to the same problem domain or not. These applications are exploited by different users, each one with his/her own goal to meet. These goals can be completely different or correlated, for example, when the application is capable to provide users with different views about the same results; in the mHealth example above, a patient could be interested in understanding if his/her physical effort has been good from the waistline reduction perspective, while a doctor in understanding if the patient shows good behavior change attitudes over the time. Applications and, consequently, users must be interconnected for gaining access to the same data sources. Moreover, the outputs of an application must be used by another one as inputs, to promote cross-fertilization among them and, consequently, maximize the benefits for the users.

The WEAR-IT level is responsible for the practical implementation of this interoperability, providing applications with primitives that enable them to be aware of the environment where they operate. As highlighted in [47], *knowledge development and diffusion* is one of the main objectives of smart systems standardization, *as it provides a forum of collective cognitive processes where actors with heterogeneous backgrounds discuss new ideas, enabling user-oriented market-driven innovations*, being *an effective channel of knowledge transfer from the R&D base, where various stakeholders can share best practice and*



1. SensorCap GUI 2. Sensors Toolbox GUI 3. LightBlue$_R$ GUI

Fig. 11. Competitors of WEAR-IT.

state-of-the-art research insights. The WEAR-IT approach allows applications and users to choose the right data source among all the available ones, to exchange results, data, and problem-solving methods with peers, with potential benefits from the increase of overall knowledge within the wearable environment.

Unlike traditional approaches to standardization, we do not focus on the design of conceptual roadmaps [47] or deployment strategies [48] involving policy makers or business units on how to design and implement *static* applications; instead, our approach is devoted to enabling applications to become *adaptive* entities in the integration and interoperability context defined by the wearable environment: through the primitives provided by WEAR-IT, they are able to look for wearables around them, connect to one or more of them, select which sensors to gather data from, and store data in the right format according to their decision support strategy.

In this sense, the WEAR-IT approach is similar to SAREF ontology [49], which, being created to support the smart appliances industry, is more focused on the pragmatic description of domain entities and relationships among them than on the reference to other ontological models (i.e., high-level upper ontologies).

B. Comparison With Similar Computational Tools

WEAR-IT is based on a conceptual and computational framework describing how a wearable environment is composed, what kind of data it can use, and what kind of decision-making process it is able to adopt. In this sense, WEAR-IT is different from similar available tools for querying sensors under the Android OS umbrella. It is important to notice that most of them are not devoted to support the definition of a complete wearable environment, as previously defined, being only interested in the visualization and query of sensors integrated in the device they are installed on (i.e., only the smartphone or a wearable device).

To our knowledge, only three of these applications provide services comparable to WEAR-IT, namely, *SensorCap, Sensor Toolbox, and LightBlue Explorer*; a sketch of their graphical user interfaces is presented in Fig. 11.

SensorCap is an Android app that permits to collect sensors data in *JSON* and *CSV* formats, as well as to modify sensors configurations. The main goal of the app is enabling researchers and developers to store quickly quantitative data for research and experimentation purposes. As shown in part 1 of Fig. 11, it is possible to browse sensors of the smartphone and a smartwatch, if and only if the smartwatch has been previously paired with the smartphone. A specific sensor can be enabled/disabled, and the frequency of data detection can be varied. Time synchronization is provided by the network time protocol, and data can be shared by means of e-mails, messengers apps, and cloud platforms.

Sensor Toolbox provides an efficient way to access the collection of sensors of the smartphone on which it is installed. Real-time data of each sensor are shown through a suitable menu, as well as specific information about a given sensor and a (real time) graphical representation of the data it detects. Moreover, data can be stored within the application itself. Considered sensors are the ones made available by the Android structure described above (see Table II), and they are presented in the menu if and only if the smartphone contains them. It is not possible to stop the detection of data, since when the menu is browsed sensor by sensor, the data detection starts automatically. Moreover, this application only focuses on smartphone sensors, with no possibility to link Android or Bluetooth devices. As in SensorCap, data can be shared with peers through messengers apps.

LightBlue Explorer connects to BLE devices. Through the app, the user can browse the services offered by connected BLE devices, supporting him/her for reading, writing, and notifying data. The received signal strength indication is also shown to quantify the distance from the BLE device.

Table IV compares the definition of wearable environment devices implemented by SensorCap, Sensor Toolbox, LightBlue Explorer, and WEAR-IT: our framework overcomes the performance of each competitor.

Although the applications above can only be used as standalone processes (i.e., they do not provide explicitly an application program interface), Table V compares them with WEAR-IT from the functions point of view, i.e., the API set as described above. Only SensorCap provides all the services provided by Wear-IT: this means that, if its APIs were made available, it could be used to develop applications at the reasoning level as WEAR-IT. Of course, this could be possible only with WearOS devices, given that SensorCap does not interact with BLE wearables. Both Sensor Tool and LightBlue Explorer do not implement the $classify$ primitive: this means that they could be used as a middleware between data and application if and only if the application knows exactly which sensor to use. Anyway, it is important to notice that only WEAR-IT can be fully integrated into a wearable environment at the moment, while the others can be used as external services to collect and store data that can then be elaborated by the application. Fig. 12 shows an example of how the application presented in Section VI elaborates data from $PulseOn$ producing a graph of the HBR and graphs of the calories burned and the intensity of the relative PA session.

C. Implications for the Management of IoT Software

The relevant characteristics of the IoT software ecosystem are the following.

TABLE IV
COMPARISON BETWEEN WEAR-IT AND COMPETITORS INTRODUCED ABOVE IN TERMS OF WEARABLE ENVIRONMENT CHARACTERIZATION, FROM THE wd SET POINT OF VIEW

	Wearable Environment devices characterization				
	$Smartphone$	$WearOS$	$WearOS_i,\, i > 1$	BLE	$BLE_j,\, j > 1$
SensorCap	Yes	Yes	No	No	No
Sensor Toolbox	Yes	No	No	No	No
LightBlue® Explorer	Yes	No	No	Yes	Yes
WEAR-IT	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

TABLE V
COMPARISON BETWEEN WEAR-IT AND COMPETITORS INTRODUCED ABOVE IN TERMS OF API CHARACTERIZATION

	API characterization					
	$sensors$	$select$	$classify$	$play$	$stop$	$store$
SensorCap	Yes	Yes	Yes	Yes	Yes	Yes
Sensor Toolbox	Yes	Yes	Yes	Yes	Yes	Yes
LightBlue® Explorer	Yes	Yes	No	Yes	Yes	Yes
WEAR-IT	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

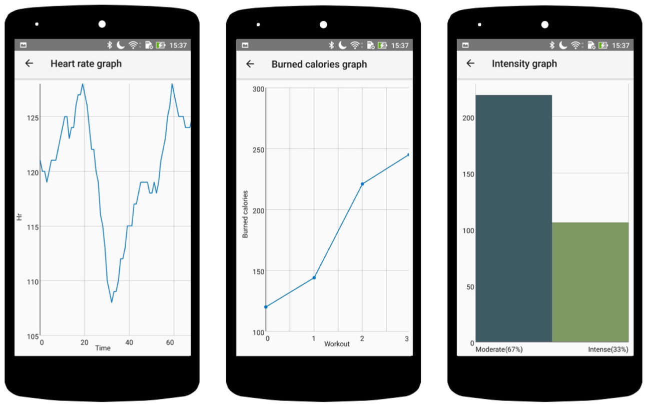


Fig. 12. Examples of outputs from an application at WE reasoning level that integrates WEAR-IT.

1) A large number of applications, often customized, exploiting sensor data.
2) Large heterogeneous ensembles of sensors providing data in different formats.
3) Several IoT networking options.

Considering this scenario, we believe that our approach to the architecture of IoT software, based on the WEAR-IT APIs, bears fruitful consequences both for the development and for the maintenance of IoT applications. As a matter of fact, WEAR-IT is a platform [50] [51], accessible to the application developer as a set of APIs: therefore, we are making a conscious effort of *platformization*.

Why do platforms matter in IoT software? In such a complex and fragmented technological scenario, those who intend to produce innovative applications must be able to concentrate on their essential elements, avoiding expensive and error-prone developments regarding, for instance, data representation, communication protocols, persistency, and so on. Therefore, in this context, the main advantages of the exploitation of software platforms are particularly relevant: they allow faster product developments resulting in more reliable code.

Moreover, if the platform gathers a sufficiently large community of developers, a lot of software adaptations, upgrades, and new functions become available (either as paid or as open-source code) through this ecosystem [52], [53], and this process reduces the obsolescence risk [54] of the product.

In our case, the obsolescence risk is lowered, in particular, by the enhanced portability of applications with respect to the evolution of the underlying software layers (for instance, the certification of WEAR-IT for a new OS release would automatically solve the large majority of compatibility issues) and by the ease of extension to new/different technological paradigms (for instance, handling the direct access to sensors connected by the narrowband IoT is a manageable problem if all the modifications are concentrated in the WEAR-IT implementation).

All these advantages, however, can be obtained only if the platform is maintained in time: therefore, the choice of a specific platform by the technology manager (be it for commercial product development of for advanced application research) must consider, above all, its long-term perspectives. In most cases, several alternative solutions are available on the market: as a matter of fact, any platform is both a value creation tool and a software product, which competes with similar products [50], [55]. The effect of this competition is an unavoidable process of concentration [55], which leads to the survival of very few competitors and the gradual disappearance of the other solutions.

Among the factors that influence the long-term survivability of a software platform, in addition to the size and dynamics of the ecosystem mentioned above, it is necessary to also consider the interoperability [50] and the conformance with existing standards [1], [47], [56]. The importance of standards in the general process of product innovation lies mainly in the opportunity to lower the development risk of new products; the recognition of this role has led us to undertake a significant effort for aligning WEAR-IT with the emerging SAREF ontology (see below).

D. WEAR-IT and the mHealth Software Domain

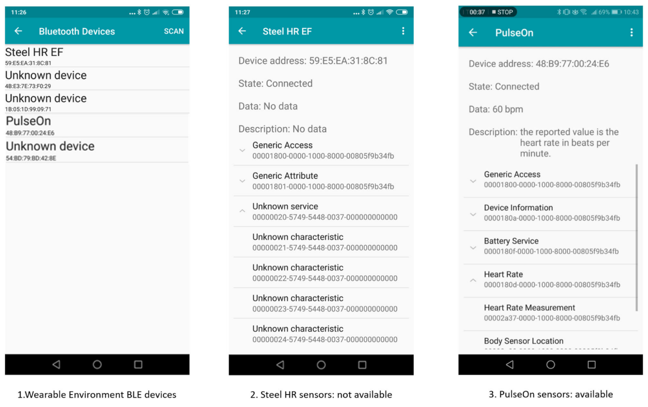
As an example of the applicability of WEAR-IT (and, more extensively, wearable environment) features in IoT research, the mHealth domain has been presented. In particular, the adoption of this middleware has allowed to improve the performance of the round table abstraction [57], a virtual community composed of many different stakeholders involved in chronic disease management. As pointed out in [58], online healthcare communities provide individuals with platforms to acquire information from peers with similar illness. The most important feature of the wearable environment in supporting this kind of community is the possibility to provide the stakeholders with different views on the same data, according to their role in the disease management. Indeed, interoperability among heterogeneous data sources and decision-making processes in virtual communities operating in healthcare sectors is an important challenge for standardization in the next future. Recent research (see, e.g., [59]) witnesses that quality of care services is not positively influenced by all kinds of integration practices; one of the goal of future research is to study if and how the round table abstraction supported by the wearable environment notion can be situated in this challenging debate.

In particular, Saref4Helth [60] is a recent extension of the SAREF ontology for cardiac activity monitoring that presents many similarities with the wearable environment used in the MoveUp case study. A *time-series measurement* extending the SAREF *measurement* concept has been introduced to manage ECG sequences. Different from that, WEAR-IT manages an ECG sequence as a *collection of values detected by a heart rate sensor, sampled at a given frequency and for a given period of time*, according to the application needs.

VIII. Conclusion

In this article, we proposed the *wearable environment* notion and architecture as a mean to cast into a unique conceptual and computational framework data acquisition from sensors, data representation through wearable devices, and their use by means of WESs. The heart of this notion was the data acquisition and representation layer, where the WEAR-IT platform was designed and implemented to build an efficient, reliable, and scalable bridge between raw data and applications.

Ongoing works are devoted to improving the interconnection between applications at WES level. One of the possible directions is shown in Fig. 13, where two distinct BLE devices, namely, $PulseOn$ and $NokiaSteelHR$ smartwatches, are included in the wearable environment definition. If the $NokiaSteelHR$ is selected to query its sensors, the WEAR-IT framework is not able to discover available data sources.


11:26
Bluetooth Devices SCAN
Steel HR EF
59:E5:EA:31:8C:81
Unknown device
4B:E3:7E:73:F0:29
Unknown device
1B:05:1D:99:09:71
PulseOn
48:B9:77:00:24:E6
Unknown device
54:BD:79:BD:42:8E

1.Wearable Environment BLE devices

11:27
Steel HR EF
Device address: 59:E5:EA:31:8C:81
State: Connected
Data: No data
Description: No data
Generic Access
00001800-0000-1000-8000-00805f9b34fb
Generic Attribute
00001801-0000-1000-8000-00805f9b34fb
Unknown service
00000020-5749-5448-0037-000000000000
Unknown characteristic
00000021-5749-5448-0037-000000000000
Unknown characteristic
00000022-5749-5448-0037-000000000000
Unknown characteristic
00000023-5749-5448-0037-000000000000
Unknown characteristic
00000024-5749-5448-0037-000000000000

2. Steel HR sensors: not available

00:37 STOP 69% 10:43
PulseOn
Device address: 48:B9:77:00:24:E6
State: Connected
Data: 60 bpm
Description: the reported value is the heart rate in beats per minute.
Generic Access
00001800-0000-1000-8000-00805f9b34fb
Device Information
0000180a-0000-1000-8000-00805f9b34fb
Battery Service
0000180f-0000-1000-8000-00805f9b34fb
Heart Rate
0000180d-0000-1000-8000-00805f9b34fb
Heart Rate Measurement
00002a37-0000-1000-8000-00805f9b34fb
Body Sensor Location

3. PulseOn sensors: available

Fig. 13. BLE device comparison in WEAR-IT.

This means that BLE configuration of the device is not standard; thus, no data can be delivered to a WES. Sensors of $NokiaSteelHR$ can only be interrogated through its official apps such as $HealthMate$; the possibility to connect a proprietary app like Health Mate to a middleware like WEAR-IT, in order to overcome the problem described so far, could open new possibilities of developing innovative application in mobile domains; to this aim, we are working on extending the APIs set presented in Section IV to allow an application a_i "playing" another application a_j:

$$\text{API}_{a_{i_{a_j}}} = \{play(a_j), stop(a_j), store(a_j)\}.$$

Another ongoing activity stems from the fact that data reliability is one of the most important factors for WES development. As a consequence, given that, in a wearable environment, Kaa acts as an intermediate data collector, it is important to ensure that the *quality* of data stored by it for use by applications at the reasoning level is high enough. To this aim, we are currently addressing our research toward the reliability of data sent to Kaa and the quality of data stored. As a first step in this direction, we have recently developed a promising algorithm based on the use of spatial and temporal information [61] for the correction of errored and missing data.

Finally, a future extension of this article concerns the definition of an extended wearable environment infrastructure, where many users, each one equipped with a *personal wearable environment*, could be asked to join an *integrated*, *geographically distributed*, and *scalable* wearable environment. In this environment, an application could query end-points and acquire data from them according to their position and reachability from mobile nodes.

In this scenario, the most interesting challenges according to [33] are the following.

1) *Contact detection:* Since communication is possible only when the nodes are efficiently reachable, it is necessary to detect the presence of a mobile node quickly and correctly. This is especially critical when the duration of contacts is short.
2) *Mobility-aware power management:* In some cases, it is possible to exploit the knowledge on the mobility pattern

to further optimize the data transmission costs. In fact, if visiting times are known or can be predicted with a certain accuracy, sensor nodes can be awake only when they expect the mobile element to be in their transmission range.

3) *Reliable data transfer:* As available contacts might be scarce and short, there is a need to maximize the number of messages correctly transferred to the sink, by choosing the protocol and communication strategy best suited to the situation.

References

[1] E. Folmer and K. Jakobs, "Standards development for smart systems—A potential way forward," *IEEE Trans. Eng. Manage.*, vol. 68, no. 1, pp. 75–86, Feb. 2020.

[2] P. Suresh, J. V. Daniel, V. Parthasarathy, and R. H. Aswathy, "A state of the art review on the Internet of things (IoT) history, technology and fields of deployment," in *Proc. Int. Conf. Sci. Eng. Manage. Res.*, 2014, pp. 1–8.

[3] EPoSS, "Strategic research agenda of the European technology platform on smart systems," 2017. [Online]. Available: https://www.smart-systems-integration.org/publication/eposs-sra-2017

[4] F. Sartori and R. Melen, "Wearable expert system development: Definitions, models and challenges for the future," *Program*, vol. 51, no. 3, pp. 235–258, 2017.

[5] S. Kumar, P. B. Southard, and M. White, "Telemedicine: Determining "critical to quality" characteristics for a healthcare service system design based on a survey of physical rehabilitation providers," *IEEE Eng. Manage. Rev.*, vol. 44, no. 2, pp. 41–55, Apr.–Jun. 2016.

[6] G. Appelboom *et al.*, "Smart wearable body sensors for patient self-assessment and monitoring," *Arch. Public Health*, vol. 72, no. 1, 2014, Art. no. 28.

[7] J. Casselman, N. Onopa, and L. Khansa, "Wearable healthcare: Lessons from the past and a peek into the future," *Telemat. Inform.*, vol. 34, no. 7, pp. 1011–1023, 2017.

[8] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surv. Tut.*, vol. 17, no. 4, pp. 2347–2376, Oct.–Dec. 2015.

[9] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[10] J. Kim *et al.*, "Standard-based IoT platforms interworking: Implementation, experiences, and lessons learned," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 48–54, Jul. 2016.

[11] D. Zeng, S. Guo, and Z. Cheng, "The web of things: A survey," *J. Commun.*, vol. 6, no. 6, pp. 424–438, 2011.

[12] B. P. Wong and B. Kerkez, "Real-time environmental sensor data: An application to water quality using web services," *Environ. Modelling Softw.*, vol. 84, pp. 505–517, 2016.

[13] M. U. H. Al Rasyid, A. Sayfudin, A. Basofi, and A. Sudarsono, "Development of semantic sensor web for monitoring environment conditions," in *Proc. Int. Seminar Intell. Technol. Appl.*, 2016, pp. 607–612.

[14] A. Bröring, S. K. Datta, and C. Bonnet, "A categorization of discovery technologies for the Internet of things," in *Proc. 6th Int. Conf. Internet Things*, 2016, pp. 131–139.

[15] S. C. Mukhopadhyay, "Wearable sensors for human activity monitoring: A review," *IEEE Sens. J.*, vol. 15, no. 3, pp. 1321–1330, Mar. 2015.

[16] J. Edwards, "Wireless sensors relay medical insight to patients and caregivers [*special reports*]," *IEEE Signal Process. Mag.*, vol. 29, no. 3, pp. 8–12, May 2012.

[17] P. A. Shaltis, A. T. Reisner, and H. H. Asada, "Cuffless blood pressure monitoring using hydrostatic pressure changes," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 6, pp. 1775–1777, Jun. 2008.

[18] M.-Z. Poh, K. Kim, A. Goessling, N. Swenson, and R. Picard, "Cardiovascular monitoring using earphones and a mobile device," *IEEE Pervasive Comput.*, vol. 11, no. 4, pp. 18–26, Oct.–Dec. 2012.

[19] P. Salvo, F. Di Francesco, D. Costanzo, C. Ferrari, M. G. Trivella, and D. De Rossi, "A wearable sensor for measuring sweat rate," *IEEE Sens. J.*, vol. 10, no. 10, pp. 1557–1558, Oct. 2010.

[20] C. Strohrmann, H. Harms, C. Kappeler-Setz, and G. Troster, "Monitoring kinematic changes with fatigue in running using body-worn sensors," *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 5, pp. 983–990, Sep. 2012.

[21] M. Ermes, J. Pärkkä, J. Mäntyjärvi, and I. Korhonen, "Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions," *IEEE Trans. Inf. Technol. Biomed.*, vol. 12, no. 1, pp. 20–26, Jan. 2008.

[22] M. M. Rodgers, V. M. Pai, and R. S. Conroy, "Recent advances in wearable sensors for health monitoring," *IEEE Sens. J.*, vol. 15, no. 6, pp. 3119–3126, Jun. 2015.

[23] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2688–2710, 2010.

[24] S. Movassaghi, M. Abolhasan, J. Lipman, D. Smith, and A. Jamalipour, "Wireless body area networks: A survey," *IEEE Commun. Surv. Tut.*, vol. 16, no. 3, pp. 1658–1686, Jul.-Sep. 2014.

[25] M. J. Mathie, A. C. Coster, N. H. Lovell, B. G. Celler, S. R. Lord, and A. Tiedemann, "A pilot study of long-term monitoring of human movements in the home using accelerometry," *J. Telemed. Telecare*, vol. 10, no. 3, pp. 144–151, 2004.

[26] E. S. Sazonov, G. Fulk, N. Sazonova, and S. Schuckers, "Automatic recognition of postures and activities in stroke patients," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, 2009, pp. 2200–2203.

[27] D. Giansanti, G. Maccioni, and S. Morelli, "An experience of health technology assessment in new models of care for subjects with Parkinson's disease by means of a new wearable device," *Telemed. e-Health*, vol. 14, no. 5, pp. 467–472, 2008.

[28] O. Aziz *et al.*, "A pervasive body sensor network for measuring postoperative recovery at home," *Surg. Innov.*, vol. 14, no. 2, pp. 83–90, 2007.

[29] M. S. Patel, D. A. Asch, and K. G. Volpp, "Wearable devices as facilitators, not drivers, of health behavior change," *JAMA*, vol. 313, no. 5, pp. 459–460, 2015.

[30] O. Amft and G. Tröster, "Recognition of dietary activity events using on-body sensors," *Artif. Intell. Med.*, vol. 42, no. 2, pp. 121–136, 2008.

[31] M. G. Benedetti *et al.*, "Physical activity monitoring in obese people in the real life environment," *J. Neuroeng. Rehabil.*, vol. 6, no. 1, 2009, Art. no. 47.

[32] A. Sciacqua *et al.*, "Validation of a flexible and innovative platform for the home monitoring of heart failure patients: Preliminary results," in *Proc. 36th Annu. Comput. Cardiol. Conf.*, 2009, pp. 97–100.

[33] M. Di Francesco, S. K. Das, and G. Anastasi, "Data collection in wireless sensor networks with mobile elements: A survey," *ACM Trans. Sens. Netw.*, vol. 8, no. 1, pp. 1–31, 2011.

[34] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 7, no. 3, pp. 537–568, 2009.

[35] K. B. Korb and A. E. Nicholson, *Bayesian Artificial Intelligence*. Boca Raton, FL, USA: CRC Press, 2010.

[36] F. Sartori and R. Melen, "Time evolving expert systems design and implementation: The KAFKA approach," in *Proc. Int. Conf. Knowl. Eng. Ontol. Develop. (Part 7th Int. Joint Conf. Knowl. Discov., Knowl. Eng. Knowl. Manage.)*, Lisbon, Portugal, Nov. 2015, pp. 84–95.

[37] R. Melen, F. Sartori, and L. Grazioli, "Modeling and understanding time-evolving scenarios," in *Proc. 19th World Multi-Conf. Syst., Cybern. Inform.*, 2015, vol. 1, pp. 267–272.

[38] T. Gindele, S. Brechtel, and R. Dillmann, "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments," in *Proc. 13th Int. IEEE Conf. Intell. Transp. Syst.*, 2010, pp. 1625–1631.

[39] C. Carmona, G. Castillo, and E. Millán, "Designing a dynamic Bayesian network for modeling students' learning styles," in *Proc. 8th IEEE Int. Conf. Adv. Learn. Technol.*, 2008, pp. 346–350.

[40] X. Li and Q. Ji, "Active affective state detection and user assistance with dynamic Bayesian networks," *IEEE Trans. Syst., Man, Cybern. A: Syst. Humans*, vol. 35, no. 1, pp. 93–105, Jan. 2005.

[41] P. C. Hallal, A. E. Bauman, G. W. Heath, H. W. Kohl, I.-M. Lee, and M. Pratt, "Physical activity: More of the same is not enough," *Lancet*, vol. 380, no. 9838, pp. 190–191, 2012.

[42] V. S. Conn, A. R. Hafdahl, and D. R. Mehr, "Interventions to increase physical activity among healthy adults: Meta-analysis of outcomes," *Amer. J. Public Health*, vol. 101, no. 4, pp. 751–758, 2011.

[43] T. Armstrong and R. Bonita, "Capacity building for an integrated noncommunicable disease risk factor surveillance system in developing countries." *Ethnicity Disease*, vol. 13, no. 2, pp. S13–S18, 2002.

[44] A. Bandura, "Self-efficacy: Toward a unifying theory of behavioral change," *Psychol. Rev.*, vol. 84, no. 2, pp. 191–215, 1977.

[45] E. A. Locke and G. P. Latham, "Building a practically useful theory of goal setting and task motivation: A 35-year odyssey," *Amer. Psychol.*, vol. 57, no. 9, pp. 705–717, 2002.
[46] D. Baretta *et al.*, "Wearable devices and AI techniques integration to promote physical activity," in *Proc. 18th Int. Conf. Human-Comput. Interact. Mobile Devices Serv. Adjunct*, 2016, pp. 1105–1108.
[47] J.-Y. Ho and E. O'Sullivan, "Strategic standardisation of smart systems: A roadmapping process in support of innovation," *Technol. Forecast. Soc. Change*, vol. 115, pp. 301–312, 2017.
[48] Q. Wang, C. Voss, and X. Zhao, "Deployment strategies for service innovation," *IEEE Trans. Eng. Manage.*, vol. 66, no. 4, pp. 514–528, Nov. 2018.
[49] L. Daniele, F. den Hartog, and J. Roes, "Created in close interaction with the industry: The smart appliances reference (SAREF) ontology," in *Proc. Int. Workshop Formal Ontol. Meet Ind.*, 2015, pp. 100–112.
[50] A. Basaure, A. Vesselkov, and J. Töyli, "Internet of Things (IoT) platform competition: Consumer switching versus provider multihoming," *Technovation*, vol. 90, 2020, Art. no. 102101.
[51] C. Y. Baldwinand C. J. Woodard, "The architecture of platforms: A unified view," *Platforms, Markets Innov.*, vol. 32, pp. 19–35, 2009.
[52] D. P. McIntyre and A. Srinivasan, "Networks, platforms, and strategy: Emerging views and next steps," *Strategic Manage. J.*, vol. 38, no. 1, pp. 141–160, 2017.
[53] M. G. Jacobides, C. Cennamo, and A. Gawer, "Towards a theory of ecosystems," *Strategic Manage. J.*, vol. 39, no. 8, pp. 2255–2276, 2018.
[54] S. K. Fixson, D. Khachatryan, and W. Lee, "Technological uncertainty and firm boundaries: The moderating effect of knowledge modularity," *IEEE Trans. Eng. Manage.*, vol. 64, no. 1, pp. 16–28, Feb. 2017.
[55] J.-C. Rochet and J. Tirole, "Platform competition in two-sided markets," *J. Eur. Econ. Assoc.*, vol. 1, no. 4, pp. 990–1029, 2003.
[56] F. Zhu and M. Iansiti, "Entry into platform-based markets," *Strategic Manage. J.*, vol. 33, no. 1, pp. 88–106, 2012.
[57] F. Sartori, R. Melen, M. Lombardi, and D. Maggiotto, "Virtual round table knights for the treatment of chronic diseases," *J. Rel. Intell. Environ.*, vol. 5, no. 3, pp. 131–143, 2019.
[58] N. Liu, Y. Tong, and H. C. Chan, "Information seeking in online healthcare communities: The dual influence from social self and personal self," *IEEE Trans. Eng. Manage.*, vol. 64, no. 4, pp. 529–538, Nov. 2017.
[59] W. Glover, Q. Li, E. Naveh, and M. Gross, "Improving quality of care through integration in a hospital setting: A human systems integration approach," *IEEE Trans. Eng. Manage.*, vol. 64, no. 3, pp. 365–376, Aug. 2017.
[60] J. Moreira, L. F. Pires, M. van Sinderen, L. Daniele, and M. Girod-Genet, "SAREF4health: Towards IoT standard-based ontology-driven cardiac e-health systems," *Appl. Ontol.*, vol. 15, no. 3, pp. 385–410, 2020.
[61] F. R. Sartori, R. Melen, and F. Giudici, "IoT data validation using spatial and temporal correlations," in *Metadata and Semantic Research*, E. F. Garoufallou Fallucchi, and E. W. D. Luca, Eds., vol. 1057. New York, NY, USA: Springer, 2019, pp. 77–89.

Fabio Sartori received the graduate and Ph.D. degrees in computer science from the University of Milano-Bicocca, Milan, Italy, in 2000 and 2005, respectively.

Since 2008, he has been an Assistant Professor of programming languages with the Department of Informatics, Systems and Communication, University of Milano-Bicocca. He is the author of more than 70 papers published in international conference proceedings or journals. He has taken care of both methodological and computational aspects in the development of decision support systems, exploiting wearable devices as information sources in the context of his research interests, which mainly include knowledge-based systems and knowledge management.

Riccardo Melen received the graduate degree in electronic engineering from the Polithecnic of Turin, Turin, Italy, in 1981.

Since 2000, he has been a Full Professor of telecommunication networks with the Department of Informatics, Systems and Communication, University of Milano-Bicocca, Milan, Italy. He is the author of more than 80 papers published in international journals or conference proceedings. His research interests include mobile applications and services, event management, and security and authentication systems.