Engineering Applications of Artificial Intelligence 67 (2018) 136–156

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

A smart mobile, self-configuring, context-aware architecture for personal health monitoring

Massimo Esposito [a,*], Aniello Minutolo [a], Rosario Megna [b], Manolo Forastiere [c], Mario Magliulo [b], Giuseppe De Pietro [a]

[a] National Research Council of Italy — Institute for High Performance Computing and Networking (ICAR), Via P. Castellino 111, 80131 Naples, Italy
[b] National Research Council of Italy — Institute of Biostructures and Bioimaging (IBB), Via Tommaso De Amicis 95, 80145 Naples, Italy
[c] Neatec S.p.A., Via Campi Flegrei, 34, 80078 Pozzuoli, Italy

ARTICLE INFO

Keywords:
Mobile health monitoring
Software architecture
Smart applications
Context-awareness
Reasoning
Ontologies

ABSTRACT

The last decade has witnessed an exponential increase in older adult population suffering from chronic life-long diseases and needing healthcare. This situation has highlighted a need to revolutionize healthcare and provide innovative, efficient, and affordable solutions to patients at any time and from anywhere in an economic and friendly manner. The recent developments in sensing, mobile, and embedded devices have attracted considerable attention toward mobile health monitoring applications. However, existing architectures aimed at facilitating the realization of these mobile applications have shown to be not suitable to address all these challenging issues: (i) the seamless integration of heterogeneous devices; (ii) the estimation of vital parameters not measurable directly or measurable with a low accuracy; (iii) the extraction of context information pertaining to the patient's activity to be used for the interpretation of vital parameters; (iv) the correlation of physiological and contextual information to detect suspicious anomalies and supply alerts; (v) the notification of anomalies to doctors and caregivers only when their detection is accurate and appropriate. In light of the above, this paper presents a smart mobile, self-configuring, context-aware architecture devised to enable the rapid prototyping of personal health monitoring applications for different scenarios, by exploiting commercial wearable sensors and mobile devices as well as knowledge-based technologies. This architecture is organized as a composition of four tiers that operate on a layered fashion and it exploits an ontology-based data model to ensure intercommunication among these tiers and the monitoring applications built on the top of them. The proposed architecture has been implemented for mobile devices equipped with the Android platform and evaluated with respect to its modifiability by employing the ALMA (Architecture Level Modifiability Analysis) method, highlighting its capability of being rapidly customized, personalized or eventually modified by software developers in order to prototype, with a reduced effort, novel health monitoring applications on the top of its components. Finally, it has been employed to build, as case study, a mobile application aimed at monitoring and managing cardiac arrhythmias, such as bradycardia and tachycardia, confirming its effectiveness with respect to a real scenario.

1. Introduction

The last decade has witnessed an exponential increase in older adult population suffering from chronic life-long diseases and needing healthcare. Indeed, the number of patients requiring health care services has raised proportionally with the growth in population, reaching approximately 2 billion by 2050, with 80% in developing countries (WHO, 2002). Therefore, it is predicted that the cost of hospitalization and patient care will rise worldwide and patients will find increasing difficulties to receive necessary treatments and assistance even in emergency situations. In light of the above, it is clear that there exists a need to revolutionize healthcare and provide innovative, efficient, and affordable solutions to patients at any time and from anywhere in an economic and friendly manner.

In such a direction, recently, a patient-oriented model is being considered, where patients are being equipped with knowledge and technologies to play a more active role in his/her health monitoring. This health monitoring can be rigorously defined as "repeated or

* Corresponding author.
E-mail address: massimo.esposito@icar.cnr.it (M. Esposito).

https://doi.org/10.1016/j.engappai.2017.09.019
Received 15 May 2017; Received in revised form 3 August 2017; Accepted 19 September 2017
Available online 17 October 2017
0952-1976/

continuous observations or measurements of the patient, his or her physiological function, and the function of life support equipment, for the purpose of guiding management decisions, including when to make therapeutic interventions, and assessment of those interventions" (Hudson, 1985). This model embraces the principles of proactivity, independence, accessibility, and cost-effectiveness by using a wide range of mobile technologies such as smartphones, tablets, and wearable sensors for the continuous monitoring of patients' behavior and vital signs (Banos et al., 2014).

This allows for the possibility of an increased focus on individual health promotion and health maintenance rather than the traditional focus on dealing with the consequences of illness and injury after they occur. Such a transformation could have a profound impact on the cost of providing healthcare and on the level of health enjoyed by individuals during their lifetimes (Giffen et al., 2015).

This new trend of mobile health monitoring applications has been made possible due to a fabulous development in mobile devices and wearable sensors alongside wireless and cellular communication networks (Serhani et al., 2016). Existing applications of this typology essentially utilize wearable sensors to continuously monitor different physiological parameters of the patient. The observed information is sent to a hub, which collects the measurements and sends them, through communication networks, to the final destinations, i.e. the healthcare professionals or doctors. These latter observe the current medical condition and activities of the patient and provide a diagnosis or assistance based on the information provided by the applications. They can also get in touch with the patient through either the reversed channel or traditional communication media, such as the telephone or the SMS system (Serhani et al., 2016). The hub can be represented by the cloud, where sensed information is sent for being analyzed and accessed by the healthcare professionals or doctors (Forkan et al., 2014). Alternatively, it can be an intermediate personal computer or a mobile device, which in turn relays the received information to the healthcare professionals or doctors. Both these solutions are often adopted due to one of the main limitations of up-to-date wearable sensors, i.e. their processing and storage capabilities (Banos et al., 2014). Indeed, they are able to measure physiological magnitudes and convert them into machine-readable information, but they are equipped with very limited resources to process this information.

However, currently, a set of issues hampers both innovation and development of new mobile health monitoring applications.

Firstly, the heterogeneity of communication protocols and the mixture of addressing schemes used by wearable sensors of different vendors and models is challenging to be handled when developing integrated mobile health monitoring applications (Evensen and Meling, 2009). Most of the applications offered today is based on proprietary all-in-one solutions, where sensors might use either a proprietary RF protocol over the 868MHz band or ZigBee, Bluetooth, WiFi or another IEEE 802.x-based protocol over the 2.4 GHz band. Furthermore, many sensors have their own application-level protocol for communicating control commands and retrieving data. This implies that applications need to know anything about the physical or logical communication protocols used by each specific sensor, and that they can become unusable when a different sensor model is used even if they share the same basic functionalities.

Secondly, due to the miniaturization of electronic devices and the development of new ways of mobile computing in recent years, wearable technology has seen substantial advances, mainly triggered by the need for non-invasive, non-obtrusive ways to monitor physiological signals over long periods of time (Marques et al., 2011).

However, some vital parameters cannot be measured precisely, easily and noninvasively. For instance, today, systolic and diastolic blood pressures can be measured by means of digital monitors able to communicate their readings to smartphones via a wireless Bluetooth connectivity. The most accurate monitors feature an upper arm cuff, which is extremely invasive to be used to perform a continuous health monitoring. On the other hand, other monitors make use of non-invasive devices placed on the wrist or on the finger, and thus are characterized by a higher degree of freedom and easiness of use, but they generate less accurate blood pressure readings.

Thirdly, physiological parameters alone are not enough to characterize the patients' health status. For instance, an increase in the heart rate is considered abnormal when evaluated singularly, whereas it is classified as normal when it is registered while the patient is running. As a result, context information, and, in particular, information pertaining to the patient's activity, should be used as an additional factor in the interpretation of the vital signs. However, context information coming from sensors is often characterized by a lack of precision and accuracy as well as a fine granularity that make it difficult to use at the application level (Clear et al., 2007). Monitoring applications respond to events that should be generally richer than a single sensor reading and sensitive to user activities also affected by uncertainty. For example, the activity "running" is richer, and of more use than, the accelerometer motion values over the x, y and z axes, but it cannot be defined precisely, for instance, in terms of number of steps.

Fourthly, the capability of correlating physiological and contextual information, detecting suspicious anomalies and supplying warnings or suggestions to enable personalized monitoring and health management is undoubtedly a real-added value for monitoring applications. Embedding intelligent components able to reason over mobile devices and locally perform an accurate and continuous analysis of the patient's health status allows minimizing network transmission with remote stations, avoiding communication delays or interruptions and maintaining appropriate levels of security and privacy (Minutolo et al., 2015). However, the capability to reason over different forms of information and knowledge, often graded and affected by uncertainty, directly on the mobile devices represents a critical point, still pending to date.

Fifthly, notifying anomaly conditions to the healthcare professionals or doctors plays an effective role in enabling personalized monitoring and health management only in case when it is accurate and appropriate. However, notifications or alerts can be generated inappropriately, especially in cases when anomaly conditions are detected without taking into account that physiological data may vary for each patient. In these situations, they become ineffective and bothersome, leading to alert fatigue, a state in which the healthcare professionals or doctors become less responsive to them in general.

In literature, various solutions, architectures, and frameworks have been proposed for personal health monitoring, but they are not completely adequate to address all the above-mentioned issues. Starting from this consideration, in this paper, a smart mobile, self-configuring, context-aware architecture for personal health monitoring is proposed. This architecture enables the rapid prototyping of personal health monitoring applications for different scenarios, by exploiting commercial wearable sensors and mobile devices as well as knowledge-based technologies, in accordance with the following functional model. Sensor data are collected by wearable sensors and transmitted through a wireless communication network, stored and analyzed on mobile phones by exploiting knowledge-based formalisms and technologies, and finally sent from mobile devices to the healthcare professionals or doctors for subsequent storage and processing. A proper action depending on the patient's condition is determined and communicated to the healthcare professionals or doctors through mobile communication channels.

The word "*smart*" refers to its capacity to be adaptive, configurable, dynamic, and reactive, which accordingly makes it able not only to provide information about physiological condition but also personalized assistance for each patient in terms of notifications when he/she may be at risk. As the term "*self-configuring*" advocates, it is not based on a specific kind of sensor, but different and heterogeneous sensors can be used, offering uniform communication interfaces to external applications. The word "*mobile*" refers to its capability of using portable and wireless sensors as well as embedding data storage and data processing completely inside a mobile device, without requiring the usage

of centralized computing platforms located in clouds and accessible over the wireless connection. Finally, "*context-aware*" stands for the ability of mainly understanding and handling context and activities that can be sensed automatically with reference to an individual and treated as implicit input to positively affect the behavior of a monitoring application.

The proposed architecture has been implemented for mobile devices equipped with the Android platform and evaluated by employing the ALMA method (Architecture Level Modifiability Analysis) (Bengtsson et al., 2004), with respect to its modifiability, i.e. its ability to be simply modified and evolve over time.

Moreover, since it is extremely general and can allow anywhere and anytime monitoring of the health status of a patient, several application areas can benefit from its facilities. In particular, as case study, it has been employed in the context of the Italian project "Bersagli" to build a mobile application aimed at monitoring and managing cardiac arrhythmias, such as bradycardia and tachycardia.

The remainder of the paper is structured as follows. Section 2 reviews and compares existing health monitoring architectures and systems, detailing their characteristics and presenting the fundamental contributions of this proposal. Section 3 presents the proposed architecture for the rapid prototyping of mobile health monitoring applications. Section 4 outlines the implementation details of the architecture and describes the modifiability analysis performed by using the ALMA method. The mobile application for monitoring cardiac arrhythmias built on the top of the architecture is diffusely depicted in Section 5. Finally, Section 6 concludes the work and introduces some future activities.

2. Related work

Extensive efforts have been made in both academia and industry in the research and development of various solutions, architectures, and frameworks for health monitoring (Chan et al., 2012; Patel et al., 2012).

In more detail, a framework is described in Mei et al. (2006) for creating different vital sign representations based on XML schema diagrams. It can cope flexibly with the constraints and requirements raised by different stakeholders. This flexibility can be also exploited for realizing adaptive or context-aware systems to overcome infrastructural changes due to patient's mobility in heterogeneous environments.

In the MobiHealth project (Jones et al., 2006), a generic platform integrating the technologies of Body Area Networks (BANs), wireless broadband communications and wearable medical devices is realized, in order to provide mobile healthcare services for patients and health professionals. These technologies enable remote patient care services, such as management of chronic conditions and detection of health emergencies. It has been trialled and evaluated in four European countries with a variety of patient groups, focusing on home care, trauma care and outdoor settings. Other projects worth mentioning that have been carried out as part of different programs of the European Commission are: MyHeart (Habetha, 2006), WEALTHY (Paradiso et al., 2004) and MagIC (Rienzo et al., 2006). These projects led to the development of garment-based wearable sensors aiming at general health monitoring of people in the home and community settings.

Another architecture is presented in Roy et al. (2007), which is aimed at efficiently supporting data fusion and user-centric situation prediction based on dynamic Bayesian networks, leading to context-aware healthcare applications in smart environments. This framework provides a systematic approach to fuse context fragments and deal with context ambiguity in a probabilistic manner, to represent context within the application, and to easily compose rules to reason efficiently to mediate ambiguous contexts. This architecture has been evaluated to monitor elderly people in small home environments.

The ElderCare platform is described in López-de Ipiña et al. (2011), which is aimed at offering an Ambient Assisted Living solution designed to give support not only to people in risk of losing autonomy but also to caretakers or people concerned about them (relatives or friends). It is a low cost, easily deployable, usable and evolvable ICT infrastructure, which combines common hardware, OSGi middleware and mobile-aided care data management and uses an interactive TV as main interaction mechanism offered.

A Personal Health Monitor system is proposed in Gay and Leijdekkers (2007), where a patient is monitored using various types of off-the-shelf sensors (ECG, accelerometer, oximeter, weight scale, blood pressure monitor). Healthcare professionals can select one or more sensors to be used for a particular patient in order to provide personalized monitoring and treatment. Sensor data are collected and transferred wirelessly to a smartphone, where they are locally processed and analyzed. Mechanisms are given for locating the patient position in case of emergency, detecting life-threatening anomalies and giving general information about the patient's health when he/she is not in a dangerous situation.

In Agarwal et al. (2013), a web-based architecture is proposed for the monitoring of patient vital statistics in a local environment using wireless medical sensor devices and a smart-phone as a local hub. It is devised to provide an available, reliable, secure, and dependable solution for a mobile platform based system with the aim of mitigating costs on individual patient care while maintaining a high-quality level of care.

Zappa is presented in Ruiz-Zafra et al. (2013), which is an extensible, scalable, highly-interoperable and customizable platform, designed to support e-Health/m-Health systems and that is able to operate in the cloud. The platform architecture is based on components and services, as well as on open source software that reduces its acquisition and operation costs. The platform can be used to develop several remote mobile monitoring m-health systems. More recently, a framework is proposed in Khalifeh et al. (2014), which provides scalable and effective monitoring services for patients' health and conditions to support them to rehabilitate and recover from their illnesses. It utilizes a systems-of-systems approach in sensing, analyzing the patients' information, and issuing real-time alert messages to the healthcare monitoring center. It also proposes mechanisms to efficiently deal with the reliability and real-time data delivery challenges that are very essential in healthcare monitoring and reporting process. A mobile monitoring application is outlined in Villarreal et al. (2014), aimed at allowing a patient to monitor a chronic disease through mobile and biometrics devices. It consists into a set of layers distributed in three elements that interact with the final application: mobile devices, biometrics devices, and central server. This distribution is performed to identify each element that interacts with the application, to identify where the application needs change and to add new elements to the application. It uses MobiPattern to define the interface, layers to distribute the application to the mobile devices and server, and ontologies to classify medical elements, such as diseases, recommendations, preventions, foods, mobile devices and diet suggestions.

An advanced multi-sensor platform is presented in Fanucci et al. (2015), which integrates all actors involved in the healthcare of chronic patients, for the provisioning of personalized telemedicine services based on the home monitoring of vital signs and the circulation of the clinical information among all the caregivers. This platform consists into three main elements: (i) a service center server that acts as a central hub for the data and information exchange; (ii) a multi-sensor tele-monitoring system, usually installed at patient's home, aimed at acquiring and transmitting vital signs to a service center; (iii) the general practitioner module that allows the family doctor to interact with the clinical information of the followed patients, realizing at distance the evolution of the disease.

In Lamprinakos et al. (2015), another platform is designed and implemented, which enables the deployment of services to follow-up the patient's health status based on a set of monitored parameters per disease, and to profile user's habits and diagnose deviations from their usual activities. A key aspect of the platform is its SOA (Service Oriented Architecture) middleware that collects data from heterogeneous Telecare and Telehealth gateways, and provides the upper service

Table 1
Comparison of different mobile solutions, architectures, and frameworks for health monitoring.

Related works	Self-configuration	Non-invasive parameter estimation	Context-awareness	Hybrid knowledge representation and reasoning	Personalization	Mobility
Mei et al. (2006)	Yes	No	No	No	No	Yes
Jones et al. (2006)	Yes	Yes	No	No	Yes	Yes
Habetha (2006)	Yes	No	No	No	Yes	Yes
Paradiso et al. (2004)	No	No	No	No	Yes	Yes
Rienzo et al. (2006)	No	No	Yes	No	Yes	Yes
Roy et al. (2007)	No	No	Yes	Yes	Yes	Yes
López-de Ipiña et al. (2011)	Yes	No	No	No	Yes	No
Gay and Leijdekkers (2007)	Yes	No	Yes	No	Yes	Yes
Agarwal et al. (2013)	Yes	No	No	No	Yes	Yes
Ruiz-Zafra et al. (2013)	Yes	No	No	No	Yes	Yes
Khalifeh et al. (2014)	Yes	No	No	No	Yes	Yes
Villarreal et al. (2014)	Yes	No	No	Yes	Yes	Yes
Fanucci et al. (2015)	Yes	No	Yes	No	Yes	Yes
Lamprinakos et al. (2015)	Yes	No	Yes	Yes	Yes	No
Serhani et al. (2016)	Yes	No	No	No	Yes	Yes
The proposed architecture	Yes	Yes	Yes	Yes	Yes	Yes

Fig. 1. The functional model of monitoring applications that can be built on the basis of proposed architecture.

layers with a unified and standards compliant message. In this way, an integrated view of Telehealth and Telecare data and alerts is made possible into a backend Web Portal, accessible by both clinicians and operators.

Finally, SME2EM is outlined in Serhani et al. (2016), which is a novel mobile-based end-to-end architecture for live monitoring and visualization of life-long diseases, offering smartness features to cope with continuous monitoring, data explosion, dynamic adaptation, unlimited mobility, and constrained devices resources. Its components are fully implemented as Web services in accordance with the SOA paradigm to be easy to deploy and integrate, and are supported by Cloud infrastructure to allow high scalability, availability of processes and data being stored and exchanged. The architecture system is formally model-checked to automatically verify its correctness against designers' desirable properties at design time. Its applicability is evaluated through concrete experimental scenarios on monitoring and visualizing states of epileptic diseases.

Summarizing, all the solutions above-mentioned are relevant and technically valid with respect to different application domains, by offering many interesting features. Nonetheless, they are still far from a productive framework to build real monitoring applications usable in daily practice. Indeed, to the best of our knowledge, the existing solutions are not completely adequate to address all the open issues outlined in Section 1, as synthetically reported in Table 1.

In other words, they do not exhibit all the functionalities necessary to simplify the development of health monitoring applications, with the characteristic of being expressly devised for running entirely on mobile devices and efficiently exploiting their available resources. As a consequence, the gap between the technological reality in terms of enabling frameworks or architectures and the mobile monitoring application requirements may be considered the most limiting factor for a widespread deployment of these solutions in daily scenarios.

On the contrary, the proposed architecture exhibits its novelty in the way that it provides an extensive and generic set of features to concretely enable the realization of mobile monitoring applications for a variety of possible scenarios. All these characteristics together with the specific architectural components in charge of offering them to build monitoring applications are diffusely described in the next section.

3. The proposed architecture

The proposed architecture has been designed for being deployed on mobile devices, offering functionalities for collecting, processing and storing information about patients' health conditions in order to support the process of identification of possible abnormal events.

Monitoring applications, which can be built based on this architecture, are characterized by the functional model visualized in Fig. 1.

A patient is monitored in order to acquire and evaluate his/her physiological conditions. A number of biomedical, low cost and non-invasive devices is attached to patient's body, periodically collects information about different vital signs or his/her movements and, then, sends the collected data to a mobile device (e.g. smart phone). The mobile device locally stores such data and, then, indirectly calculates both further vital signs and user's movement intensity starting from biomedical or accelerometer information monitored in a direct manner. Successively, it locally correlates all this information, by means of medical rules also able to handle uncertainty and vagueness, in order to identify abnormalities altering the health status of the patient. These rules can be customized and configured according to the type of patient and his/her state of health, in line with existing medical guidelines. Finally, by using different channels, i.e. email and text messages, it can send a daily monitoring report to the doctor for his/her evaluation as well as it can notify the occurrence of potential abnormalities in order to require the intervention of a caregiver or of the doctor, depending on the severity of the abnormality.

The architecture promotes high levels of mobility for both patients, doctors and caregivers. This mobility has a straightforward impact on availability since they would be able to use monitoring applications built on the top of it anytime and anywhere. Mobility is supported at different levels: (i) wireless and wearable sensors based on Bluetooth Low Energy 4.0 (BLE) communication protocol are chosen to not limit the patient's daily activities, by ensuring hours of continuous monitoring, granting the requirements of safety and reliability and, contextually, reducing the energy consumption, the interference and the transmission power; (ii) each application is completely deployable on a mobile device, whose battery life is usually enough for a fully monitoring day, before a new wired charging session is requested, thus giving the patient the possibility of being continuously evaluated with respect to his/her

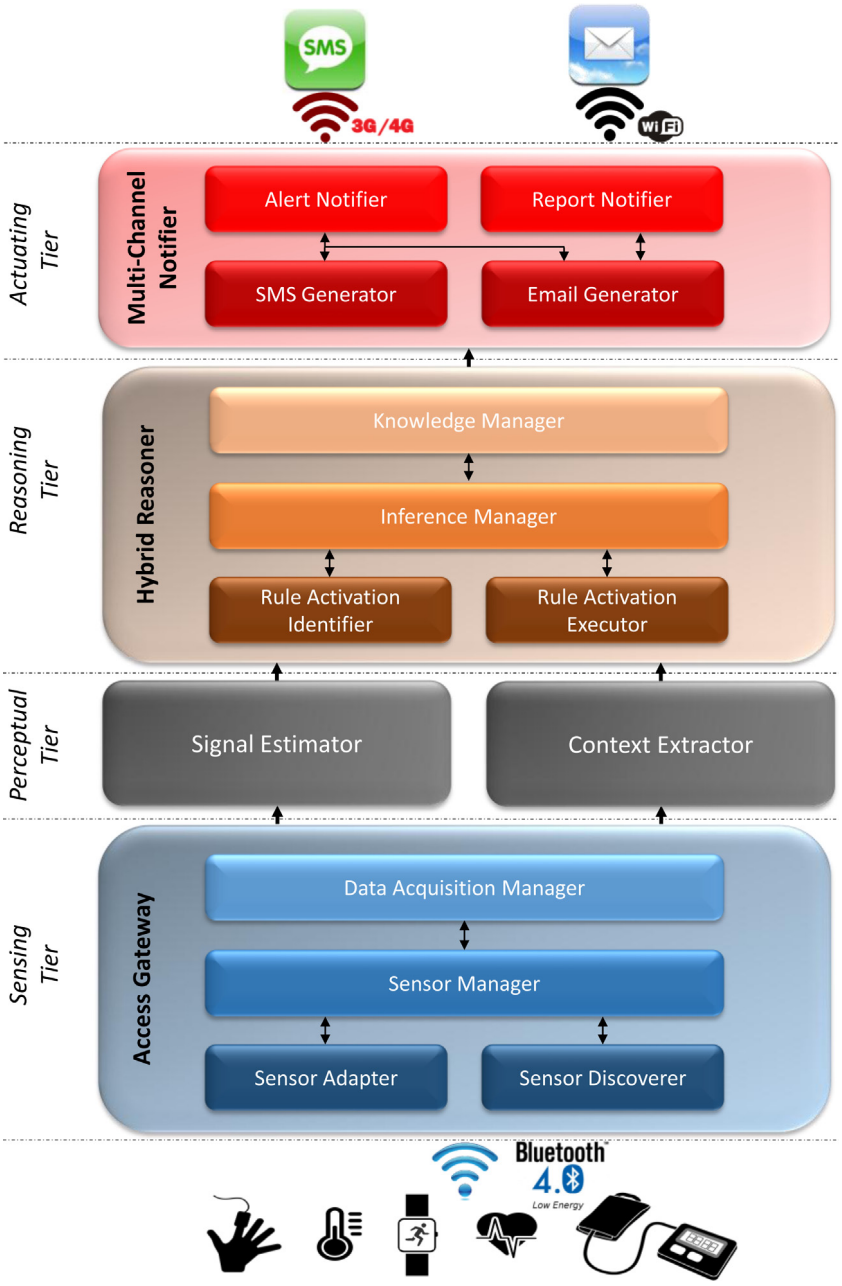


Fig. 2. The main components of the proposed architecture.

health status, without any service interruption; (iii) Wi-Fi and/or 3G/4G networks, available with a very high probability in most areas, allow applications to timely communicate, when necessary, with doctors or caregivers, reaching them in every place.

Fig. 2 presents the overall architectural view, organized as a composition of four tiers that operate on a layered fashion. In detail, each tier provides a particular set of functionalities with a predefined interface. The advantage of a layered architecture is that each tier depends only on lower tiers and does not have knowledge about higher ones. It has been also designed modularly so that changes in one component of a tier do not alter the whole architecture and do not affect the configuration of other components belonging to the same tier or to other ones. A unified data model has been defined to enable the data interoperability required to ensure intercommunication among the components of the architecture and the monitoring applications built on the top of them.

The four tiers are as follows:

- A sensing tier comprising the set of components aimed at gathering parameters characterizing the health status of the patient. Sensors are used to collect both biomedical signals and accelerometer information from the patient. All the sensed information is then locally stored on the mobile device in a dedicated database.
- A tier of perceptual components based on signal-processing and context-extraction algorithms. Perceptual components extract vital and context cues, by processing biomedical and accelerometer sensed data. Information derived from perceptual components relates to vital signs not-directly measured on the patient and the intensity of his/her physical activity.
- A reasoning tier made of intelligent components that model and correlate vital parameters and contextual information pertaining to the patient's physical activity in order to detect

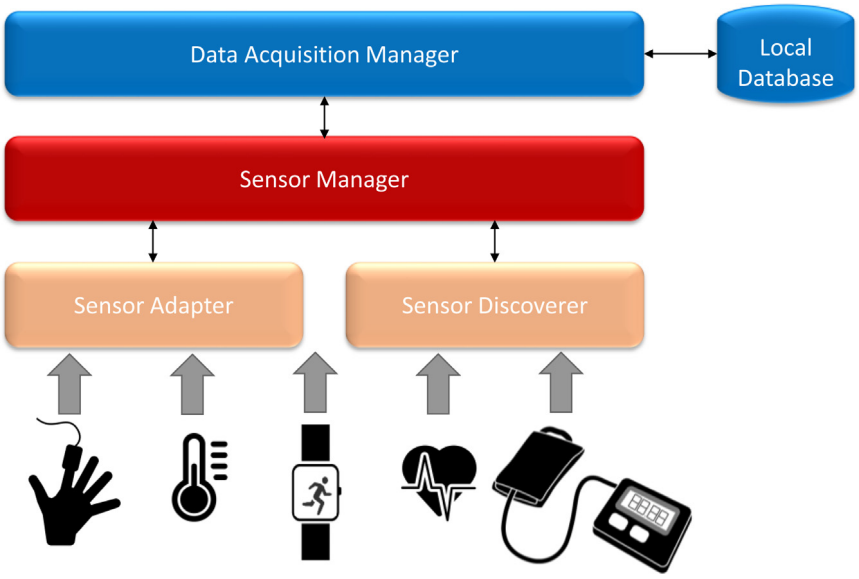


Fig. 3. The main components of the access gateway.

possible anomalies, by incorporating a set of personalized and patient-specific rules characterizing each particular monitoring application.

- A tier of actuating components that provide the means to communicate with caregivers and doctors, by sending emails or text messages. This tier selects the optimal communication channel according to the health status of the patient and the severity of the detected anomalies.

In the following, first, the main components of each layer of this architecture and, then, the data model are diffusely described.

3.1. Sensing tier

This tier is aimed at interfacing with multiple heterogeneous mobile, wearable, biomedical devices, as well as acquiring, processing and understanding their data streams. This behavior is realized by means of an access gateway devised to provide the abstraction level required to enable the functioning of the upper layer of the architecture independently of the underlying sensors. In other words, this component concretely gives the architecture the capability of configuring itself with respect to several heterogeneous wearable sensors, by offering "plug-and-play" facilities to grant not only a transparent and automatic connectivity and communication, by hiding the complexity of proprietary and cumbersome protocols, but also the correct data retrieval and interpretation, by exploiting the data model presented successively.

The main sub-components of the access gateway are shown in Fig. 3.

The *Data Acquisition Manager* (DAM) is the component devised to receive by the upper layers of the architecture the requests for some typologies of sensed data, according to the requirements of a specific monitoring application, and enable their acquisition independently of the underlying physical sensors to be concretely used for their measurement. Generally, a physical sensor is sensitive to one special phenomenon, monitors some relevant change and, then, it converts this change into data (Li et al., 2015). The approach here adopted is to represent the different observable phenomena through specific entities of the proposed data model, so giving them a uniform and shared semantics. Such a way, the DAM is able to receive requests for semantically well-defined entities that model observable phenomena, and, successively, communicates with the *Sensor Manager* (SM) to identify typologies of sensors able to measure them.

In particular, first, the SM is in charge of registering, into a local directory, one or more BLE individual sensors for each typology. Two different modalities of communication can be configured for all the sensor typologies. The first one consists in sending the sensor reading once at a certain configurable frequency and then returning to sleep mode, with a low-power consumption extending battery life. On the other hand, the second one consists in enabling a higher data rate in order to monitor a specific phenomenon with a finer grain, but with a superior power consumption.

Moreover, the SM is devoted to search for all the individual devices belonging to a certain typology among the ones preliminarily registered. Successively, it communicates with the *Sensor Discoverer* (SD) in order to detect how many among the registered devices are active for establishing a BLE connection with them. Finally, it selects one among these active devices according to the requirements of the monitoring application, such as the need of being not obtrusive or of using a more complex sensing device able to simultaneously measure more phenomena, or, more simply, in a random fashion.

Once the sensor is chosen by the SM, this information is used to find a matching *Sensor Adapter* (SA) that knows how to communicate with it in full capacity. Indeed, each sensor is designed to respond to a different message-passing sequence depending on the sensor manufacturer. Even though sensors and external mobile devices may use the same communication technology/ protocol (e.g. TCP, UDP, Bluetooth), the exact communication sequence can be varied from one sensor to another.

Therefore, many different SAs are provided to enable transparent communication and data retrieval with heterogeneous BLE sensing devices, by means of specific Application Programming Interfaces (APIs) able to facilitate access to their low-level capabilities. Such a way, every SA manages the connection with a specific device, interprets the received data and maps it to the unified data model. More specifically, these APIs consist of a hierarchy of sensor-related classes and their operations. The class hierarchy includes vendor-independent classes for typologies of sensors based on BLE communication protocol, e.g. pulse oximeters, thermometers, activity trackers, glucometers, heart rate and blood pressure monitors and so on. On the other hand, vendor-independent operations offer virtualized access to the capabilities of the underlying sensors. Any component interfacing with a certain sensor type exploits the same single API, regardless of the sensor vendor. As a result, the upper layers of the architecture need only to leverage a simple API to interface with each sensor type. This modularity makes the approach simply extensible and evolvable to future devices and technologies.

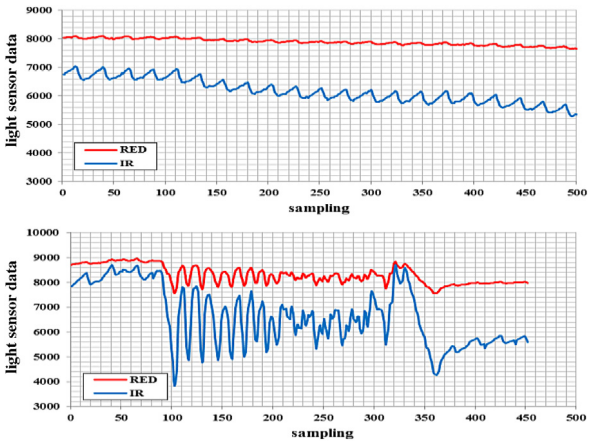


Fig. 4. Reliable acquisition with $R_{index} = 1$ (upper part) and noisy acquisition with $R_{index} = 4$ (lower part) of a pulse oximeter signal. Sample intervals and light sensor signals are displayed on x and y axes, respectively.

All the data measured by a specific SA are considered meaningful by the DAM only if accurate, reliable, and effective. To this aim, a *reliability index* (R_{index}) is defined, whose range is set from 1 to 4. The value 1 indicates the highest level of reliability, i.e. the sensed value has a high Signal to Noise ratio (S/N), whereas the value 4 indicates the lowest level of reliability, i.e. the value acquired has a low S/N and is not truthful since affected by noise, due to motion artifacts or to other effects, such as photon diffusion. As a result, the DAM can be configured in order to recognize as meaningful and store in the local database all the sensed data or a subset of them according to the value of the R_{index}. For instance, all the sensed data with R_{index} equal to 1 or 2 can be stored in the local database for further analysis since considered trustworthy, whereas all the ones with R_{index} equal to 3 or 4 can be discharged and not persistently saved, since evaluated as not reliable.

As a further example, Fig. 4 reports a reliable acquisition ($R_{index} = 1$) of a pulse oximeter signal by means of a wrist-type sensing device in its upper part, and a noisy acquisition ($R_{index} = 4$) of another signal affected by motion artifacts, which is obtained by the same device and subject, in its lower part.

3.2. Perceptual tier

This tier contains primary mechanisms for the treatment and refinement of data acquired by means of physical sensors and communicated by the lower-layer components of the architecture. It is made of two components, namely the Signal Estimator (SE) and the Context Extractor (CE).

In more detail, the SE is devised to estimate both physiological and movement information from data sensed by wearable sensors. First, it is aimed at emulating biomedical sensors that are fundamental for assessing the person's health status, since able to provide meaningful information about some vital signs. In other words, it is able to generate information about certain vital signs starting from sensors that are properly designed to observe other ones. As an example, the SE can be used to measure heart rate by means of wrist-type pulse oximeters, which are specifically designed to measure the blood oxygen saturation (SpO_2).

Moreover, it is also able to replace, even if, often, not satisfactorily, invasive devices attached to the human body, that are usually annoying for patients and limit their movement. As an example, it can be used to measure SpO_2 by using wrist-type pulse oximeters instead of fingertip ones. Fingertip pulse oximeters are largely adopted and very accurate in static environments, but they cannot be worn, interfere with daily activities and generate problems associated with their placement. On the other hand, wrist-type pulse oximeters are easy to wear and do not interfere with daily activities, but they generate measurements affected by motion artifacts that often corrupt the observed signal. Another important source of artifacts for these devices is the photon diffusion; in fact, the measurement of the light is made for diffusion in wrist-type pulse oximeters, whereas it is made for transmission in fingertip ones.

To effectively estimate physiological information from a signal observed by a wearable sensor used in a dynamic environment, the SE implements signal processing techniques to analyze sensed data, filter the noise, extract desired signals from moving windows over the input data streams and, finally, calculate measures of specific vital signs. The extracted signals can be further elaborated in order to filter outliers and provide a more stable reading by fusing different estimates over successive windows.

Secondly, the SE is in charge of calculating more complex movement information starting from raw accelerometer data. To this aim, it integrates signal processing techniques to assess the intensity of physical activities performed by the patients by analyzing the characteristics of movement-induced accelerations and filtering interfering acceleration signals, which do not always correlate well with the movements taken. Accelerometers have been preferred to measure physical activity because acceleration is proportional to external force and, hence, can reflect intensity and frequency of human movement. On the other hand, since only the intensity of the physical activity is of interest, the SE neither calculates information on the pattern or duration of specific activities (i.e., how many steps a person accumulated at 2:00 pm while going to the restaurant), nor distinguishes one intensity level from another (i.e. if one person sprinted 100 steps and a second person walked 100 steps, the SE simply records approximately 100 steps for each person).

Finally, the CE is essentially a means of turning physical and simulated sensor data into symbolic context descriptions. Generally speaking, context is any piece of information that characterizes a situation regarding an entity, such as a person. According to this definition, the CE is in charge of constructing primitive context types that map directly to measurable physical aspects or calculated ones, and are characterized by a set of conceptual states that a certain entity can assume. In order to make these primitive context types meaningful to the upper layers of the architecture, i.e. usable for generating inference and producing higher-level knowledge, their states are represented in the CE such that they reflect human reasoning, handling uncertainty and imprecision. More concretely, the conceptual states of a primitive context have been modeled by means of Fuzzy Logic (Zadeh, 1965), making the distinction between the gradual states as opposed to well-defined boundaries.

Indeed, Fuzzy Logic is based on the concept of *fuzzy set*, which is a set characterized by a gradual degree of membership of its elements. Differently, the boundary of a classical set is crisp, i.e. the membership of an element to a classical set is ruled by a dichotomic condition: either it belongs or does not belong to the set (Zadeh, 1965). The main advantage of Fuzzy Logic, over more conventional approaches in solving complex, nonlinear and/or ill-defined problems, lies in its capability of incorporating a priori qualitative knowledge and providing a procedural morphology of approximate reasoning on the top of it for drawing imprecise conclusions from existing imprecise data. This makes Fuzzy Logic almost indispensable for obtaining a transparent qualitative insight and human-like inferential procedures for systems, where knowledge representation and reasoning with exact mathematical models is poor and inadequate.

As a result, primitive context types modeled by means of Fuzzy Logic are able to provide a more realistic representation of a particular aspect characterizing an entity, with a better tolerance for imprecision, uncertainty and partial truth that can arise in data gathered from physical sensors or calculated from simulated ones.

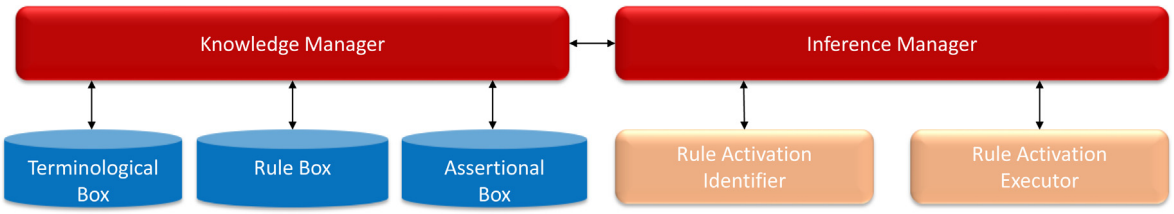


Fig. 5. The architecture of the Hybrid Reasoner.

3.3. Reasoning tier

The essential necessity of producing reliable alarms or notifications makes the *Reasoning tier* the core of the architecture. Essentially, this tier is aimed at receiving preprocessed data from physical or simulated sensors as input by means of the lower layers, conducting an analysis to determine abnormal situations, and identifying which of them must effectively generate notifications to the caregivers or to the doctor. This behavior is realized by means of a hybrid, rule-based reasoner designed to support inferential procedures directly on mobile devices. It relies on hybrid production rules as its basic unit of computation and employs the classical recognize-act cycle, made of two main phases, namely *pattern matching* and *rule firing*. In the pattern matching phase, the system looks for the first applicable rule instance, also named *rule activation*, which is matched by the most recent data characterizing the system state. Then, in the rule firing phase, the system executes the matched rule activation, updates its state and cycles back to the pattern matching phase. The main sub-components of the reasoner are shown in Fig. 5.

In detail, the *Knowledge Manager* (KM) and the *Inference Manager* (IM) are the main interfaces between the reasoner and the other architecture layers.

The KM accesses and updates the current system knowledge after being invoked by the IM by handling knowledge base repositories. In particular, the repositories called *Terminological Box* (Tbox) and *Assertional Box* (Abox) contain, respectively, knowledge about a domain, expressed in terms of classes and properties, and collections of domain objects, also named facts, described as individuals (instances of classes) with the corresponding instances of properties. In addition, the repository named *Rule Box* (Rbox) contains different production rules, namely *classic rules* operating only on precise information and *hybrid rules* working with both precise and vague information and eventually grouped when cooperating to generate a shared outcome. Both these typologies of rules contain a conjunction of condition elements (CEs) in its left-hand side (LHS), and a set of action elements (AEs) in its right-hand side (RHS).

The IM enacts a forward chaining scheme, searching for eligible rules in the Rbox starting from the elements contained in both the Tbox and Abox, and drawing all possible new inferred facts. It is in charge of invoking the other components of the Reasoner, namely the *Rule Activation Identifier* (RAI) and the *Rule Activation Executor* (RAE), in order to ensure the correct flow of inference execution, the proper knowledge updating, and the notification of inference outcomes to external components.

In more detail, at each recognize-act cycle, the IM invokes the RAI to determine the rule activation to execute. To this aim, the RAI first applies a specific pattern-matching algorithm to process the Tbox and Abox elements and tests if they satisfy CEs contained in the LHSs of rules stored in the RBox. Such a way, all the eligible rule activations are determined and, among them, only one is selected according to a predetermined resolution scheme. Successively, the IM invokes the RAE to execute the AEs included in the RHS of the rule activation identified and produce possible updates on the Tbox and Abox.

3.4. Actuating tier

An important characteristic of several monitoring applications is the generation of alerts, notifications, reports and other future advanced functionalities. Essentially, this tier is aimed at receiving and assembling the set of data observed or estimated enriched with context information calculated and possible anomalies detected by means of the lower layers, and providing this information to doctors and caregivers. This behavior is realized by means of a Multi-Channel Notifier (McN) designed to generate appropriate notifications by exploiting the communication channels offered by mobile devices, i.e. e-mails and text messages. For this reason, both the phone numbers and email addresses of caregivers and doctors should be given as inputs to the McN.

In particular, this McN is made of two sub-components, namely an Alert Notifier (AN) and a Report Notifier (RN), as shown in Fig. 6.

The AN provides mechanisms to trigger alerting procedures when abnormalities are detected. These mechanisms are automatic emails and text messages, respectively provided by an SMS Generator (SG) and an Email Generator (EG). Emails and text messages may be delivered to both caregivers and doctors in the event of a critical abnormal situation, i.e. when an anomaly is detected at least twice consecutively from a time perspective (i.e. labeled with a “red" color). In particular, text messages are chosen to contain only short descriptions of the anomalies detected, whereas the emails should contain more accurate reports of all the information acquired, estimated or inferred until the occurrence of the last anomalies detected. All this information is stored in the local database foreseen by the architecture. Such a way, it is possible to enable caregivers or doctors to have a complete view of the health status of the patient in the last monitoring period.

On the other hand, the RN is in charge of offering mechanisms to generate reporting procedures at the end of a scheduled monitoring period. In this case, these mechanisms are offered by the EG and consist in automatic emails that may be delivered to both caregivers and doctors, for instance at the end of each day. In particular, the emails should contain daily reports of all the information acquired, estimated or inferred, inclusive of also non-critical abnormal situations, i.e. anomalies that are detected only once (i.e. labeled with a “yellow" color). Such a way, it is possible to reduce the number of alerts generated inappropriately, thus reducing the alert fatigue and improving the responsiveness of caregivers and doctors to the really critical situations.

3.5. Data model

The proposed data model has been conceived as formal, semantically well-defined, flexible and extensible to support the representation and integration of heterogeneous data between the above-mentioned components of the architecture. An ontological approach has been chosen to represent this model for the following reasons: (i) a common ontology enables knowledge sharing in an evolvable environment, where new sensing devices can be added; (ii) ontologies with their high and formal expressiveness together with their well-defined declarative semantics provide a means for enabling automatic reasoning mechanisms on the represented information; (iii) explicitly formalized ontologies allow devices and architectural components to simply interoperate among them.

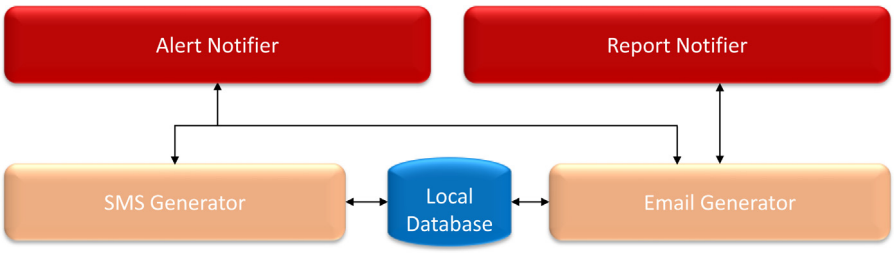


Fig. 6. The architecture of the Multi-Channel Notifier.

Generally speaking, an ontology is intended as a vocabulary and a set of terms and relations that define, with the needed accuracy, a set of entities enabling the definition of classes, hierarchies, and other relations among them (Gruber, 1995; Guarino, 1995). In other words, it enables to produce a representation of knowledge about a part of an abstract or real world with the characteristics of being: (i) formal, since it is machine readable and interpretable; (ii) semantically well-defined, since it defines concepts, properties, relationships, that characterize a domain of interest using a shared vocabulary; (iii) flexible and extensible, since it can be simply altered, by adding more classes to its scheme.

In detail, the proposed data model has been designed as "an application ontology" (Guarino, 1997), i.e. describing concepts dependent on a particular domain and task; in particular, the domain chosen is the healthcare and the task is the monitoring of individual health conditions by means of wearable sensors and mobile devices. For this reason, the vocabulary for this ontology model has been constructed as a collection of concepts strictly related to the specific class of applications that can be built by exploiting the proposed architecture. This collection of concepts has been encoded into a **high-level ontology**, which captures more general knowledge about health monitoring applications, and one or more **low-level ontologies** built on the top of it, which define the details of this general knowledge for each particular monitoring application.

The high-level ontology has been realized by employing a middle-out approach in order to identify the basic concepts composing the domain structure. This strategy enables to identify the most relevant concepts in the domain of interest before moving on to more abstract and more concrete ones. In such a way, it strikes a balance in terms of the level detail. Indeed, detail arises only as necessary, by specializing the basic concepts, so some effort is avoided and higher-level concepts are more likely to be stable. This, in turn, leads to less re-work and less overall effort (Uschold and Gruninger, 1996). The concepts identified by means of this approach have been categorized with respect to four distinctive but related themes: (i) concepts that define sensors and the typologies of physically observable or simulated parameters they are associated to, (ii) concepts that define the measurements associated to each parameter, (iii) concepts that describe the subject and their characteristics, and (iv) concepts that describe the potential anomalies that can be detected and their attributes. Next, the properties of the concepts, i.e. attributes, and the relationships between these concepts have been specified.

Successively, on the top of the high-level ontology, low-level ontologies have been realized, by identifying instances of more general domain concepts and proprieties, in order to define peculiar aspects depending on the specific application of interest.

Both these two typologies of ontologies have been concretely developed using the Protégé tool (Musen, 2015) and encoded in Ontology Web Language (OWL), since characterized by an appropriate expressive power. In particular, in OWL, *classes* are used to represent concepts, *individuals* to model instances of concepts, *datatype properties* to express characteristics of individuals, and *object properties* to indicate relations between individuals. The Protégé tool, and, in particular, the Pellet reasoner, which is an OWL checker coming together with it, has been also used to test and validate the ontologies realized with respect to their correctness and consistency.

For the purposes of this paper, each ontology is formally defined as the tuple $O := \langle O^T, O^A \rangle$ where O^T is the terminological part of the ontology and O^A is its assertional part defined over the entities in O^T. In the following, classes and properties are denoted in **bold,** whereas individuals are indicated in *italics*. Moreover, assertions in the form *x:* **C** and **P***(x,y)* state that "individual *x* is an instance of class **C**" and "individual *x* is related to *y* by means of **P**", respectively.

3.5.1. The proposed high-level and low-level ontologies

The proposed high-level ontology defines the terminological part O^T by formulating statements about concepts and properties as graphically described in Fig. 7.

In more detail, the set of subjects is represented by the class **Subject**, whose individuals are the patients to be monitored. The class **Measure** represents the sets of possible measurements that can be acquired when a patient is monitored, whereas the class **Parameter** indicates the possible typologies of parameters, either directly measurable or indirectly calculated, to which a measure belongs. This class is further specified into two sub-classes, namely **ActivityParameter** and **VitalParameter**, representing the sets of parameters pertaining to the physical activity performed by the patient or his/her vital signs, respectively. The class **Sensor** defines the sets of possible sensing devices usable to monitor a patient. Also, this class is further specialized into two sub-classes, namely **PhysicalSensor** and **SimulatedSensor**, representing the sets of real sensors that can be physically worn by the patients or of virtual sensors that can indirectly estimate the value of a measure by elaborating other sensed information, respectively. Finally, the class **Anomaly** indicates the sets of possible abnormal situations that can affect the health status of a patient.

The class **Subject** is linked to the class **Measure** through the object properties **measure**(**Subject**, **Measure**) and **restMeasure**(**Subject**, **Measure**) to indicate, with respect to a certain parameter, a measurement observed or estimated in a time instant or a reference measurement evaluated at rest.

Moreover, it is also linked to the class **Anomaly** through two object properties, namely **anomaly**(**Subject**, **Anomaly**) and **previousAnomaly**(**Subject**, **Anomaly**), to correlate a patient with the current anomaly and, if existing, also to the last occurred one.

Finally, it is further characterized by means of six datatype properties, namely **surname(Subject, string), name(Subject, string), age(Subject, integer), sex(Subject, string), height(Subject, string), weight(Subject, string),** for modeling some relevant anthropometric information characterizing a patient.

The class **Measure** is linked to the class **Parameter** through the object property **parameter** (**Subject**, **Measure**) to correlate a measure to the type of parameter it refers to. It is further characterized by means of three datatype properties, namely **date(Measure, date), relIndex(Measure, float), value(Measure, float),** for modeling the date of acquisition of the measure, its reliability index and, finally, the observed or estimated value.

The class **Parameter** is linked to the class **Sensor** through the object property **sensor**(**Parameter**, **Sensor**) to correlate a parameter to the sensor able to produce it. It is further characterized by means of three datatype properties, namely **name(Parameter, string),**

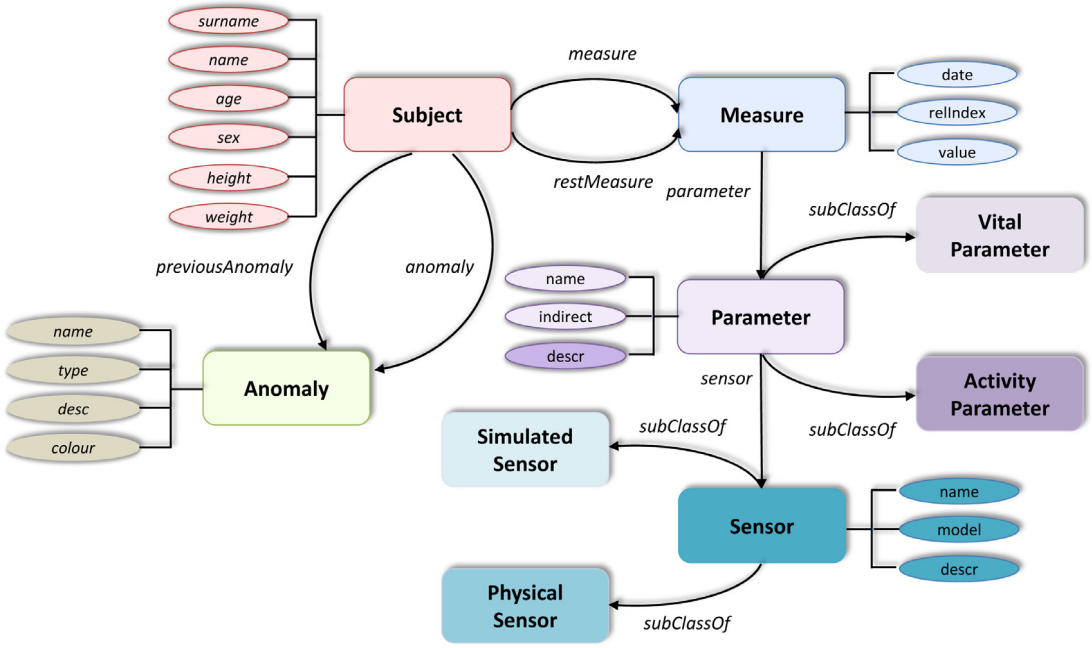


Fig. 7. The high-level ontology for describing data shared by the components of the architecture.

indirect(Parameter, bool), descr(Parameter, string), for modeling the name of the parameter, its typology of acquisition and, finally, a description of it.

The class **Sensor** is characterized by means of three datatype properties, namely **name(Sensor, string), model(Sensor, string), descr(Sensor, string),** for modeling the name of the sensor, its model and, finally, a description of it.

Finally, the class **Anomaly** is characterized by means of four datatype properties, namely **name(Anomaly, string), type(Anomaly, string), descr(Anomaly, string), color(Anomaly, string),** for modeling the specific name of the anomaly (such as tachycardia at rest), its type (such as tachycardia), an extensive description of it and the color (such as yellow or red) indicating its severity calculated depending on the number of occurrences.

On the other hand, the low-level ontologies are essentially constructed by populating the assertional part O^A defined over the classes and properties specified in the terminological part O^T of the high-level ontology. On other words, depending on the specific scenario of interest, a set of individuals can be defined that are instances of the general classes of the high-level ontology in order to model peculiar aspects not priory included in it.

As an example, class assertions that can be defined in a low-level ontology pertaining to an application for monitoring blood pressure are:

- *bloodPressure*: **VitalParameter**
- *sphygmomanometer*: **Sensor**
- *hypertension, hypotension*: **Anomaly**

4. Implementation and evaluation of the proposed architecture

The proposed architecture has been implemented in Java and released to operate on the Android Operating System (OS). The choice of Android OS is due to the fact that it is open source and has received a great support in the community for developing mobile applications and, contextually, integrating these ones with sensors and devices for the monitoring and acquisition of data and/or parameters of interest. Moreover, different software components to interface with biomedical devices produced by third parties are developed for the Android OS. Furthermore, SQLite database[1] is used to implement the local persistence, since specifically devised to operate on resource-limited mobile devices, consuming a minimal stack space and a very little heap.

The architecture has been then evaluated by adopting the ALMA method, since it has had a remarkable impact and acceptance in the scientific community (Villarreal et al., 2014). This method has been applied to perform a modifiability analysis of the architecture, i.e. to evaluate its ability to be simply modified and evolve over time. It consists of the following five steps:

- *Set goal*: determine the aim of the analysis, choosing among risk assessment, maintenance and costs prediction, or software architecture selection.
- *Describe software architecture (s)*: provide a description of the most important parts of the architecture, by decomposing it into components, the relationship between components, and the relationships between it and its environment.
- *Elicit scenarios*: select the set of relevant scenarios that may play a role in architecture modifiability, by cooperating with the relevant stakeholders.
- *Evaluate scenarios*: determine the effect of the set of scenarios and express the result in a suitable and measurable way for the goal of the analysis.
- *Interpret results*: interpret the results in accordance with the goals of the analysis and verify them against system requirements.

With reference to the first step, the goal of this analysis has been to make a prediction of the costs of modifying or adapting the architecture for new health monitoring applications. To this end, a prediction model for relating impact and effort required to implement change scenarios is used, whose form is reported as follows:

$$C_{average} = \frac{\sum_{i=1}^{n} C\left(change_i\right) \times w\left(change_i\right)}{n}$$

where $C(change_i)$ denotes the effort or cost required to realize the i-th change scenario, and $w(change_i)$ denotes the specific weight of this scenario.

[1] SQLite, Available online: http://www.sqlite.org/.

Table 2
Summary of the architectural components and their functions.

Function	Component	Relation between components
Sensing	Access Gateway	Signal Estimator, Context Extractor
Perception	Signal Estimator	Access Gateway, Context Extractor, Hybrid Reasoner
Perception	Context Extractor	Access Gateway, Signal Estimator, Hybrid Reasoner
Reasoning	Hybrid Reasoner	Signal Estimator, Context Extractor, Multi-Channel Notifier
Actuation	Multi-Channel Notifier	Hybrid Reasoner

The second step has determined the parts of the architecture realizing a specific function and the relationships between these parts, as outlined in Table 2.

The third step has regarded the elicitation of the change scenarios that are most likely to occur in a short period of time, i.e. one year. A bottom-up elicitation technique is used by interviewing the software designers in possess of knowledge of the architecture, producing twenty change scenarios that can occur when new health monitoring applications have to be realized. For this paper, we have selected five main change scenarios, each of which has been chosen in order to evaluate modifiability with respect to one specific component of the architecture reported in Table 2:

- S1. Add a new Bluetooth sensing device to replace an old one.
- S2. Change the vital parameters to be indirectly estimated from raw sensed data.
- S3. Change the context information to be extracted from raw sensed data.
- S4. Add a new domain-specific ontological model and a new collection of rules for monitoring and managing a specific health condition.
- S5. Add a new channel for transmitting reports and alerts.

Successively, weights have been associated to the scenarios in order to indicate their probability of occurrence. These weights $w(change_i)$ have been calculated as the number of times that a specific change scenario is expected to occur during the prediction period. These estimates have been also normalized by dividing the number of occurrences of each change scenario by the sum of occurrences of all the scenarios, as shown below:

$$w\left(change_i\right) = \frac{occurrences\left(change_i\right)}{\sum_{j=1}^{n} occurrences\left(change_j\right)}.$$

The result of this normalization is that all the scenarios have a weight between zero and one and that the sum of all scenario weights is exactly one. The list of change scenarios with their normalized weights represents the scenario profile outlined in Table 3.

Successively, the next step has consisted in evaluating the impact of each change scenario contained in the profile. To this aim, first, the functions already present in the architecture and required to realize each change scenario have been identified. Successively, the effects of the changes on the components directly associated to the identified functions in Table 2 have been assessed with respect to the lines of code (LoC) that are needed for the changes.

In this evaluation, the architecture has been carefully examined and the modifications to be performed on the existing components have been conceived and estimated as described in the following:

- S1. Only a new Sensor Adapter must be added to the Access Gateway, whereas no change affects its remaining sub-components. Taking into account the average size of Sensor Adapters already implemented, this change can be estimated approximately as equal to 800 LoC.
- S2. Since the Signal Estimator implements a family of signal processing techniques to analyze and filter sensed data, the changes necessary to estimate a novel vital sign impact only on the configurations of these techniques or on very light customizations to properly work on the specific case considered. As a result, this change can be estimated approximately as equal to 150 LoC.
- S3. The Context Extractor integrates a family of signal processing techniques to analyze the characteristics of movement-induced accelerations and extract context information pertaining to the physical activities performed by the user. For this reason, more impacting changes must be performed to estimate another type of context information, since completely novel algorithms should be implemented and integrated. On the other hand, no change must be performed to integrate these novel algorithms into the Context Extractor and represent the produced information by using Fuzzy Logic. As a result, this change can be estimated approximately as equal to 500 LoC.
- S4. The Hybrid Reasoner is built to support customization and personalization, since it gives the possibility of changing domain-specific ontological models and rules built on the top of them. For this reason, taking into account the average size of other ontological models and collection of rules already implemented, the changes required to realize this scenario can be estimated approximately as equal to 350 LoC.
- S5. Only a new component to physically transmit a notification must be implemented and added to the Multi-Channel Notifier, whereas no change affects its remaining sub-components. Taking into account the average size of the similar components already implemented, this change can be estimated approximately as equal to 500 LoC.

Moreover, the ripple effects have been considered in this evaluation, but the other components of the architecture not directly associated to the identified functions have resulted as not impacted by the changes. Indeed, the interfaces of the aforementioned components affected by the changes have been not altered at all.

Summarizing, a set of estimates of the modification volume for each affected component has been obtained, as depicted in Table 4. In addition, also the average effort per scenario has been calculated by applying the prediction model introduced above.

The last step has regarded the interpretation of the results achieved. In detail, change scenarios S1 and S5 require more effort because they impact on components of the architecture that are more linked to specific monitoring applications to be developed. Indeed, these scenarios involve the architectural components that are used to gather different data from the patient or to transmit information to doctors and caregivers, and, thus, can be more affected by modifications depending on the specific application requirements. In scenarios S2 and S4, changes made to the architecture are minimal, requiring little customization or lightweight model personalization. Finally, scenario S3 requires a substantial effort for changes due to its actual specialization with respect to a certain typology of context information that it is able to handle.

In conclusion, assuming that the overall productivity per software engineer in making and testing the modifications to the code baseline of the architecture is in accordance with the values published in the software engineering literature (Henry and Cain, 1997), i.e. 40 LoC/month for modifying existing code, the average time required to perform the changes is approximately equal to 2.325 man-months per scenario. As a result, the architecture appears sufficiently flexible and robust for being rapidly customized, personalized or eventually modified with respect to possible change scenarios, highlighting its capability of enabling software developers to prototype, with a reduced effort, novel health monitoring applications on the top of its components.

Table 3
Scenario profile.

ID	Scenario	Occurrences	Weight
S1	Add a new Bluetooth sensing device to replace an old one.	8/30	0.27
S2	Change the vital parameters to be indirectly estimated from raw sensed data.	6/30	0.20
S3	Change the context information to be extracted from raw sensed data.	4/30	0.13
S4	Add a new domain-specific ontological model and a new collection of rules for monitoring and managing a specific health condition.	10/30	0.33
S5	Add a new channel for transmitting reports and alerts	2/30	0.07

Table 4
The efforts and the calculation of average effort per scenario.

ID	Scenario	Weight	Effort
S1	Add a new Bluetooth sensing device to replace an old one.	0.27	~800 LoC
S2	Change the vital parameters to be indirectly estimated from raw sensed data.	0.20	~150 LoC
S3	Change the context information to be extracted from raw sensed data.	0.13	~500 LoC
S4	Add a new domain-specific ontological model and a new collection of rules for monitoring and managing a specific health condition.	0.33	~350 LoC
S5	Add a new channel for transmitting reports and alerts	0.07	~500 LoC
	Average effort per scenario		~93LoC

5. Case study: a mobile application for monitoring cardiac arrhythmias

As a real case study, the proposed architecture has been employed in the context of the Italian project "Bersagli" to realize a mobile application for monitoring and managing cardiac arrhythmias, such as bradycardia and tachycardia. This application has been thought and realized in cooperation with medical experts involved in the project, who have followed and supported its design and development. It has been built by using a bracelet equipped with multiple sensors in order to acquire or estimate different vital and context parameters. It has been designed to detect a set of cardiac anomalies due to bradycardia and tachycardia by correlating all the parameters directly acquired or indirectly calculated, together with other anthropometric information manually inserted by means of the graphical user interfaces of the application itself. The occurrence of these anomalies is notified to doctors and caregivers when they persist over time.

The main features together with the graphical user interfaces of this application are diffusely described in the following.

5.1. Data acquisition

Data acquisition is performed by the application by using the bracelet of the Amiigo[2] fitness tracker shown in Fig. 8. It is equipped with a 3-axis accelerometer, a temperature sensor and a reflectance-based pulse oximeter, composed of a light source and a detector, with red and infrared (IR) light-emitting diodes (LEDs).

As a result, a specific Sensor Adapter (SA) has been implemented and integrated in the architecture to enable transparent communication with the bracelet by using BLE transmission protocol and retrieve from it the following parameters: three values of acceleration, one for each axis, one skin temperature value and two values of light intensity related to red and infrared LEDs. This SA allows continuously acquiring measures of both skin temperature and 3-axis acceleration at frequencies equal to 1 per minute and 4 Hz, respectively. On the other hand, it allows cyclically acquiring measures of light intensity related to red and infrared LEDs for a determined time span, whose frequency and period are configurable. The modality of acquisition chosen is the following: each reading lasts 30 s, with a frequency equal to 30 Hz, and is repeated periodically every 30 min. Such a way, the battery life has been experimentally proven to be preserved, surviving for about 2 days. Finally, the SA stores all the sensed data locally into the SQLite database installed on the mobile device.

[2] http://www.amiigo.co.

Fig. 8. The Amiigo bracelet.

5.2. Data processing

Data processing is realized by the application by operating on the information sensed from the bracelet, i.e. 3-axis acceleration, skin temperature and red and infrared light intensities.

5.2.1. Estimation of heart rate and SpO_2 from the pulse oximeter waveform

The analysis of the pulse oximeter waveform has been performed in the frequency domain in order to filter noise due to motion artifacts and estimate both SpO_2 and heart rate. In particular, in order to calculate SpO_2 and heart rate, a moving time window (Hamming) of 100 samples and a shift of 1 sample ahead is considered. For each time window, the Fast Fourier Transform (FFT) is applied to the infrared light intensity and the resulting power spectrum is analyzed. According to the assumption that the heart rate ranges from 45 to 200 beats per minute (bpm), a bandpass filter is preliminary applied to eliminate noise at frequencies lower and higher of those borderline values respectively. If the power spectrum still contains some significant harmonics, first, the heart rate is estimated as the frequency corresponding to the largest one in magnitude. Second, the SpO_2 is calculated as the mean of the logarithmic ratio between red and infrared light intensity values in the window considered. The final heart rate and SpO_2 are calculated as mean of all the values estimated in the different time windows considered. A formal description of the algorithm is given in Algorithm 1.

Algorithm 1 has been experimentally tested on ten volunteers, 5 male and 5 female, aged between 25 and 45 years, with no cardiac and pulmonary disease evidenced. They have been simultaneously equipped

Table 5
Comparison of heart rate calculated with Algorithm 1 on values from the Amiigo bracelet (**A**) and obtained from the Onyx II Model 9560 (**O**) with respect to three different readings of 30 s.

Volunteer	Gender	Age	$\mathbf{A_1}$ (bpm)	$\mathbf{O_1}$ (bpm)	$\left\|\frac{A_1-O_1}{O_1}\right\|$ (%)	$\mathbf{A_2}$ (bpm)	$\mathbf{O_2}$ (bpm)	$\left\|\frac{A_2-O_2}{O_2}\right\|$ (%)	$\mathbf{A_3}$ (bpm)	$\mathbf{O_3}$ (bpm)	$\left\|\frac{A_3-O_3}{O_3}\right\|$ (%)
1	Male	25	80	75	6.67	76	72	5.56	78	73	6.85
2	Male	28	78	73	6.85	79	78	1.28	79	75	5.33
3	Male	33	77	73	5.48	76	75	1.33	81	78	3.85
4	Male	41	70	65	7.69	72	67	7.46	71	68	4.41
5	Male	45	73	71	2.82	72	69	4.35	69	65	6.15
6	Female	27	85	79	7.59	84	80	5.00	83	80	3.75
7	Female	29	82	77	6.49	84	79	6.33	82	78	5.13
8	Female	35	72	78	7.69	75	80	6.25	79	81	2.47
9	Female	40	75	77	2.60	73	77	5.19	76	79	3.80
10	Female	41	68	73	6.85	66	70	5.71	74	75	1.33

Table 6
Comparison of SpO_2 calculated with Algorithm 1 on values from the Amiigo bracelet (**A**) and obtained from the Onyx II Model 9560 (**O**) with respect to three different readings of 30 s.

Volunteer	Gender	Age	$\mathbf{A_1}$ (%)	$\mathbf{O_1}$ (%)	$\lvert A_1 - O_1\rvert$	$\mathbf{A_2}$ (%)	$\mathbf{O_2}$ (%)	$\lvert A_2 - O_2\rvert$	$\mathbf{A_3}$ (%)	$\mathbf{O_3}$ (%)	$\lvert A_3 - O_3\rvert$
1	Male	25	98	97	1	100	99	1	98	98	0
2	Male	28	99	99	0	99	98	1	99	100	1
3	Male	33	96	97	1	97	97	0	99	99	0
4	Male	41	93	94	1	97	96	1	96	97	1
5	Male	45	95	95	0	92	93	1	95	95	0
6	Female	27	100	99	1	98	98	0	98	99	1
7	Female	29	98	97	1	97	96	1	98	97	1
8	Female	35	97	96	1	95	95	0	97	98	1
9	Female	40	93	94	1	94	95	1	95	95	0
10	Female	41	96	96	0	95	96	1	97	98	1

Table 7
Comparison of number of steps calculated with Algorithm 2 on values from the Amiigo bracelet (**A**) and obtained from the HJ-112 Digital Pocket Pedometer from Omron (**H**) with respect to three different time windows of 5 min.

Volunteer	Gender	Age	$\mathbf{A_1}$	$\mathbf{O_1}$	$\left\|\frac{A_1-O_1}{O_1}\right\|$ (%)	$\mathbf{A_2}$	$\mathbf{O_2}$	$\left\|\frac{A_2-O_2}{O_2}\right\|$ (%)	$\mathbf{A_3}$	$\mathbf{O_3}$	$\left\|\frac{A_3-O_3}{O_3}\right\|$ (%)
1	Male	25	25	26	3.85	37	36	2.78	46	48	4.17
2	Male	28	47	45	4.44	24	25	4.00	36	37	2.70
3	Male	33	42	44	4.55	35	34	2.94	45	43	4.65
4	Male	41	21	22	4.55	22	23	4.35	39	38	2.63
5	Male	45	36	35	2.86	46	44	4.55	45	47	4.26
6	Female	27	22	21	4.76	36	36	0.00	23	24	4.17
7	Female	29	38	37	2.70	42	42	0.00	23	22	4.55
8	Female	35	33	34	2.94	31	32	3.13	43	44	2.27
9	Female	40	20	21	4.76	26	27	3.70	20	21	4.76
10	Female	41	41	40	2.50	41	42	2.38	32	33	3.03

with the Amiigo bracelet and a wireless finger pulse oximeter, i.e. the Onyx II Model 9560 from Nonin Medical. According to the modality of acquisition chosen for the bracelet, three different readings of 30 s have been performed for each volunteer by using both the devices. Next, the heart rate and SpO_2 calculated by using Algorithm 1 on the values acquired by the Amiigo bracelet and the ones directly collected by the finger pulse oximeter have been compared, as shown in Tables 5 and 6.

Therefore, for each reading, a mean difference less than 8% for the heart rate and less than 2 units for SpO_2 has been obtained between them. These results confirm the validity of Algorithm 1 applied to the Amiigo bracelet, since comparable with the ones achievable by means of a more accurate but invasive commercial solution.

5.2.2. Estimation of the number of steps from 3-axis accelerometer data

As further data processing functionality, starting from the 3-axis acceleration, the number of steps performed has been estimated by using a step-counting procedure implemented in the SE and inspired to the algorithm proposed in Oner et al. (2012). This procedure applies a peak detection technique to the 3-axis accelerometer data in order to count steps, by simplifying and adapting that algorithm to this context. It essentially assumes that, according to research about human walking dynamics, walking is a cyclical pattern and, thus, steps can be calculated by finding the peaks within a period, where these points represent the highest accelerations along the axes. In detail, for each reading of the bracelet, a number of 3-axis accelerometer samples is collected, depending on both the acquisition time and the sampling frequency. To simplify the computation, the total magnitude of each accelerometer sample is considered, calculated as the square root of the sum of the squares of each of the 3-axis values. At the startup, a baseline is initialized with the total acceleration value of the first sample. After setting this baseline, pairs of consecutive samples are iteratively compared until their total acceleration values are different. In particular, if the oldest sample in the pair is less than or equal to the newest one, the baseline is set with the value of this latter. Alternatively, if the oldest sample in the pair is greater than the newest one, it is compared with the baseline to decide whether it is a peak or not. In particular, if that sample is greater than or equal to the baseline and, contextually, it is also greater than or equal to a threshold characterizing misleading data (i.e. quick repetitious movements), it is recognized as a step. That threshold is calculated as the average of all the total acceleration values of the samples processed thus far, multiplied by a correction factor. This factor allows widening the margin for determining misleading data with respect to sensed accelerometer activities and has been experimentally fixed equal to 1.20 since able to reduce false positive peak detections of about 90 percent, so granting a better tolerance to noisy acceleration values. When all the samples are processed, the total number of steps is represented by the number of peaks detected. A formal description of the algorithm is given in Algorithm 2.

Also, Algorithm 2 has been experimentally tested on the same ten volunteers aforementioned. They have been equipped with the Amiigo bracelet and a commercial pocket pedometer, i.e. the HJ-112 Digital

Algorithm 1

Purpose: Estimation of heart rate and SpO_2 from the pulse oximeter waveform.

Input:

n /* *the number of samples for acquisition* */

$RED = [r_1, r_2, \ldots, r_n]$ /* *the vector of n red light intensity values of the discrete pulse oximeter signal* */

$IRED = [i_1, i_2, \ldots, i_n]$ /* *the vector of n infrared light intensity values of the discrete pulse oximeter signal* */

w /* *the window length* */

sh /* *the shift head* */

f_l /* *the lower bound frequency for heart rate* */

f_h /* *the upper bound frequency for heart rate* */

Output:

hr /* *the estimated heart rate* */

spo_2 /* *the estimated pulse oximeter* */

1: $HR \leftarrow \{\varnothing\}$ /* *initialize the vector of heart rate values estimated in the acquisition period* */

2: $SPO_2 \leftarrow \{\varnothing\}$ /* *initialize the vector of pulse oximeter values estimated in the acquisition period* */

3: $k \leftarrow 0$ /* *initialize the counter for moving time windows on the acquired vector of samples* */

4: **repeat** /* *evaluate iteratively time windows of w samples* */

5: **for** *i = 1 to w, step size 1* **do** /* *extract vectors of w red and infrared light intensity values for the* k^{th} *time window* */

6: $RED_w[i] \leftarrow RED[i + k]$

7: $IRED_w[i] \leftarrow IRED[i + k]$

8: **end for**

9: $IRED_h \leftarrow \text{Hamming}(IRED_w)$ /* *apply a Hamming window to the extracted vector of infrared light intensity values**

10: $IRED_f \leftarrow \text{FFT}(IRED_h)$ /* *apply FFT to the windowed vector and generate the power spectrum* */

11: $IRED_{bpf} \leftarrow \text{BandPassFilter}(IRED_f, f_l, f_h)$ /* *filter the power spectrum at frequencies lower than* f_l *and higher than* f_h */

12: **if** $(IRED_{bpf} \neq \{\varnothing\})$ **then** /* *verify if the power spectrum still contains some significant harmonics* */

13: $ired_{max} \leftarrow \text{Maximum}(IRED_{bpf})$ /* *determine the largest harmonic in magnitude among the significant ones* */

14: $HR(k) \leftarrow \text{Frequency}(ired_{max})$ /* *estimate heart rate in* k^{th} *time window as the frequency of the largest harmonic* */

15: $SPO_{2w} \leftarrow \log_{10}(RED_w/IRED_w)*100$ /* *estimate* SpO_2 *as the mean of log ratio between red and infrared light intensity value*

16: $SPO_2(k) \leftarrow \text{Mean}(SPO_{2w})$ *in the window considered in a percentage form* */

17: **end if**

18: $k \leftarrow k + sh$ /* *update the counter to move the time window ahead of sh samples* */

19: **until** $(w \leq n - k)$ /* *repeat until the number of remaining samples is greater than or equal to w** */

20: $hr \leftarrow \text{Mean}(HR)$ /* *calculate final heart rate and* SpO_2 *as mean of all the values estimated in the different*

21: $spo_2 \leftarrow \text{Mean}(SPO_2)$ *time windows considered* */

Pocket Pedometer from Omron. Three different time windows of 5 min have been considered for each volunteer by using both the devices. Next, the number of steps calculated in these time windows by using algorithm 2 on the accelerometer values acquired by the Amiigo bracelet and the ones directly collected by the commercial pedometer have been compared, as shown in Table 7.

Therefore, for each reading, a mean difference less than 5% has been obtained. Also in this case, this result confirms the validity of algorithm 2 applied to the Amiigo bracelet, since it is comparable with a more accurate but single-purpose solution.

5.2.3. Estimation of the intensity of physical activity from the number of steps calculated

Next, starting from the classification proposed in Tudor-Locke et al. (2005), the intensity of physical activity has been first categorized with respect to the number of steps in an interval of 5 min, as shown in Table 8.

Then, the CE has been used to represent the primitive context "*intensity of physical activity*" on the basis of classes and intervals of steps reported in Table 8 by means of the concepts of *fuzzy variables* and *fuzzy sets*.

A *fuzzy variable* x is defined as the quintuple $FV = (x, T(x), X, G, M)$, where x is the name of the variable, $T(x)$ is the set of symbolic *terms* associated to x, X is the set of numerical values u spanned by x, called *universe of discourse* of x, G is the syntactic rule to generate the name of the terms and M is the semantic rule to associate each term with its meaning, i.e. a fuzzy set defined on X. A *fuzzy set* F on a nonempty set X is defined by its *membership function* $\mu_F: X \rightarrow [0, 1]$, where $\mu_F(x)$ is interpreted as the degree of membership of the element x in the fuzzy set F for each $x \in X$, i.e., $F = \{(u, \mu_F(u)) | \forall u \in X, \mu_F(u) \in [0, 1]\}$ (Zadeh, 1965). By exploiting these fuzzy concepts, the context "*intensity of physical activity*" has been modeled as a *fuzzy variable*, whose terms have been set equal to the classes reported in Table 8. In order to associate a meaning, i.e. a fuzzy set, to these terms, first, the membership function shape has been chosen, and, successively, its design parameters have been defined.

With reference to the shape, a trapezoid has been selected, since it allows introducing vagueness not over all the entire universe of discourse, but only around its boundaries. Indeed, its formal representation is the

Table 8
The classification of the intensity of the physical activity.

Class	Number of steps in 5 min
Zero activity	0 –17
Low activity	17–26
Somewhat activity	26–35
Moderate activity	35–44
High activity	44 to higher

Algorithm 2

Purpose: Estimation of the number of steps from 3-axis accelerometer data.

Input:

	n	/* the number of samples for acquisition */
	$X = [x_1, x_2, \ldots, x_n]$	/* the vector of X-axis accelerometer data */
	$Y = [y_1, y_2, \ldots, y_n]$	/* the vector of Y-axis accelerometer data */
	$Z = [z_1, z_2, \ldots, z_n]$	/* the vector of Z-axis accelerometer data */
	f	/* the correction factor to widen the margin for determining misleading data */

Output:

	st	/* the estimated number of steps */
1:	$ACC \leftarrow \{\varnothing\}$	/* initialize the vector of total acceleration values sensed in the acquisition period */
2:	$st \leftarrow 0$	/* initialize the estimated number of steps */
3:	$avg \leftarrow 0$	/* initialize the average of total acceleration values*/
4:	$th \leftarrow 0$	/* initialize the threshold to detect misleading data*/
5:	**for** $k = 1$ *to* n, *step size* 1 **do**	/* calculate the total magnitude of each accelerometer sample sensed in the acquisition
6:	$ACC[k] \leftarrow \sqrt{X[k]^2 + Y[k]^2 + Z[k]^2}$	period as the square root of the sum of the squares of each of the 3-axis values*/
7:	**end for**	
8:	$bas \leftarrow ACC[1]$	/* initialize a baseline with the total acceleration value of the first sample*/
9:	$i \leftarrow 1$	/* initialize the counter for moving on the vector of total acceleration values */
10:	**repeat**	/* evaluate iteratively pairs of consecutive samples*/
11:	**if** $(ACC[i] \leq ACC[i+1])$ **then**	/* verify if the oldest sample in the pair is less than or equal to the newest one */
12:	$bas \leftarrow ACC[i+1]$	/* set the baseline with the value of the newest sample in the pair */
13:	**else**	
14:	**if** $(ACC[i] \geq bas$ *and* $ACC[i] \geq th)$ **then**	/* alternatively verify if the oldest sample is simultaneously greater than or
15:		equal to the baseline and the threshold */
16:	$st \leftarrow st + 1$	/* increment of one unit the number of steps recognized */
17:	**end if**	
18:	**end if**	
19:	$avg \leftarrow (avg * (i-1) + ACC[i]) / i$	/* update the average of total acceleration values of samples processed so far */
20:	$th \leftarrow avg * f$	/* calculate the threshold as the average multiplied by the correction factor f */
21:	$i \leftarrow i + 1$	/* update the counter to move ahead of one sample */
22:	**until** $(i < n)$	/* repeat until all the samples in the acquisition period are evaluated*/

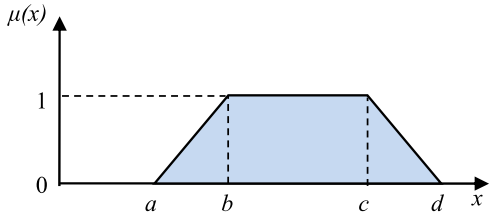


Fig. 9. Trapezoidal membership function.

following:

$$\mu_X(x) = \begin{cases} (x-a)/(b-a) & a < x < b \\ 1 & b \leq x \leq c \\ (d-x)/(d-c) & c < x < d \\ 0 & otherwise \end{cases}$$

where $\mu_X(x)$ is the membership grade of x on X, b and c are *lower* and *upper* bounds for its core (defined as the set of its elements with membership grade equal to 1, and thus not affected by vagueness), a and d are *lower* and *upper* bounds for its support (defined as the set of all its elements with non-zero membership grade), as shown in Fig. 9.

With reference to the design parameters of these fuzzy sets, each of the intervals detailed in Table 8 has been mapped to a trapezoid, by preserving the overall interpretability, i.e. all the resulting fuzzy sets verify the following criteria (Gacto et al., 2011):

$$\begin{cases} \forall x \in X \quad \sum_{i=1,\ldots,F} \mu_{X_i}(x) = 1 \\ \forall X_i, i = 1, \ldots, F \quad \exists x \in X | \mu_{X_i}(x) = 1 \end{cases}$$

where F is the number of fuzzy sets and $\mu_X(x)$ is the membership grade of x on X_i.

Lower and upper bounds for core and support of each fuzzy set have been determined as follows. Mean and standard deviation have been calculated for each interval reported in Table 8. Trapezoids not placed at the edges of the universe of discourse, i.e. associated to the classes *Low*, *Somewhat* and *Moderate*, have been centered in their mean, with a core size assumed equal to twice the minimum of their standard deviations, approximated by defect. Moreover, the upper bounds of the intervals reported in Table 8 have been mapped to the crossover points of these contiguous trapezoids, fixing their membership grade equal to 0.5 in order to preserve the role of points characterized by the maximum grade of vagueness. Such a way, also the supports of these trapezoids have been univocally determined. On the other hand, trapezoids placed at the edges of the universe of discourse, i.e. associated to the classes *Zero* and *High*, have been automatically built, since the crossover points to their contiguous fuzzy sets have been previously fixed and the slopes of their vertical sides have been forcedly determined by the criteria of interpretability aforementioned.

The resulting collection of five distinct fuzzy sets is shown in Fig. 10, by exhibiting more graded transitions between two contiguous classes, with a semantics closer to human reasoning.

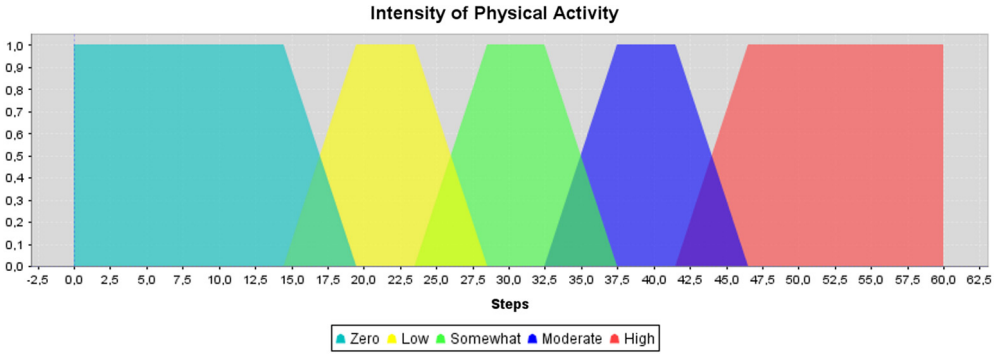


Fig. 10. The primitive context "physical activity" whose conceptual states are modeled as a collection of fuzzy sets.

5.3. Reasoning

The reasoning facilities offered by the application enable to process and correlate all the sensed or estimated data. To this aim, first of all, some specific technical solutions have been chosen for representing the information stored into the repositories of the Hybrid Reasoner, i.e. Tbox, Abox and Rbox. All the information stored in the first two boxes is codified as triples, each of them in the form *<subject, predicate, object>* , according to the N-Triples syntax (RDF 1.1 N-Triples, 2014). On the other hand, all the rules stored in the Rbox have been expressed by using the syntax outlined in Fig. 11.

In this syntax, three kinds of rule atoms have been defined, namely *Triple pattern* (TP), *Negated triple pattern* (nTP) and *Function to call* (FC).

A TP is an N-triple object containing both static values and variables, these latter indicated using the standard convention of prefixing them with a question mark. A TP can be used both in a CE, to test the existence of facts in the Abox matching it, and in an AE, to assert a fact in the Abox. A nTP is an N-triple object that can be used in a CE to test the absence of a fact in the Abox matching it. Finally, FCs are internal procedures invoked to evaluate logical conditions, compute arithmetic expressions or perform fuzzy evaluations, if inserted in the LHS of a rule, and assert and retract facts according to a non-monotonic strategy or a fuzzy inference scheme, if inserted in the RHS.

A particular type of FC is *FuzzyFunction* (FF), which enables the hybridization of the classic inference scheme with Fuzzy Logic. It can be used in the LHS of a rule in order to associate a fuzzy term to a variable with a smoothed degree of membership (*fuzzification*), whereas it can be used in the RHS of a rule to perform the fuzzy inference procedure (*fuzzy inference and aggregation*) and produce a precise value as output (*defuzzification*). Hybrid rules built by using FF are eventually grouped depending on the fuzzy variable used in their AE.

On the top of these choices, a specific algorithm for both pattern matching and rule firing has been defined and integrated in the Hybrid Reasoner, by extending the lazy evaluation approach proposed by the authors in Minutolo et al. (2015) and offering the additional functionality of supporting hybrid rules and, contextually, granting an efficient handling of memory and computational resources. In particular, this algorithm, referred as Algorithm 3, essentially consists in the following steps.

First, it starts the pattern matching phase by processing Tbox and Abox in order to detect novel elements not evaluated yet, and, successively, it verifies if they satisfy TPs or nTPs contained in the LHSs of both classical and hybrid rules stored in the RBox by calculating the so-called *intra-condition tests*. The results of such tests are elements matching TPs or nTPs, which are stored into ad-hoc memory structures. After filling these memories, Algorithm 3 processes classical and hybrid rules and labels them as *active* if all the memories linked to their TPs contain a matching element at least.

After generating the list of active rules, these latter are processed to determine the first eligible rule activation. Different situations can occur, i.e. an active rule can admit one or more activations. In particular, a rule admits one activation in case when its CEs do not contain variable references, and more activations in case when its CEs contain one or more variable references. In this last case, each activation is determined by calculating the so-called *inter-condition tests*, which visit the space of all possible variable bindings and choose the ones that do not violate any other CEs. Successively, a degree is assigned to each activation. It is equal to 1 for classical rules, where only TP and nTPs are evaluated, and can vary from 0 to 1 for hybrid rules, where also FFs are taken into account, first singularly, by calculating their degree of membership, and then together, by aggregating the different contributions and choosing their minimum as final output. Obviously, in case of degree equal to 0, the activation is discarded since ineligible. As soon as an eligible activation is found, the research of other possible ones is paused and, then, it is passed to the RAE for the execution.

At this point, Algorithm 3 starts the rule firing phase, which essentially proceeds in the following way. In case the activation to be executed is associated to a classical rule, the TP in its AE is executed. On the other hand, in case the activation is associated to a hybrid rule belonging to a group, it is considered as the main one and activations of other rules of the same group and having a coherent binding space are searched for. Indeed, a group of rules with the same fuzzy variable in their AE must be processed as a whole since all of them contribute to produce a new assertion about that variable, in accordance with the dictates of Fuzzy Logic in case of multiple levels of inference. For this reason, the FFs occurring in the AEs of all the coherent activations in a group are processed according to the classical fuzzy inference scheme, generating a precise defuzzified value. Thus, the TP present in the AE of the main activation is linked to this value and, then, executed.

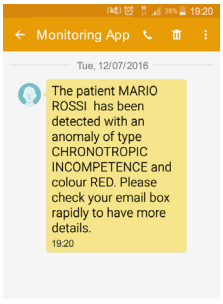
Finally, after the execution of an AE, possible updates to Tbox and Abox are timely produced, by changing also the memories associated to the CEs of involved rules. A formal description of the whole algorithm is reported in Algorithm 3.

Algorithm 3 has been experimentally tested in terms of overall response time and memory usage on a smartphone with the following characteristics: Samsung Galaxy S5, Android 4.4.2, 2 GB RAM, CPU Quad-Core Snapdragon-801 2.5 GHz. In particular, three different configurations of Rbox have been considered: (i) 30 classical rules; (ii) 5 groups of 6 hybrid rules; (iii) 12 classical rules and 3 groups of 6 hybrid rules. For each Rbox, a set of 10 experiments has been performed with a growing number of individuals populating both Tbox and Abox, varying from 1k to 10k triples. Moreover, the number of potential activations has been fixed equal to 10 and chosen for the three configurations of Rbox in the following way: (i) 5 activation for 2 classical rules; (ii) 2 activations for each group of hybrid rules; (iii) 2 activations for 2 classical rules and 2 activations for each group of hybrid rules. Each experiment has been executed 5 times for a total of 50 runs. The results achieved in terms of overall response time and memory usage have been calculated as the average obtained over the five repeated runs of each experiment. In particular, Algorithm 3 has produced timely responses in about 300–400 ms with a memory usage less than 150 KB. These performances are

```
Rule ::= Name: if Antecedents then Consequents;
Name ::= TextValue;
Antecedents ::= ConditionElement [conjunctionOperator ConditionElement];
ConditionElement ::= (N-Triple) | not(N-Triple) | FunctionToCall
Antecedents ::= ActionElement [conjunctionOperator ActionElement];
ActionElement ::= N-Triple | NonMonotonicFunction | FuzzyFunction ;
conjunctionOperator ::= , | &
N-Triple ::= Subject Predicate Object;
Subject ::= Term | Variable;
Predicate ::= Term | Variable;
Object ::= Term | Variable | Value;
Term ::= uri-reference (e.g. http://exuri#ex1) | prefix:name (e.g. expre:ex1);
Variable ::= ?variableName;
Value ::= NumericalValue | BooleanValue | TextValue
NumericalValue ::= number (e.g. 10 or 20.5);
BooleanValue ::= 'true' | 'false';
TextValue ::= 'a string';
```

```
FunctionToCall ::= LogicalFunction | ArithmeticalFunction | NonMonotonicFunction | FuzzyFunction;
LogicalFunction ::= equal(fArg, fArg) | notEqual(fArg, fArg) | lessThan(fArg, fArg) | lessEqual(fArg, fArg) | greaterThan(fArg, fArg) | greaterEqual(fArg, fArg);
fArg ::= Variable | Value;
ArithmeticalFunction ::= sum(fArg, fArg, Variable) | product(fArg, fArg, Variable) | difference(fArg, fArg, Variable) | quotient(fArg, fArg, Variable);
NonMonotonicFunction ::= newInstance(Variable, Term) | retract(N-Triple);
FuzzyFunction ::= IS(Variable, FuzzyVariableName, FuzzyTermName, FuzzyShape) | ISnot(Variable, FuzzyVariableName, FuzzyTermName, FuzzyShape);
FuzzyVariableName ::= TextValue;
FuzzyTermName ::= TextValue;
FuzzyShape ::= TriangularShape | TrapezoidalShape;
TriangularShape ::= NumericalValue, NumericalValue, NumericalValue;
TrapezoidalShape ::= NumericalValue, NumericalValue, NumericalValue, NumericalValue
```

Fig. 11. The rule syntax used for the Rbox.



Instant Report

Patient: Mario Rossi
Age: 50 Rest Heart Rate: 70 Maximum Heart Rate: 170
Date 12/07/2016
Number of Anomalies detected: 1

TIME	HEART RATE (bpm)	SPO_2	SKIN TEMPERATURE	STEPS	PHYSICAL ACTIVITY INTENSITY	ANOMALY NAME	ANOMALY TYPE	ANOMALY COLOUR
18:00	88	96%	33.4°C	27	Low/ Somewhat	-	-	-
18:20	70	97%	32.4°C	17	Zero/Low	-	-	-
18:40	69	98%	32.2°C	12	Zero/Low	-	-	-
19:00	78	96%	33.5°C	25	Low/ Somewhat	Chronotropic incompetence during low/ somewhat activity	Chronotropic incompetence	Yellow
19:20	79	96%	33.7°C	27	Low/ Somewhat	Chronotropic incompetence during low/ somewhat activity	Chronotropic incompetence	Red

Fig. 12. An example of text message and report generated when an anomaly is detected by the application.

perfectly coherent with health monitoring demands, with the additional benefit of preserving the available runtime memory of the device hosting the application.

5.4. Reporting and alerting

The reporting and alerting functionalities offered by the application allow contacting doctors and caregivers. To this aim, the application exploits the mechanisms offered by the architecture to generate reports at the end of a scheduled monitoring period or to trigger alerts when anomalies are detected.

An example of text message and report generated when an anomaly is detected and persists over time, i.e. at least twice consecutively, is depicted in Fig. 12. In particular, the text message contains only a short description of the anomaly detected, whereas the email contains the whole report, including of all the data sensed by the bracelet or estimated by means of the components of the architecture until the occurrence of the anomaly. Both the text message and report are automatically sent to the doctor and the caregiver by means of an SMS and an email, respectively.

5.5. Application-specific knowledge and data model

Application-specific knowledge and data have been represented as follows. First, the low-level ontology for the monitoring application has been formalized, consisting into the following individuals that statically populate the assertional part of the high-level ontology defined in the data model of the proposed architecture:

- *temperature*: **VitalParameter**
  - *name*: "Skin temperature"; *indirect*: false; *descr*: "This parameter measures the temperature of the skin of the subject".
- *heartRate*: **VitalParameter**
  - *name*: "Heart Rate"; *indirect*: true; *descr*: "This parameter measures the heart rate of the subject".
- *spO2*: **VitalParameter**
  - *name*: " Saturation of Peripheral Oxygen"; *indirect*: true; *descr*: "This parameter measures the O_2 saturation in the blood of the subject".
- *stepCount*: **ActivityParameter**
  - *name*: " Number of steps"; *indirect*: true; *descr*: "This parameter measures the number of steps walked by the subject".
- *bracelet*: **Sensor**

Algorithm 3:
Purpose: Pattern matching and rule firing.
Input:
$R_{box} = \{R_c, R_h\}$ /* the rulebox made of a set of classical rules R_c and hybrid rules R_h */
T_{box} /* the terminological box */
A_{box} /* the assertional box */
MEM /* ad-hoc memory structures */
Output:
T_{box} /* the terminological box updated after the execution of the rules in the rulebox */
A_{box} /* the assertional box updated after the execution of the rules in the rulebox */

1: $F_{new} \leftarrow \{\varnothing\}$ /* initialize the set containing novel facts not evaluated */
2: $R_{active} \leftarrow \{\varnothing\}$ /* initialize the set containing active rules */
3: **repeat** /* evaluate iteratively the set of active rules */
4: $F_{new} \leftarrow$ checkForUpdates(T_{box}, A_{box}) /* extract facts not yet evaluated from Tbox and Abox */
5: **if** ($F_{new} \neq \{\varnothing\}$) **then** /* verify if novel facts have been found */
6: $MEM \leftarrow$ IntraConditionTests(R_{box}, F_{new}) /* determine facts matching TPs or nTPs contained in the LHSs of rules in
7: the R_{box} and store them into memories */
8: $R_{active} \leftarrow$ getActiveRules(R_{box}, MEM) /* get active rules from the R_{box} if all the memories linked to their TPs
9: contain at least a matching element */
10: $i \leftarrow 1$ /* initialize the counter for moving on the set containing active rules */
11: **end if**
12: **if** ($R_{active} \neq \{\varnothing\}$) **then** /* verify if the set of active rules is empty */
13: $r_{active} \leftarrow R_{active}[i]$ /* get the i^{th} element from the set of active rules */
14: $[r_{activation}, r_{degree}] \leftarrow$ getFirstActivation(r_{active}, MEM) /* get the first eligible activation of the selected active rule and calculate
15: its degree of activation */
16: **if** ($r_{activation} \neq \varnothing$ and $r_{degree} > 0$) **then** /* verify if an eligible rule activation exists and its degree of activation
17: is greater than 0*/
18: **if** ($r_{active} \in R_c$) **then** /* verify if the active rule is a classical one */
19: $[T_{box}, A_{box}] \leftarrow$ execute ($r_{activation}$, r_{degree}) /* execute the rule activation and timely update Tbox and Abox */
20: **else if** ($r_{active} \in R_h$) **then** /* alternatively verify if the active rule is a hybrid one */
21: $G \leftarrow$ getGroup(r_{active}) /* determine the group the active rule belongs to*/
22: $[G_{activation}, G_{degree}] \leftarrow$ getGroupActivation($r_{activation}$, G, MEM) /* get activations of other rules belonging to the same
23: group and having a coherent binding space */
24: $[T_{box}, A_{box}] \leftarrow$ execute ($G_{activation}$, G_{degree}) /* execute the rule activation according to the classical fuzzy inference
25: scheme and timely update Tbox and Abox */
26: **end if**
27: **else** /* alternatively get another element from the set of active rules */
28: $i \leftarrow i + 1$ /* update the counter to move ahead of one active rule */
29: **end if**
30: **end if**
31: **until** ($i \leq$ length(R_{active})) /* repeat until all the active rules are evaluated*/

- *name*: "Bracelet"; *model*: "Amiigo"; *descr*: "This bracelet is a wristband sensor equipped with a 3-axis accelerometer, a temperature sensor and a reflectance-based pulse oximeter".

On the other hand, all the individuals of the classes **Subject** and **Measure** are instanced the first time the application is initialized and when it is dynamically utilized by the user, respectively.

Finally, all the properties related to the individuals of the class **Anomaly** are valued dynamically only when a specific abnormal situation is detected.

A collection of hybrid rules has been defined on the top of the above-mentioned individuals in order to detect bradycardia and tachycardia. These rules have been formulated in order to apply definitions of bradycardia and tachycardia in a personalized manner, by allowing the evaluation of each heart rate measure with reference to the specific subject, to his/her actual situation (such as after dinner) and to the intensity of physical activities just performed. In these rules, the value ranges used for the skin temperature at rest and during a physical activity have been elicited from Trinity et al. (2010). Moreover, the classical definitions of bradycardia and tachycardia have been used in case when the patient is at rest, whereas the definition of chronotropic incompetence, often defined as "sustained relative arrhythmia" (Brubaker and Kitzman, 2011), has been used to express an inadequate heart rate response during physical exercise. Finally, the relationships between hypoxia and bradycardia as well as between hypothermia, hyperthermia and cardiac arrhythmias have been also formalized as evidenced in Skinner (1997); Deussen (2007), respectively.

In more detail, with reference to the bradycardia, the rules expressed in natural language are reported in the following:

- *at rest*: the current measure of heart rate is detected as lower than 15% of the measure of heart rate at rest of the subject, having a skin temperature included between 30 °C and 32 °C, for at least two consecutive measurements;

```
[Rule1: if (?y age ?age), difference(220, ?age, ?mhr), product(0.50, ?mhr, ?mhr_050)
        (?y measure?m1), (?m1 parameter heartRate),(?m1 value ?v1), lessThan(?v1, ?rhr_050),
        (?y measure ?m2), (?m2 parameter temperature),(?m2 value ?v2), le(?v2, 37),
        (?y measure ?m3), (?m3 parameter stepCount), (?m3 value ?v3),
        IS(?v3, PhysicalActivity, Low, 14, 20, 23, 29), IS(?v3, PhysicalActivity, SomeWhat, 23 29, 32, 38),
        (?y anomaly ?a), (?a name 'Chronotropic incompetence during low/somewhat activity'^^xsd:string),
        (?a type 'Chronotropic incompetence'^^xsd:string)
      then (?z, colour, Yellow, 1), (?a colour ?z) ]

[Rule2: if (?y age ?age), difference(220, ?age, ?mhr), product(0.50, ?mhr, ?mhr_050)
        (?y measure?m1), (?m1 parameter heartRate),(?m1 value ?v1), lessThan(?v1, ?rhr_050),
        (?y measure ?m2), (?m2 parameter temperature),(?m2 value ?v2), le(?v2, 37),
        (?y measure ?m3), (?m3 parameter stepCount), (?m3 value ?v3),
        IS(?v3, PhysicalActivity, Low, 14, 20, 23, 29), IS(?v3, PhysicalActivity, SomeWhat, 23 29, 32, 38),
        (?y previousAnomaly ?a1), (?a1 name 'Chronotropic incompetence during low activity'^^xsd:string),
        (?a1 type 'Bradycardia'^^xsd:string) (?a1 colour 'Yellow'^^xsd:string)
        (?y anomaly ?a), (?a name 'Chronotropic incompetence during low/somewhat activity'^^xsd:string),
        (?a type 'Chronotropic incompetence'^^xsd:string)
      then (?z, colour, Red, 2), (?a colour ?z) ]
```

Fig. 13. Exemplary rules for detecting chronotropic incompetence during a lowly or somewhat intense physical activity.

- *due to hypothermia*: the current measure of heart rate is detected as lower than 15% of the measure of heart rate at rest of the subject, with a skin temperature lower than 28°C, for at least two consecutive measurements;
- *due to hypoxia*: the current measure of heart rate is detected as lower than 15% of the measure of heart rate at rest of the subject, with a SpO_2 lower than 92%, for at least two consecutive measurements.

On the other hand, the rules referred to the tachycardia are the following:

- *at rest*: the current measure of heart rate is detected as higher than 40% of the measure of heart rate at rest of the subject, having a skin temperature included between 30 °C and 32 °C, for at least two consecutive measurements;
- *due to hyperthermia*: the current measure of heart rate is detected as higher than 40% of the measure of heart rate at rest of the subject, with a skin temperature higher than 36°C, for at least two consecutive measurements.

Finally, the rules referred to the chronotropic incompetence are the following:

- *during a lowly or somewhat intense physical activity*: the current measure of heart rate is detected as lower than 50% of the maximum heart rate value of the subject (calculated as 220 — his/her age), having a skin temperature lower than 37 °C, for at least two consecutive measurements;
- *during a moderately or highly intense physical activity*: the current measure of heart rate is detected as lower than 70% of the maximum heart rate value of the subject (calculated as 220 — his/her age), having a skin temperature lower than 37 °C, for at least two consecutive measurements.

These rules have been codified according to the syntax outlined in Fig. 11 on the top of the high-level and domain ontologies defined. For the sake of brevity, only two exemplary rules are reported in Fig. 13, which detect, as above described, the chronotropic incompetence during a lowly or somewhat intense physical activity.

All these rules can be dynamically loaded into the monitoring application, customizing their behavior depending on the specific subject to be monitored, i.e. for instance, by modifying the thresholds used in them.

5.6. Graphical user interfaces

The main graphical user interfaces of the application are shown in Fig. 14. After launching the app, the user is asked to insert name and password, to uniquely identify and log him/her into the application contents (Fig. 14(a)). Registration is only required for the first access, where the user is asked to insert some personal information (Fig. 14(b)), such as name, age, height, weight, and gender and his/her caregiver(s) (Fig. 14(c)). Thereupon the user is logged and the application modules loaded, the main menu is available (Fig. 14(d)). By pushing the button "Monitoring", the user starts the monitoring procedure and gets access to some of its main functionalities (Fig. 14(e)). First, patient-specific knowledge and data model must be loaded (Fig. 14(f)), successively, the scan of available BLE devices is started and the pairing between the mobile device and the bracelet is performed (Fig. 14(g)). Finally, the user can start/stop the monitoring (Fig. 14(h)).

In conclusion, the usage of the architecture, as already envisioned during its conception, has implied a rapid prototyping of the whole application. In fact, most of the application development efforts have been mainly devoted to the design and realization of these graphical user interfaces, whereas the core functionalities above described has been easily released by exploiting the architecture.

6. Conclusions and future work

This paper has presented a smart mobile, self-configuring, context-aware architecture devised to enable the rapid prototyping of personal health monitoring applications for different scenarios, by exploiting both commercial wearable sensors and mobile devices. It has been organized as a composition of four tiers aimed at: (i) acquiring biomedical signals and accelerometer information pertaining to the patient; (ii) extracting vital and context cues; (iii) correlating them in order to detect possible anomalies, by means of personalized rules; (iv) communicating with caregivers and doctors by sending emails or text messages. A unified data model has been defined to enable data interoperability among the components of the architecture and the monitoring applications built on the top of them. This data model is codified in terms of a high-level ontology and more specific low-level ontologies built on the top of it so that the features of particular monitoring applications can be captured.

The proposed architecture has been implemented in Java and released to operate on the Android Operating System (OS). Successively, to prove its modifiability and evolvability, it has been evaluated by employing the ALMA method, highlighting its capability of being rapidly and effortlessly customized, personalized or eventually modified to prototype novel health monitoring applications. Finally, as a real

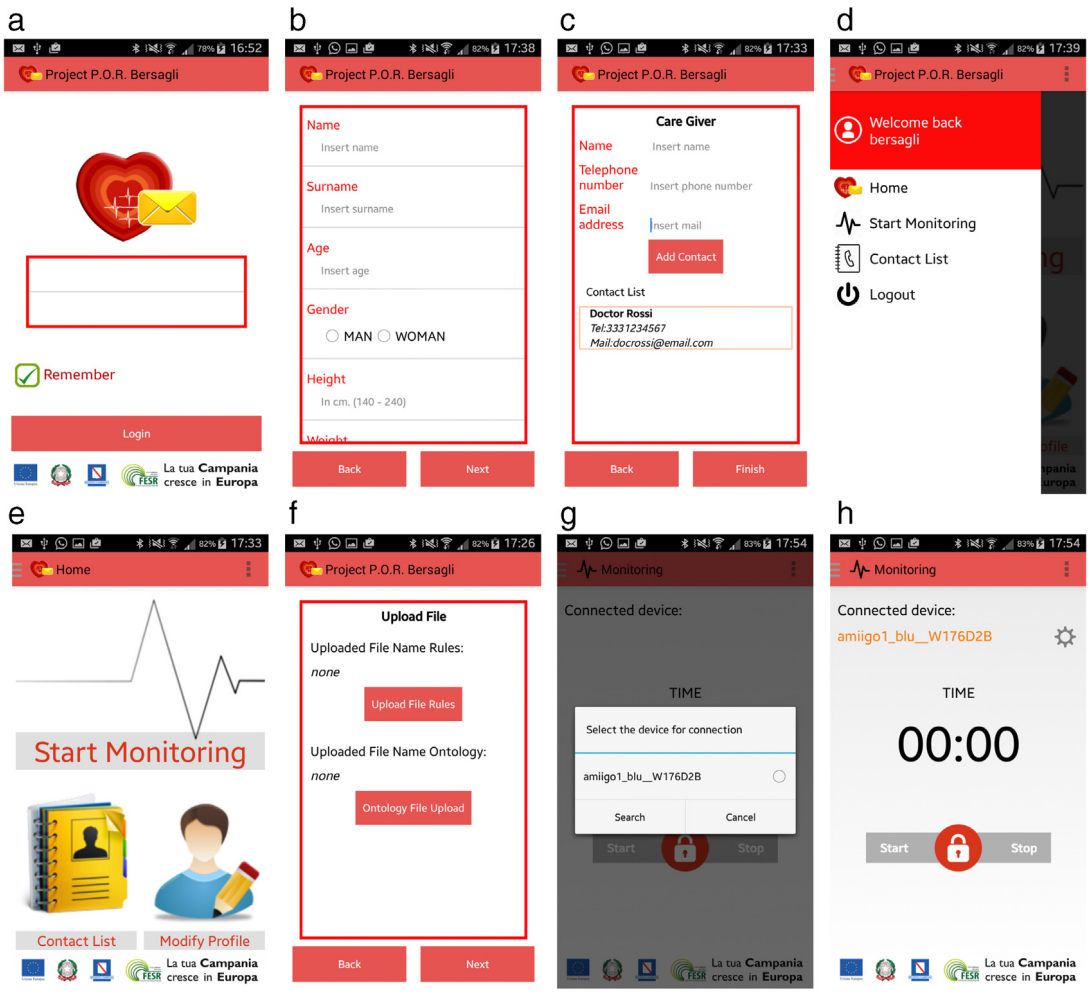


Fig. 14. The Login Interface (a), the Registration Interfaces (b, c), the Main Menu (d), the Main Monitoring Interface (e), the Configuration Interface (f), the Device Searching Interface (g), the Monitoring Control Interface (h).

case study, it has been used in the context of the Italian project "Bersagli" to realize a mobile application for monitoring and managing cardiac arrhythmias, such as bradycardia and tachycardia. Even if the requirements of this project have not enabled to take advantage of all the sensing facilities of the architecture (only one sensing device is used), all the other features offered by it have been fully exploited to build the application, confirming its effectiveness with respect to a real scenario.

The current version of the proposed architecture leaves room for some improvements. First of all, the architecture is planned to be extended with further Sensor Adapters for supporting smart watches, since their computational power and rich sensor interfaces together with their proximity to the human body make them a very promising means for continuously monitoring people's behaviors through collecting contextual data.

Secondly, the described architecture will be further improved in the future by means of deep learning algorithms to infer behaviors and contexts from sensor data collected by mobile devices. In particular, algorithms capable of performing a variety of sensor inference tasks directly on the mobile device will be investigated, assuming to train deep models in an offline manner with conventional tools. Finally, further development will also address issues tied to the integration of some mechanisms for security and privacy enforcement as well as Quality of Service provision.

Acknowledgments

This work has been partially supported by the Italian project "Bersagli" funded by POR FESR 2007–2013 — Regione Campania. The authors are grateful to Neatec S.p.A. for its collaboration in developing the mobile application for testing the proposed architecture.

References

Agarwal, A., Furht, B., Conatser, M., Baechle, C., 2013. Secure mobile framework for monitoring medical sensor data. In: Handbook of Medical and Healthcare Technologies. pp. 355–369.

Banos, O., Villalonga, C., Damas, M., Gloesekoetter, P., Pomares, H., Rojas, I., 2014. Physiodroid: Combining wearable health sensors and mobile devices for a ubiquitous, continuous, and personal monitoring. Sci. World J.

Bengtsson, P., Lassing, N., Bosch, J., van Vliet, H., 2004. Architecture-level modifiability analysis (ALMA). J. Syst. Softw. 69 (1), 129–147.

Brubaker, P.H., Kitzman, D.W., 2011. Chronotropic incompetence causes, consequences, and management. Circulation 123 (9), 1010–1020.

Chan, M., Estève, D., Fourniols, J.Y., Escriba, C., Campo, E., 2012. Smart wearable systems: Current status and future challenges. Artif. Intell. Med. 56 (3), 137–156.

Clear, A.K., Dobson, S., Nixon, P., 2007. An approach to dealing with uncertainty in context-aware pervasive systems. In: UK/IE IEEE SMC Cybernetic Systems Conference.

Deussen, A., 2007. Hyperthermia and hypothermia. Effects on the cardiovascular system. Anaesthesist. 56 (9), 907–911.

Evensen, P., Meling, H., 2009. Sensor virtualization with self-configuration and flexible interactions. In: Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems, pp. 31–38.

Fanucci, L., Donati, M., Celli, A., Spingola, G., Aragno, C., Cristiano, L., Spingola, G., Ottonelli, F., Ghilardi, M., 2015. Advanced multi-sensor platform for chronic disease home monitoring. In: Instrumentation and Measurement Technology Conference (I2MTC), 2015 IEEE International, pp. 646- 651.

Forkan, A., Khalil, I., Tari, Z., 2014. CoCaMAAL: A cloud-oriented context-aware middleware in ambient assisted living. Future Gener. Comput. Syst. 35, 114–127.

Gacto, M.J., Alcalá, R., Herrera, F., 2011. Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures. Inform. Sci. 181 (20), 4340–4360.

Gay, V., Leijdekkers, P., 2007. A health monitoring system using smart phones and wearable sensors. Int. J. ARM 8 (2), 29–35.

Giffen, R., Fung, M.F.K., Rotaru, M., 2015. Remote patient monitoring to improve health: challenges and opportunities. In: Proceedings of the 25th Annual International Conference on Computer Science and Software Engineering. IBM Corp., pp. 277–280.

Gruber, T., 1995. Towards principles for the design of ontologies used for knowledge sharing. Int. J. Hum. Comput. Stud. 43, 907–928.

Guarino, N., 1995. Formal ontology, conceptual analysis and knowledge representation. Int. J. Hum.-Comput. Stud. 43, 625–640.

Guarino, N., 1997. Understanding and building, using ontologies. Int. J. Hum.- Comput. Stud. 46, 293–310.

Habetha, J., 2006. The MyHeart project-fighting cardiovascular diseases by prevention and early diagnosis. In: Conf Proc IEEE Eng Med Biol Soc, pp. 6746–6749.

Henry, J.E., Cain, J.P., 1997. A quantitative comparison of perfective and corrective software maintenance. J. Softw. Maint.: Res. Pract. 9 (5), 281–297.

Hudson, L.D., 1985. Design of the intensive care unit from a monitoring point of view. Respir. Care 30 (7), 549–559.

Jones, V., van Halteren, A., Dokovsky, N., Koprinkov, G., Bults, R., Konstantas, D., Herzog, R., 2006. Mobihealth: Mobile Health Services Based on Body Area Networks. pp. 219–236.

Khalifeh, A., Obermaisser, R., Abou-Tair, D.E.D.I., Abuteir, M., 2014. Systems-of-systems framework for providing real-time patient monitoring and care. In: Proceedings of the 8th International Conference on Pervasive Computing Technologies for Healthcare, pp. 426–429.

Lamprinakos, G.C., Asanin, S., Broden, T., Prestileo, A., Fursse, J., Papadopoulos, K.A., Kaklamani, D.I., Venieris, I.S., 2015. An integrated remote monitoring platform towards telehealth and telecare services interoperability. Inform. Sci. 308, 23–37.

Li, X., Eckert, M., Martinez, J.F., Rubio, G., 2015. Context aware middleware architectures: Survey and challenges. Sensors 15 (8), 20570–20607.

López-de Ipiña, D., Blanco, S., Laiseca, X., Díaz-de Sarralde, I., 2011. ElderCare: an interactive TV-based ambient assisted living platform. In: Activity Recognition in Pervasive Intelligent Environments, pp. 111–125.

Marques, F.A.F., Ribeiro, D., Colunas, M.F., Cunha, J.P.S., 2011. A real time, wearable ECG and blood pressure monitoring system. In: 2011 6th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1–4.

Mei, H., Widya, I., Van Halteren, A., Erfianto, B., 2006. A flexible vital sign representation framework for mobile healthcare. In: Pervasive Health Conference and Workshops, pp. 1–9.

Minutolo, A., Esposito, M., De Pietro, G., 2015. Design and validation of a light-weight reasoning system to support remote health monitoring applications. Eng. Appl. Artif. Intell. 41, 232–248.

Musen, M.A., 2015. The protégé project: A look back and a look forward. AI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence 1 (4), 4–12.

Oner, M., Pulcifer-Stump, J.A., Seeling, P., Kaya, T., 2012. Towards the run and walk activity classification through step detection-An android application. In: Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE, pp. 1980–1983.

Paradiso, R., Loriga, G., Taccini, N., 2004. Wearable system for vital signs monitoring. Stud. Health Technol. Inform. 108, 253–259.

Patel, S., Park, H., Bonato, P., Chan, L., Rodgers, M., 2012. A review of wearable sensors and systems with application in rehabilitation. J. Neuroeng. Rehabil. 9 (1), 1.

RDF 1.1 N-Triples, 2014. A line-based syntax for an RDF graph W3C Recommendation, https://www.w3.org/TR/n-triples/.

Rienzo, M.D., Rizzo, F., Meriggi, P., Bordoni, B., Brambilla, G., Ferratini, M., Castiglioni, P., 2006. Applications of a textile-based wearable system for vital signs monitoring. In: Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE, pp. 2223–2226.

Roy, N., Pallapa, G., Das, S.K., 2007. A middleware framework for ambiguous context mediation in smart healthcare application. In: Wireless and Mobile Computing, Networking and Communications, 2007. WiMOB 2007. Third IEEE International Conference on, pp. 72–72.

Ruiz-Zafra, Á., Benghazi, K., Noguera, M., Garrido, J.L., 2013. Zappa: An open mobile platform to build cloud-based m-health systems. In: Ambient Intelligence-Software and Applications. pp. 87–94.

Serhani, M.A., El Menshawy, M., Benharref, A., 2016. SME2EM: Smart mobile end-to-end monitoring architecture for life-long diseases. Comput. Biol. Med. 68, 137–154.

Skinner, D.V., 1997. Cambridge Textbook of Accident and Emergency Medicine. Cambridge University Press.

Trinity, J.D., Pahnke, M.D., Lee, J.F., Coyle, E.F., 2010. Interaction of hyperthermia and heart rate on stroke volume during prolonged exercise. J. Appl. Physiol. 109 (3), 745–751.

Tudor-Locke, C., Burkett, L., Reis, J.P., Ainsworth, B.E., Macera, C.A., Wilson, D.K., 2005. How many days of pedometer monitoring predict weekly physical activity in adults?. Prev. Med. 40 (3), 293–298.

Uschold, M., Gruninger, M., 1996. Ontologies: principles, methods and applications. Knowl. Eng. Rev. 11 (2), 93–136.

Villarreal, V., Fontecha, J., Hervas, R., Bravo, J., 2014. Mobile and ubiquitous architecture for the medical control of chronic diseases through the use of intelligent devices: using the architecture for patients with diabetes. Future Gener. Comput. Syst. 34, 161–175.

World Health Organization, 2002. Active Ageing: A Policy Framework: A Contribution of the Second United Nations World Assembly on Ageing. World Health Organization, Geneva.

Zadeh, L.A., 1965. Fuzzy sets. Inf. Control 8, 338–353.