Applying Ontology to WBAN for mobile application in the context of sport exercises

Lina Nachabe
University of Telecom Sud Paris
Lebanon, Koura
961 3 320587
Lina_nachabe@yahoo.com

Marc Girod-Genet
University of Telecom Sud Paris
France, Paris
33 6 76923511
Marc.girod_genet@telecom-sudparis.eu

Bachar El Hassan, Fadi Aro
Lebanese University
Lebanon, Koura
961 3 538420
Bachar_elhassan@ul.edu.lb

ABSTRACT

The increasing use of wireless networks and the constant miniaturization of electrical devices have empowered the development of Wireless Body Area Networks (WBANs). In these networks various sensors are attached on clothing or on the body or even implanted inside the body. The wireless nature of the network and the wide variety of sensors lead to the birth of practical and innovative applications to improve health care and the quality of life. While exercising is becoming primordial in our daily life, various mobile applications were developed for runners capturing vital signals from sensors in order to calculate the number of burned calories. However, these applications still lack of compatibility and interoperability with other systems. This paper proposes an Android mobile application for calories burned during running exercise. This mobile application is fully relying on a dedicated and simplified WSN generic ontology that is introduced for overcoming the compatibility and interoperability issues aforementioned. This dedicated ontology is presented in this paper and some of its benefits are pointed out through the mobile application evaluation.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Language Constructs and Features – *abstract data types, control structures.*

General Terms

Design, Experimentation, Verification.

Keywords

WBAN, sensors, Ontology, semantic, Reasoning, Inference, m-health, mobile application, runner.

BODYNETS 2014, September 29-October 01, London, Great Britain

DOI 10.4108/icst.bodynets.2014.258224

1. INTRODUCTION

Obesity is becoming a severe problem in the societies. Based on the statistics done by the World Health Organization [1], 35% of adults aged 20 and over were overweight in 2008, and 11% were obese. Thus, it suggests that everyone should engage in regular physical activity (60 minutes a day for children and 150 minutes per week for adults). Running is one of the profitable exercises that helps controlling weight, combatting health diseases, improving moods and boosting energy.

On the one hand, the spectacular evolution in wireless technologies (Bluetooth, Wi-Fi, 3G, 4G etc…), and on the other hand, the integration of sensors within the mobile and wearable devices have made the concept of m-health emerged. A typical m-health system consists of sensors integrated within the mobile device and/or sensors embedded in wearable devices equipped with wireless technologies [2]. The sensors will capture biomedical data and send it to the mobile, smart phone or PDA which treats and displays the data. In almost all cases, the mobile will send the data to a remote medical server for further treatments. In addition, it can request data from the server and receive notifications in case of urgent cases.

Thus, various mobile applications were proposed in order to encourage running exercises [3]. Where some applications used only sensed data from the mobile in order to calculate burned calories and provide location's tracking, other applications served of the smart heart rate sensors equipped with Bluetooth capability to calculate the burned calories.

However, these applications lack of interoperability, information reusability, and extensibility. That's why this paper details a new mobile application for runners that calculates the burned calories based on a dedicated ontology. Some of the main goals for using this ontology are: the interoperability management, the ease of information management and the minimization of the battery consumption of the mobile device by requiring less network resources and data processing due to the use of ontologies.

This paper is divided as follow. Section 2 will provide a description of the considered WBAN's system architecture and use case. Section 3 will present the benefits of using ontologies. Then, section 4 will detail our proposed technical architecture and application. And finally, section 5 will present our evaluation and validation results.

2. WBAN SYSTEM ARCHITECTURE

A Wireless Body Area Network (WBAN) consists of tiny sensors/actuators attached on clothing, or on the body or even under the skin or inside the body. In general, these nodes are equally important and only used when needed in the application. These nodes could be mobile and the network topology could change frequently (high dynamicity). Those sensors/actuators are directly connected to the Sink or the gateway of the WBAN [4]. Within the WBAN, the data could be transmitted using Bluetooth, or other lower power wireless communication protocols.

The sink, also called Body Control Unit (BCU) gathers the information form the sensors, setups and controls the WBAN, provides graphical or audio interface to the user, and transfers the information to an external medical server. The BCU is the PDA or the mobile of the users. Once it has access to the internet, the BCU transmits data to the monitoring and control server(s). It can keep track of the sensed data, do some basic treatments and alert the users in urgent cases. The medical servers authenticate the user, accept health monitoring session uploads and format, insert this session data into corresponding medical records, analyze the data patterns, and finally identify serious health anomalies. A server agent may inspect the uploaded data and create an alert in the case of a potential risky medical condition.

Figure 1 shows the general architecture of m-health applications.

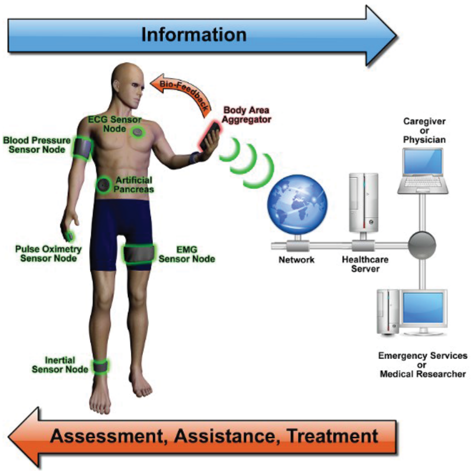

Figure 1-General Architecture of m-health applications.

Based on the general architecture, we can see that WBANs are characterized by their heterogeneity in terms of nodes, available resources, computing power, sensed data, wireless technologies, and medical application.

One additional crucial issue in m-health applications is the power consumption. Thus we need to reduce the power consumed by the nodes and extends the battery lifetime of the BCU which is the main brain of the system.

In order to minimize the power consumption in the BCU, we have to offer the main functionality of the application with the minimum data storage and processing. Here came the role of inference and reasoning where additional data and facts could be derived from basic data. Consequently, in the last decade, ontologies have widely used for modeling information in mobile computing domains [5].

3. USAGE OF ONTOLOGIES

T. Gruber defines the ontology that it is a specification of a conceptualization [6]. It describes the domain and the relationships between the concepts, properties, attributes, and individuals. Fortunately, the creation of ontology allows the explicit assumption of the domain enabling the reusability of the knowledge to avoid re-inventing the wheel, and provide a shared platform understandable by machines and humans. Thus, it permits the data heterogeneity management, the information reusability and the interoperability between different systems. In that way, instead of building each m-health system as independent application, the use of a common ontology allow the reuse of the predefined platform in order to customize the application as needed. Moreover, inference rules can be added to the description of the ontology allowing the deduction of advanced data from basic data. The efficiency will then be improved in proportional to the number of inference rules.

The most used and well known language to describe ontologies is Ontology Web Language (OWL) proposed as standard by W3C's Web Ontology Working Group in 2004 [7]. Its latest version is OWL 2 published in 2009 [8]. In our application, we used the Protégé 4.3 for defining the ontology model and the Jena API [9] for implementing the ontology on the mobile application and the server. Rules are added to the ontology using Semantic web Rule Language SWRL [10]. The Pellet reasoner is used in order to extract inferred information form the ontology [11]. SPARQL is dedicated for querying knowledge from the ontology [12].

4. ONTORUN MOBILE APPLICATION

This section is dedicated for the description of our "OntoRun" Mobile application.

4.1 GENERAL DESCRIPTION

In order to use OntoRun application, the user will be equipped with polar H7 heart rate sensor [13] shown in figure 2, that sends real time and accurate heart rate data via Bluetooth 4.0.

Figure 2-H7 Polar Heart Rate Sensor

OntoRun architecture is therefore formed of one network for each user. The network consists of two clusters, the H7 cluster, and the mobile cluster. Both clusters are using the Mobile as a sink in order to send sensed data to the web server using Wi-Fi or 3G connection. For location tracking, the GPS in the mobile is used. In addition, the speed is calculated, based on the accelerometer built in the mobile application.

Once the user downloads the mobile application, he will be asked to sign in using an existing Gmail account. The authentication will be done by the Google server built on the top of the Google plus API.

It is the time now for the user to fill his profile: weight, height, date of birth and gender. These parameters will be used to calculate the burned calories and alert the user if his heart rate exceeds the maximum limit.

The user can then update his profile, see the previous reports, start a new task by specifying the distance he wants to achieve, and check his neighbors.

Once the task is terminated, the number of burned calories will be displayed, in addition to a map showing the route. During exercising, if the heart rate exceeds the limit, a notification will be displayed in the mobile in order to alert the runner. Figure 3 shows the flow chart of the general description for OntoRun application.

4.2 TECHNICAL ARCHITECTURE

We have designed and instantiated a simplified WSN ontology directly derived from both our considered WBAN architecture (cf. section 4.1).

To offer extensibility and lightness, we have section 2) and our OntoRun Mobile Application specific context divided our simplified WSN ontology into three parts: WSN, Nodes, process and data.

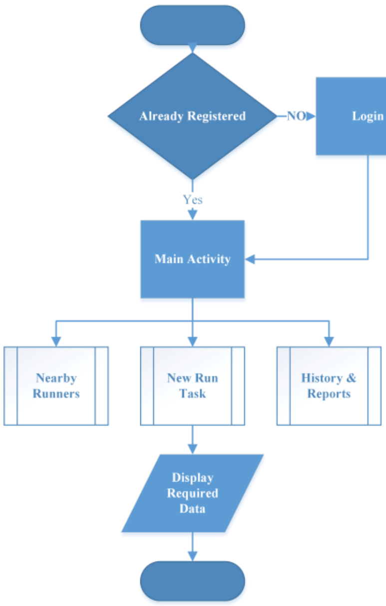

Figure 3-Flow Chart of OntoRun Application

Therefore, our WSN ontology contains the data properties describing the network, in addition to the cluster's characteristics. The use of a WSN as independent class/object paves the way to a new step toward network cooperation where each WSN is formed of clusters, or contains nodes, each cluster being formed of nodes and elect a sink.

In our ontology, a node can be a sensor, an actuator a sink or simply a node (for example router). To enhance the remote management of the WBAN, we are keeping track of the processor, the modes, the interfaces, the energy characteristics and the current mode for each node. Thus, if we want to send a notification from the server, it will be more efficient to send it when the node is in receiving or transmitting mode.

Each node is used in certain process (Heart rate, speed, or location). The process has a unit of measurement, delay, drift and other properties. Each process has data type, data format, data constraint (legal constraint, validity constraint, security constraints…). Each data has many measurements captured by one sensor. Furthermore, we are adding the value, the timestamp and the TTL for each measurement in order to guarantee the freshness of the data. Figure 4 summarizes the UML class diagram fully describing our simplified WSN ontology.

After understanding the main components of the used ontology, let's see in the next sub-section how we are using the aforementioned ontology in our application.

4.3 "ONTORUN": AN ONTOLOGY DRIVEN APPLICATION

First, when the user installs the application and signs in, the mobile will send using JSON the following information to the server: Gmail account, profile's parameters, mobile's MAC address and mobile type. Using the Jena API over a java framework for building semantic web application, the server will create a new individual of type WSN identified by the WIFI MAC address. This WSN' individual is having the runner identified by his Gmail account as contact. In addition the server will create a new cluster "Mobile", identified by the MAC address of the mobile and belonging to the created WSN. Having the WSNID and the ClusterID as functional properties assures that only one Gmail account could be assigned to the application. In that way, the user can only install one version into his mobile.

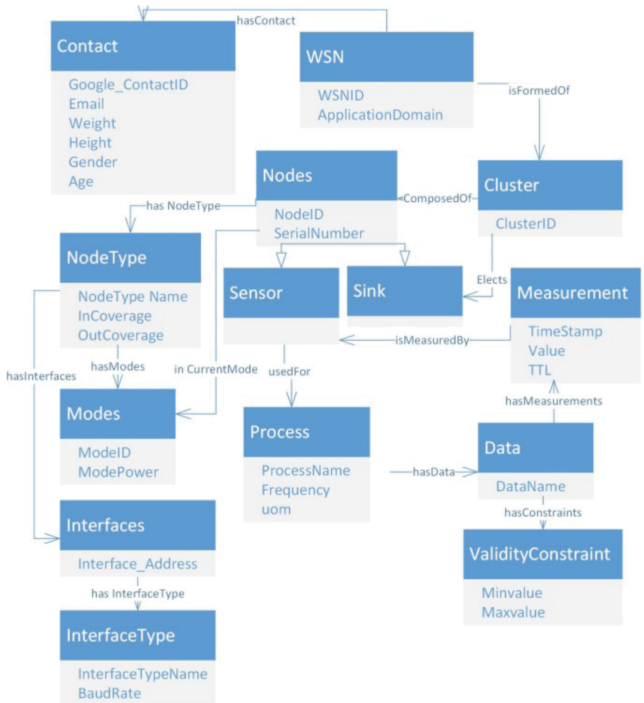

Figure 4-WSN UML Class Diagram

Furthermore, the mobile's cluster will be formed of the GPS Sensor and the Mobile as Sink. The Mobile Cluster will elect the Mobile Node to be the Sink. Figure 5 shows the relations between WSN, cluster and nodes ontology's classes/objects that are created on the server side.

One of the main goals of the ontology usage is to infer all the properties related to the mobile type. This is done by creating new node type, (for example Nexus Android device) and by adding that the mobile has this node type as type.

Using the reasoner, we can deduce the processor's characteristics, the interfaces (Bluetooth, WIFI, etc.), the in coverage distance, the out coverage distance, and other attributes. We introduce the interface type and node type classes in order to allow the users to re-use any required data related to the mobile or sensor properties, without the need to recreate all the features. The interface Type will describe the protocol used. For example, in the case of the H7, polar the type is Bluetooth 4.0.

Furthermore, when the user turns on the H7 Polar and the mobile receives the synchronization message via Bluetooth, it will send to the server that the H7 Polar is used in order to measure the heart rate. The server will then create a new individual H7 of type Cluster that elects the Mobile as Sink. Due to the use of a specific rule described in SWRL language, the server will automatically infer that this H7 cluster belongs to the runner's WSN because it elects the same sink that the one elected by the Mobile cluster(the mobile cluster belongs to the runner's WSN). Additionally, the server can deduce the owner of the H7 polar identified by the Gmail account predefined in the profile. H7 cluster will be formed of the heart rate sensor used to measure the heart rate in beats per minute.

All the previous individuals are created and inferred on the server side mainly for two reasons. The first one is to minimize the resources usage on the mobile, in terms of memory and processing, which lead to a minimal power consumption. The second one is the cooperation between different WSNs belonging to different runners. Consequently, the server could discover the neighbors of the runners while exercising. Furthermore, it can calculate the percentage of runners in certain regions, or deduce the peak time for exercising. In other terms, the server can calculate medical or social statistics for further studies or applications.

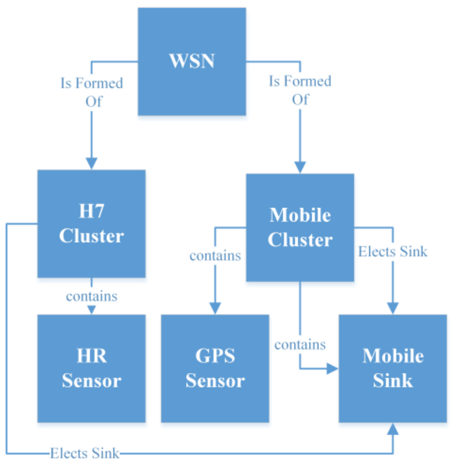

Figure 5-WSN, Cluster and Nodes OntoRun hierarchy

Let's pass now to the process, once the user selects a new task and sets the distance, the mobile updates the current mode of the Mobile Sink to active. The server can infer that the GPS and H7 sensor are in 'active' modes because the Sink elected by the cluster they are belonging to is active. The mobile then sends its current location to the server. The GPS sensor will measure the latitude and longitude. These values, associated with a timestamp, will be saved in the mobile ontology in addition to the heart rate.

The H7 is characterized by the node discovery time saved in the ontology. So, if the mobile try to create a heart rate measurement before this time, the reasoner will generate an inconsistency error indicating that it is an invalid value.
Afterwards, based on the runner's age, the reasoner will infer the maximum heart rate and triggers a notification if the measured heart rate will exceed this limit. The maximum rate is calculated using the following SWRL rule:

WSN(?W), composedOf(?C, ?N), hasData(ECG, ?D), hasContact(?W, ?P), isFormedOf(?W, ?C), hasConstraint(?D, ?V), usedFor(?N, ECG), Age(?P, ?age), multiply(?y, ?age, 0.7), subtract(?x, 208, ?y) -> MaxValue(?V, ?x)

Once this task is done, the mobile will display the burned calories, in addition to the route, and send its current location to the server. Figure 6 depicts the new task process workflow.

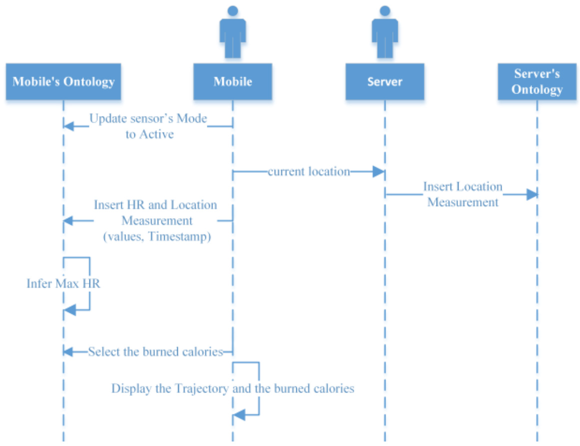

Figure 6-New Task Process workflow

A rule is defined in the server side in order to infer the location of the runner from the measurements done by the GPS sensor. This rule is of the form: *WSN(?W), composedOf(?C, ?S), measures(?S, ?M), hasData(Location, Longtitude), hasMesurement(Longtitude, ?M), isFormedOf(?W, ?C), hasLongtitude(?W, ?Q), usedFor(?S, Location), value(?M, ?x) -> hasvalue(?Q, ?x).*

In that way, once the user ask for his neighbors, the server run a SPARQL query to retrieve the Gmail accounts of the active runners in the requested runner's region.

Once the user quits the application, it informs the server which will update the current mode of the mobile to passive mode. Hence, when calculating the runner's neighbor only runners having their nodes in active modes will be selected.

5. TEST & RESULT

The application was tested on many users wearing the H7 Polar belt and others without H7 polar and using different android devices having KitKat version in order to support Bluetooth LE interfaces. The tracking system accuracy depends on the region where the application where tested. For users wearing H7 polar the number of burned calories were very accurate because the calculation is based on the heart rate measurements. However, for users without H7 Polar, the number of calculated burned calories precision is influenced by runner's speed. This inconvenience comes from the fact that we are calculating the speed from the GPS and that the burned calories depends on the speed. Thus, we are working on calculating the instant speed from other sensors built in the mobile device (accelerometer, gyroscope or pedometer). From the tests, we point out that the neighbors were successfully discovered and positively reflected on the application's feedback. The scan options permit the user to detect the heart rate without the need of setting up or adding any other information. This application can be extended to any sensor equipped with Bluetooth LE interface, since it is designed using the GATT profiles offered by the Bluetooth LE that permits the auto discovery of the devices in addition to the process used (Heart rate, glucose level, blood pressure and others) [14].The notification mechanism was effective and encourages the users to accelerate their speed without any fear. Figure 7 and Figure 8 shows the sign in process using Google plus, the reports for the done tasks, the nearby runners and the starting of a new task in OntoRun application.

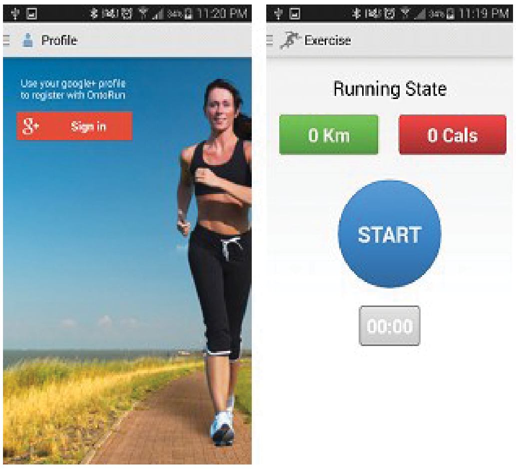

Figure 7-Signing in and starting a new task in OntoRun

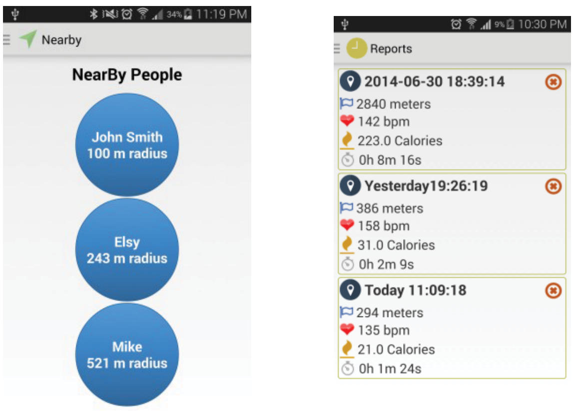

Figure 8-Reports & nearby runners

6. BEYOND ONTORUN

Statistics show that more than 97,000 health and fitness related mobile apps currently exist on Google Play and Apple App Store and that 4 million downloads are done per day [15].This assures that m-health applications are growing tremendously. Almost all these applications require user information, and especially user's medical profile. It would be a hassle if each service requested and managed the required user information on its own and if the user would have to provide the required personal data to each single service. This would not only be a burden to the user during the instantiation phase but also later for the management and modification of existing user data. Hence, it would be very beneficial from the user's point of view if these services could share the same set of user data. This can be achieved through the use of our proposed ontology model. Where services can share medical data, query for medical statistics or any required data, and develop its own application. For example, instead of entering the age, the weight, the height or any additional biomedical data, the user will be able to fill his profile once he creates an account, and all m-health applications can retrieve these data from the open ontology created on the cloud services. Moreover, these semantic data can be used by hospitals to identify a patient and retrieve necessary information like the maximum heart rate, the mass body, the allergies, etc.

The ability for health information to be shared among devices and enterprise health systems will improve the patient safety where the cooperation between the different applications can refine the predication of certain medical anomalies and enhance the user's life style. Here comes the role of the reasoner on the server side, where it can combine data coming from different applications and infer severe problems or malfunctioning in order to notify the user or the responsible party.

Where OntoRun application focuses on health entertainment, the OntoRun ontology can be used in other biomedical scenarios where continuous remote monitoring of the patient is needed. In some cases, some acts should be taken to regulate the dose of medicine or the dose of insulin using the actuator. Due to the usage of ontologies and the unified resource identifier (URI), these sensors and actuators are accessible by internet servers that can manage individual health and thus lowering the health costs.

Among this enormous number of applications, the major challenge is the power consumption. Power- hungry applications will lose the market very soon. The big question is how to minimize this power consumption. Experiences show that one of the main energy consumers on mobile devices is the network interfaces. Accordingly, we should minimize the data sent to the backend server. But in the same time, we cannot overload the mobile by complex data treatment because it also affects dramatically on the battery consumption.

Due to the use of the inference model within the mobile application, the mobile can infer additional data without the need to save it in the database. To clarify this point, let's go back to OntoRun application. Once the mobile detects the presence of H7 Polar sensor, it will only create in the ontology a heart rate sensor that elects as sink the mobile that received the data. The reasoner can infer all the runner's information, the maximum heart rate, the gateway interface where it can access the internet and others, by using SWRL rules SPARQL to query information form the mobile ontology. This will minimize the saved data and the CPU usage. In OntoRun only the measurements are saved on a local database for calculation purposes.

On the server side, where the overall ontology is created, the opportunity to add rules and infer additional knowledge is really large. This convenience allows the mobile to send only basic data to the server once it has a 3G or WIFI connection. In fact, the calories calculation and the heart rate notification are done on the mobile side without the need to send any data to the server. In that way, network resources are only used in special cases where the mobile cannot infer it. Since the primary source of mobile's battery usage is the network resources, reducing sending data via network interfaces will reduce the battery power consumption. In the heart of this purpose, the 'NodeType' class was created. Where the mobile will just inform that the heart rate sensor is of type H7 Polar, the server can infer the baud rate, the power consumed during the transmission and all other characteristics related to the Bluetooth BLE and H7 Polar node type.

The highest network consumption in OntoRun will be once a new user is created in order to permit to the server to create the ontology explained in figure 5. But during a new task, only the actual mode of the mobile will be send to the server which is enormously reducing the data transfer between the mobile and the server.

Moreover, saving the current mode of each sensor will help the server to decide when to send optional data like advertisement or news, without consuming more battery on the mobile side. It is preferable to send new updates or notifications during a running task because the mobile is already in the active mode and ready to receive data.

In addition, the CPU processing is decreased due to the additional rules defined in our ontology allowing the mobile to calculate the burned calories with one SPARQL query.Here we should add that the ontology is light enough to be treated within the mobile, and the time spent by the reasoner to infer the model is smaller than if we are using a local database on the mobile where many tables and relations should be created to retrieve data.

7. CONCLUSION

In the paper, we have presented a new ontology-based application that has been used for calculating the burned calories and showing the trajectory of a runner during exercises. In addition, the runner can see the history of his previous tasks and check the nearby runners. The key point was the use of a modular ontology model created on the mobile and on the server in a way to enable interoperability between applications and to curtail the power consumption on the mobile side. Due to the SWRL rules incorporated within our model, data can be inferred and treated more efficiently. SPARQL queries have also permitted the runner's neighbors selection with minimum processing time. The use of our ontology model enhances the cooperation between different runners and creates new opportunities in m-health applications. The application is written in a way that can be used with any other Bluetooth LE devices, and the calories calculation can be done for any type of sport exercises (walking, biking, etc.), where only a constant in the formula can be modified.

We are also trying to optimize the inference model and the ontology architecture and to estimate the power consumption based on the task duration. New features will be added to the application for extending its usage to nutrition and diet. Social media will also be incorporated within the application for more interactivity using the Google plus features already incorporated in the existing application version.

Finally, m-health is spreading rapidly. What is becoming crucial is to create a common platform to ensure heterogeneity management and reusability of health data, as well as interoperability between different devices/applications, and extensibility without the need to rebuild all the application.

Well-constructed ontologies attend not only to make it easier to reuse existing knowledge in new settings, but also as foundation for standardization initiatives.

Definitely, we should use ontology models in m-health to competently handle all the progress in smart phones and medical services, as pointed out within this paper.

REFERENCES

[1] World Health Organization, 20014. Obesity and overweight. Fact Sheet Number 311 (May 2014).DOI= http://www.who.int/mediacentre/factsheets/fs311/en/

[2] Gay, V., Leijdekkers, P. AND Jones, V. 2010.Body sensor networks for Mobile Health Monitoring: experience in Europe and Australia. *Proceedings of the Fourth International Conference on Digital Society, ICDS 2010* (February 10-16, 2010 - St.Maarten, Netherlands Antilles), CPS 2010.

[3] 2014.10 Running Apps for 2013: Android, iPhone, Windows Phone. January 15-2014. DOI= http:// /www.runningshoesguru.com/2013/01/10-running-apps-for-2013-android-iphone-windows-phone/

[4] Otto, C., Jovanov, E., and Milenkovic, A. 2010.A WBAN-based System for Health Monitoring at Home. In *Proceedings of 3rd IEEE/EMBS International Summer School Conference, Medical Devices and Biosensors.* (Cambridge MA, USA, September 4-6 2006). DOI=10.1109/ISSMDBS.2006.360087.

[5] Prestes, R., Carvalho, G., Paes, R., Lucena, C. and Endler, M. 2004. *Applying ontologies in Mobile Application Applying Ontologies in Open Mobile Systems.*

[6] Gruber, T. 1992. *A Translation Approach to Portable Ontology Specifications*. Technical Report. University of Stanford, California, Knowledge systems laboratory.

[7] McGuinness, D.L and Harmelen, F. 2004. *OWL Web Ontology Language Overview*. W3C Recommendation, February 2004.

[8] Motik, B., Patel-Schneider, P. F. and Parsia, B. 2009. *OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax.* W3C Recommendation, September 2009

[9] 2014. *Jena Ontology API. (*Reviewed On 31 May 2014).DOI= http://jena.apache.org/documentation/ontology/.

[10] Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B. and Dean, M. 2014. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. W3C Recommendation, May 2004.

[11] Parsia, B. and Sirin,E. *Pellet: An OWL DL Reasoner*. DOI= www.pellet.owldl.com.

[12] 2014. *SPARQL Tutorial.* (Reviewed on 31 May 2014).DOI= http://jena.apache.org/tutorials/sparql.html.

[13] 2014. *H7 Heart Rate Sensor Polar Global*. (May 31 2014).DOI=**http://www.polar.com/en/products/accessories/H7_heart_rate_sensor.**

[14]Buetooth SIG. (2014). DOI=https://developer.bluetooth.org/gatt/profiles/Pages/ProfilesHome.aspx

[15] Moosa, S. 2014.Infographics: Is Mobile Health Care the Future? February 2 2014.DOI=**https://sites.google.com/a/jozifamilymedicine.org.za/woncaafrica1/family-medicine-in-africa/useful-stuff/infographicismobilehealthcarethefuture**