Patient-Buddy-Build: Customized Mobile Monitoring for Patients with Chronic Diseases

Vitor Pinheiro de Almeida, Markus Endler e Edward Hermann Haeusler
Department of Informatics
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)
Rio de Janeiro, Brazil
{valmeida, endler, hermann}@inf.puc-rio.br

***Abstract*—We present a means of generating a mobile client-server application that enables customized remote monitoring of patients with chronic diseases. The customization is based on parameters and formal descriptions of: type of chronic disease, patient preferences, the monitoring procedure required by the doctor, prescribed medication and information about the current context (i.e. environment) of the patient, obtained from sensors. Based on this, the system determines which pieces of information should be regularly obtained from the patient through questionnaires and/or sensors of the smart phone and sensors connected to it. Relevant information are the ones that help to detect possible changes in the patient's health condition and the monitoring process of a patient by the doctor. The medical treatment and the kind of chronic disease will define the set of data to be collected. It should be stressed that the goal is not to support automatic medical diagnosis, but only to provide means for physicians to obtain updated information about their patients, so as to allow remote monitoring of patients.**

Keywords—Mobile Health; Patient Monitoring; Ontologies; Context-aware application; Knowledge Representation;

I. INTRODUCTION

In the past 10 years the worldwide popularization of smart phones has enabled access to the Internet anytime and anywhere, opening a wide range of new applications and services. For example, in 2012 the number of smart phone users in Brazil reached 27 millions (14% of its population) [1]. In parallel, the coverage of mobile networks also increases continuously. According to Telebrasil, in 2013, the coverage of 3G networks in Brazil already reached 3,315 municipalities (59.5%), and grew by 20% since 2012. It is expected that in the near future even citizens with low income will own smart phones and have access to 3G or 4G networks.

In this scenario there is increasing demand for mobile application development in almost every field, including health care. Mobile Health characterizes the practice of medicine and healthcare through mobile devices [2]. It is an area witnessing rapid growth, where more and more hospitals and healthcare professionals are using mobile technology. Moreover, it is expanding and changing the way medical care is done. In particular, engaging patients and health professionals in using mobile devices with, or connected to, sensors, can be a valuable tool for monitoring, preventing and treating diseases. Furthermore, patients that have difficulties to go to a hospital or to their doctors, either because they live far away, cannot afford the transportation costs, or have movement disabilities, may benefit from a more convenient, cheaper and yet effective medical care through their smart phones.

This work aims at supporting remote monitoring of patients with chronic diseases, mainly because [3]: (a) it is characteristic of the growing ageing population; (b) it typically involves high cost of the treatment; and (c) the affected people may include high risk patients. Thus, remote monitoring of chronic disease patients has both economic and medical advantages. The main problem, however, is that even for a specific chronic disease, every patient develops it in a particular way, and hence requires a customized monitoring process, that includes the collection of specific data and information from the patient and his/her environment. In this paper, we describe Patient-Buddy-Build (PBB)[5][8], an ontology-based framework for generating mobile applications that allow customized remote monitoring of patients with chronic diseases using wireless Internet.

In order to make possible the generation and customization of a patient mobile monitoring application, we needed the representation of medical knowledge and of context information using ontologies [4][6], and the automatic selection of the set of information to be monitored using: (a) this ontology and (b) information about the patient preferences, the sensor data available on his/her mobile device, chronic disease and the required monitoring process. By automatic selection, we mean to create an automatic procedure (a simulation of a diagnosis) that helps the system to choose which set of information is more relevant to obtain from the patient depending on his/her current situation (i.e.: doing sports, resting, or having a rapid heart rate).

The diagnosis procedure for each patient is based on the monitoring procedure defined by the corresponding responsible doctor. Since the monitoring procedure (represented in PBB as an ontology) contains detailed information about how to diagnosis each relevant situation of the patient, PBB is able to select the set of information to be monitored. This is done by identifying which are the next possible health situations that the patient can be in. The set of information that is relevant at each phase of the remote monitoring is the minimum set of information that can determine if the patient's situation has changed or not.

978-1-4799-6644-8/14/$31.00 ©2014 IEEE

The purpose of our work is not to develop a complete and readily usable mobile application, but to use it as a proof of concept of the purposed model for describing monitoring process from formal descriptions (including patient and doctor preferences, symptoms of disease, treatment method, etc) and to identify benefits and limitations of the approach. As a prove of concept of our purposed representation for context information, we develop an example application using the PBB framework to remote monitor patients with Atrial Fibrillation[1].

The remainder of the paper is organized as follows: In next section we discuss the representation in PBB of medical and context information through ontologies; in Section III we then explain the organization of the knowledge base and the possible customizations of our framework. Section IV describes the implementation of a concrete application, and in Section V we evaluate the application described in Section IV. Finally, in Section VI we close the paper with concluding remarks and lines of future work.

II. The PBB Framework Representation Model

A. Forms to acquire context information

PBB framework assumes that disease-relevant context information from a patient can be obtained in three different forms: (a) using sensors embedded in the smart phone, attached to the patient, or carried by him; (b) from web services, e.g.: a weather condition/forecast service; (c) from the input of automatically generated questionnaires about the patient's health condition, answered by the patient. Questionnaires are necessary because information about several health conditions or symptoms (e.g. headache) cannot be obtained in other way. All these forms of obtaining disease-relevant context information are described by an ontology.

B. Representation of medical knowledge

Medical knowledge bases have specific characteristics that must be taken into account when designing them. Clancey & Shortliffe [4] defined a list of important aspects for designing medical knowledge bases: (a) they are inherently incomplete, so modularization and indexing is important; (b) they must support incremental development and easy maintenance because improvements to the knowledge base must be allowed as human experts learn and social judgments change; (c) knowledge sharing and reuse is needed [9]; (d) they must support the ability for a program to explain its reasoning and deductions, since this is important for the acceptability of a system; (e) they must contain well-structured, explicit statements of disease relations and diagnostic procedures;.

We choose the use of ontologies because they facilitate knowledge reuse, knowledge sharing and because ontologies have high degree of formality. Our proposed model for representation of context/medical information is divided into a set of loosely-coupled ontologies, which enhances modularity. It separates the monitoring process from the disease knowledge, the patient specific data and from how each of the context information will be acquired. The representation of information by using ontologies provides explicit statements (formal methods) that gives the system the capability of explaining its decisions and it is also an important point when representing medical knowledge.

C. Representation of context information

Our proposed representation of context information relies on the concept of contextualized ontologies [7], which can be use to represent any context-aware system. Contextualized ontologies are algebraic structures that establish a link between two ontologies. The source of the link is the entity and the target is the context. For our case of remote monitoring, the entity would be the patient, while the context would be the monitoring process to which the entity (i.e., patient) is related. The representation of the *patient ontology* (i.e., entity patient described through an ontology) and the *monitoring process ontology* (context monitoring process described through an ontology) are self contained. This self containment of representations gives a high degree of flexibility. However the links between the entity and the context must be made explicitly defined; and specify how the entity is to "be viewed" from within the corresponding specific context.

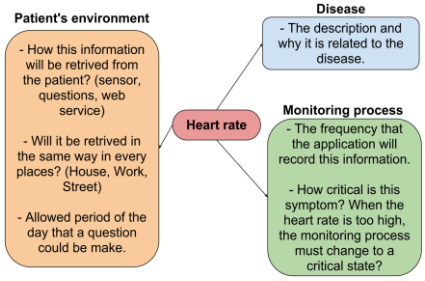

Fig. 1. Context data representaion.

To give a more practical example, Figure 1 shows how heart rate information can be regarded under the point of view of different contexts. When the heart rate is interpreted under the point of view of the disease, it acquires properties like: medical description and how it relates to the disease. On the other hand, when heart rate is seen under the point of view of the monitoring process, it acquire properties like the sampling frequency and how the monitoring process (the sampling) must change according to the situation, e.g., critical state vs. normal state.

In order to give a system this ability - to view a concept (i.e. the heart rate) under the point of view of different contexts (i.e., the disease and the monitoring) - the Algebra of Contextualized Ontologies[7] is very suitable: the heart rate is seen as an entity and represented through an ontology, and each of the contexts where this entity can be contextualized is also represented by an ontology and linked to the entity. These links determine how the ontologies will be aligned by the system. Alignment between ontologies is the operation that produces another ontology that will represent an entity in some context (i.e.: the heart rate in the context of the monitoring process).

III. PBB Ontologies and Customizations

To explain each ontology and how they can be customized, we must define the term "*conditions*", which are regarded as

[1]Atrial Fibrillation description: http://www.webmd.com/heart-disease/atrial-fibrillation/heart-disease-atrial-fibrillation-basics.

any context information that is relevant to the remote monitoring of the patient. Thus, a condition can be, for example, a health symptom of the patient (e.g.: high fever, headache), a data from the patient such as his/her heart rate or blood pressure or even the altitude of the place where the patient is located, if this affects his/her health condition.

A. The organization of the knowledge base

As mentioned, the PBB framework is composed of several self-contained and modular context ontologies, which are listed below:

- ***Disease**:* Contains information about one or more diseases, and what conditions are related to each of the diseases.
- ***Environment**:* Each of the monitored condition has a specific form to be obtained from the patient's environment. This ontology thus describes the means by which data of a monitored condition will be acquired. E.g.: the heart rate may be measured from a sensor attached to the patient, or else, from the patient's answer to a.
- ***Questionnaire**:* Describes how each question of a questionnaire is presented to the user, and how it should be answered, (i.e.; in terms of GUI elements).
- ***Person**:* Personal information about the patient that is relevant for the monitoring process, e.g. his/her age, gender, weight, height, emergency contact information etc.
- ***Patient**:* Contains information about his/her chronic disease(s), prescribed medications, specific characteristics of the patient, allergies, food restrictions, name of the responsible doctor, etc.
- ***Monitoring**:* This ontology specifies how ths system will monitor the patient and how to identify the relevant situations of patient demanding a remote monitoring (e.g.: patient should be monitored only if he/she is doing a physical activity, only when away from his/her home town, etc.).

B. Customizations and some examples

There are two ways to customize an application instantiated using the PBB framework: through the customization of the ontologies or by extension at two hot spots, the classes *ContextProvider* and *PBB_ContextConsumer*. The extension of these two classes gives the possibility to incorporate new sources of information, i.e. sensors and/or web services. All other code of the PBB framework stays the same for any PBB-instantiated application. All other required customizations are done by editing the ontologies. The following are possible customizations of the second sort.

- ***Disease and the set of monitored conditions**:* Define each condition that the doctor wants to be monitored and how each of them will be acquired. The monitored conditions can be any condition or situation (e.g.: heart rate or feeling headache or even a specific interval of time just after a person stopped doing any physical activity) that is possible to be acquired or derived using information from questions, sensors or web services.
- ***The form that each information will be acquired**:* Which sensor will be used for the monitoring process. Maybe it is not necessary to use all the available sensors with a specific patient. Represent each available sensor in the ontology to associate each of them to a condition or a set of conditions to be monitored (one sensor can monitor acquire more than one condition from the environment). It is also possible to add new questions and relate them to conditions that you want to monitor.
- ***Monitoring process**:* Which situation is considered critical and the doctor must be notified. Which patient situation the application should record more intensively the electrocardiogram signal. Which symptoms are interested to be monitored in each of the patient's context. Which information must be persisted by the system and in which case? E.g.: During normal monitoring states the doctor can only want to record the electrocardiogram signal in 6/6 hours by 10 seconds.
- ***Questionnaire**:* The doctor could want to ask questions to the patient and choose the answering form of the question. For example, a doctor could prefer using multiply answers to a headache question that must have only one answer. Other doctor could give the patient the possibility of choosing more then one answer.

All the ontologies have assertions in Description Logic with the objective to help the ontology engineer to avoid mistakes (the insertion of inconsistences) while customizing the ontologies for a new remote monitoring of a patient. The ontology engineer can only import the ontologies in the PBB framework if it passes a consistency check. We use Pellet[1] tool for the consistency check.

IV. USING PBB FOR A CONCRETE HEALTH MONITORING APPLICATION

To demonstrate the feasibility of the proposed context modeling approach and for testing the PBB framework, we developed a context-aware application for remote monitoring patients with chronic atrial fibrillation (AF) disease. In order to monitor patients with AF, we used a sensor[1] that is attached to the patient's chest. This sensor has the capability of monitoring electrocardiogram data (ECG), body temperature, posture, heart rate, breath rate and user activity (using an accelerometer). The data of this sensor is sent to the patient's smart phone (running Android) using a Bluetooth connection.

A. Monitoring process for Atrial Fibrillation

To define how a patient with AF should be monitored, we had several discussions with a cardiologist. The monitoring process is a model, represented by an ontology, that can be viewed as a state machine (an automata). In this paper, the term *state* will be used to denote a health/activity situation that a patient can be in, i.e. a situation that the doctor needs to worry about, and hence, that must be considered in the monitoring process. Each state has three types of variables and any number of actions and behaviors linked to it. Figure 2 shows an example of a state related to AF, the Critical State 2.

[1]Pellet: http://clarkparsia.com/pellet

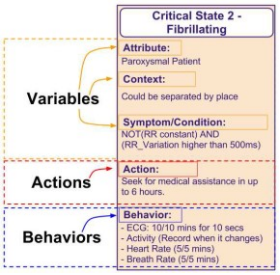

Fig. 2. Critical State 2 for AF, with its variables, actions and behaviors.

The variables, represented in orange color in Figure 2, define if the patient is to be considered or not in *Critical State 2,* or not. If all of the variables are evaluated true, the patient's current situation or activity is the one represented by the variable's state (i.e.: Patient is fibrillating). When any variable is evaluated as true, it asserts a minor situation about the patient (a minor situation is not the situation represented by the state, it's the situation represented by the variable). For example, if his difference between R-waves in an electrocardiogram is varying more than 500 milliseconds or if he is doing any physical activity.

B. *Patient mobile and web server implementation*

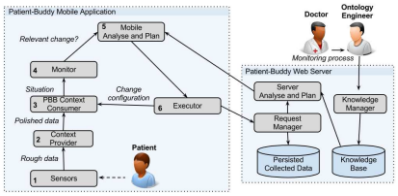

Fig. 3. Patient-Buddy Mobile Application.

Figure 3 represents how the Android-based Patient-Buddy Mobile Application (PBMA) is structured. Block *number 1* (**Sensors**) is the element that retrieves context data from the patient and from the environment. Block *number 2* is the **Context Provider**, that will process the data received by the **Sensors** and send to the **PBB Context Consumer**. The **PBB Context Consumer** (*number 3*) will relate the retrieved context information to a condition defined in the disease ontology and send it to the **Monitor**. Both the **PBB Context Consumer** and the **Context Provider** are implemented using the Context Management Service[2] (CMS). The **Monitor** (*number 4*) considers a new context information to be relevant whenever this new information causes a state variable to become true. After the relevant context information are computed by the **Monitor**, it is sent to the **Mobile Analyze and Plan** (*number 5*), which records all the past relevant changes of each condition and is also responsible for detecting a change in the monitoring state of the patient. When this happens, the **Mobile Analyze and Plan** will send a request to the **Executor** (*number 6*) module to change the monitoring configuration to the new state. The **Mobile Analyze and Plan** is also responsible to send relevant changes of context data to be persisted in the Patient-Buddy Web Server (PBWS). The **Executor** is responsible for executing the orders sent by the **Mobile Analyze and Plan**, which can be: (a) Change the monitoring configuration of the mobile device; (b) Send a context data to be persisted - in the server or the mobile device; or, (c) Send an alert message to the patient's mobile device.

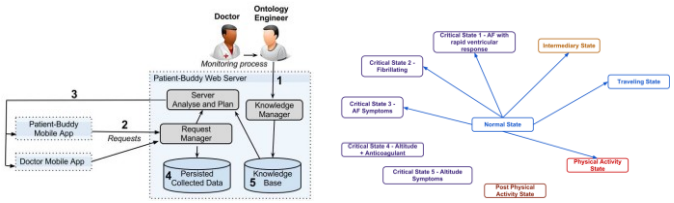

Fig. 4. Patient-Buddy Web Server (left side) and the states reachable by the Normal State(right side).

The PBWS was implemented using JSP[3], JAVA and Tomcat[4]. The database for the **Persisted Context Data** was implemented using the Virtuoso Triple Store[5] and the **Knowledge Base** was implemented using the OWL-Api[6]. The Protégé[1] was the tool used to customize the ontologies. First, the **Knowledge Manager** will receive the ontologies imported by the ontology engineer (*number 1*) and the smartphone number to identify the patient and persist them into the **Knowledge Base**. Interaction *number 2* is activated whenever the PBMA consider that the patient changed his/her state in the monitoring process. After receiving the new state that the patient is inserted, the server (PBWS) sends to the PBMA the appropriate partition of the knowledge base related to current state of the monitoring process. To illustrate how the server selects a part of the monitoring process, the right side of Figure 4 shows a conceptual representation of the possible transitions from the *Normal State*. Let's assume that the PBMA tells to the PBWS that the new state for the patient is the *Normal State*. Therefore, the partition of the monitoring process sent to the mobile device will be only the information related to the states reachable by the *Normal State*. To send the set of states to the PBMA, first the PBWS reads this information in OWL and converts it to JAVA classes. These classes are created representing each of the OWL classes of the ontologies inserted in the knowledge base (containing only the essential data to be sent to the mobile device). Finally, this set of information is converted from JAVA classes to the JSON format, which is then sent to the mobile device. Partitioning the knowledge base is important because ontologies like SNOMED-CT[1] can be used to represent disease concepts, and is not feasible to persist and handle all this ontologies in the mobile device.

One may wonder if the processing of the states on the server is feasible in terms of time to process and space in memory. In order to answer this question we developed some tests that we will show and describe in the next section.

V. Evaluation

As we explained in the last section, every time the PBMA identified that the patient changed his/her state it sends a message to the PBWS requesting for the new configuration, and the PBWS process this request by partitioning the knowledge base (written in OWL) in order to generate a JSON representation of this partition and then send it to the PBMA. In this section, we will focus exclusively on performance tests which access the elapsed time of this communication between

[1]Zephyr sensor: http://www.zephyranywhere.com/products/bioharness-3/
[2]CMS – Context Management Service: http://lac-rio.com/cms/
[3]JSP: http://www.oracle.com/technetwork/java/javaee/jsp/index.html
[4]Tom Cat: http://tomcat.apache.org/
[5]Virtuoso Triple Store: http://virtuoso.openlinksw.com/
[6]OWL-Api: http://owlapi.sourceforge.net/

the PBWS and the PBMA. This process is illustrated by the left side of Figure 4 (with arrows *number 2* and *3*).

A. Tests configuration and simulation parameters

All the tests were executed with the following system configurations and simulation parameters: (a) All the ontologies were the same and the monitoring process ontology contains 10 states; (b) We variate the outdegree number of the current state with the values 4, 6, 8 and 10 (outdegree number is the number of states that the current state can reach in the graph of the monitoring process); (c) The arithmecic mean of context information monitored by each state is 11,1; (d) Total number of monitored context information by the system is 21. Furthermore, we evaluated if there was a difference in the processing time for generating the partition of the knowledge base (KBase) in the PBWS and in the communication between the PBWS and the PBMA if the number of the states were higher.

B. Setup of the Experiment

To test all the communication performance and the processing time we used two laptops interacting through a 2G wireless network. The PBWS was implemented in JAVA, using OWL-API 3.4.2 for data storage and deployed in Apache Tomcat 6.0.37 web server. It was running on Windows 8 at one laptop with a Core I7 2.5GHz processor and 8GB RAM. To simulate the Android application we ran the Android emulator in Eclipse with the ADT Plugin 22.3.0 on the other laptop with the same hardware characteristics as the one used for the PBWS.

We created five (one for each test and hypothetic patient) sets of ontologies in a KBase to simulate five different patients. The ontologies of patient, person, disease, environment and monitoring were almost the same, the only difference is the patient's names, smart phone numbers and theirs monitoring ontology. For the *tests 1* to *4* we set the current state of the patient in the graph (represented in the monitoring ontology) with different out-degrees (4, 6, 8 and 10 states) and in *test 5* we created a total of 30 states and we set the current state to have an out-degree equal to 6.

C. Resutls

The results are presented in Figure 5 and 6. All times are in milliseconds, and all results are an arithmetic mean value of 10 measurements.

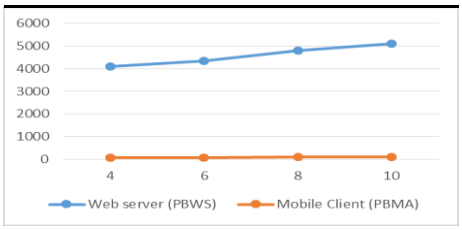

Fig. 5. Processing time in the web server (PBWS) and in the mobile client (PBMA).

The blue line in Figure 5 represents the time that PBWS takes from when it receives the message from the PBMA requesting the new configuration of states until it replies with the new configuration (this process is triggered by the request *number 2 in* Figure 4-left side). As we increase the out-degree number of the current state, the time needed for generating the JSON file that contains the partitioning of the KBase also grows. It can be explained because the partition of the KBase that will be send to PBMA will contain the current state plus all the states that are reachable from this current state. Figure 5 also reveals that almost all the processing is done by PBWS (server side). The PBWS works with the KBase written in OWL-DL, generates a set of JAVA objects that represent a partition of the KBase, convert this set of objects to JSON format and send this string to PBMA. The PBMA receives the string in the JSON format and generate the JAVA classes (process triggered by the transaction represented by *number 3* in the left side of Figure 4). The conversion between the JAVA classes and the JSON format is done using the GSON[2] API 2.2.3[1] (both in PBWS and in PBMA). The significant difference in the processing time between PBWS and PBMA is explained by the manipulation of the OWL-DL ontologies that PBWS must do in order to partition the KBase and generate the next configuration of states for the monitoring process of the patient.

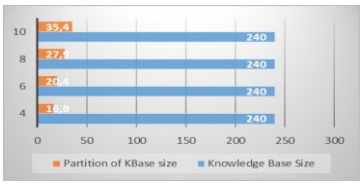

Fig. 6. Sizes of the Knowledge Base and generated partition (in KB).

The blue bar of Figure 6 represents the size of the KBase of each test (~240 KB), and the orange bar represents the size of the partition generated by the PBWS that is sent to the mobile client. As we increase the out-degree of the current state, the size of the partition also becomes larger. This is because the string in JSON format that will be send to the PBMA will contain the current state plus all the states that the current state can reach. As we can see, the partitioning of the KBase significantly reduces the size of the information that would be persisted in the mobile device by PBMA. Hence, we use less resources of the mobile device as only the partition of the KBase related to the current state of the patient has to be stored. Figure 6 also shows that the maximum size of a partition with this configurations and parameters (defined in subsection VI.A) is 35,4 KB, which is a reasonable amount of data to be transferred to a mobile device. Since the maximum out-degree of a state is the total number of states in the monitoring process (in this case, is 10), the first orange bar represents the maximum size that the KBase partition can have.

In test 5, we only evaluated the processing time and partition size when the total number of states of the monitoring ontology is equal to 30. For this test we used a current state with the out-degree equal to 6, and we asserted that the processing time and the partition size was the same as if the monitoring process had 10 states. It can be explained because the number of states does not affect the algorithm to partition the KBase. The total number of states only affects the size of the KBase, which for 30 states reached 451 KB.

[1]SNOMED: http://bioportal.bioontology.org/ontologies/SNOMEDCT
[2]GSON: https://code.google.com/p/google-gson/

TABLE I. TIME ELAPSED UNTIL PBMA RECEIVES THE NEW MONITORING CONFIGURATION

State Out degree	Time (ms)
4	**4263**
6	**4565**
8	**5053**
10	**5362**

Finally, we evaluated the time elapsed during the communication cycle (represented by actions transactions 2 and 3 of Figure 4-left side) from the request for a new monitoring configuration sent by PBMA until it receives the new configuration and retrieves the partition from JSON format to JAVA classes. As we can see in Table I, when the state's out-degree grows the elapsed time also gets higher. It is mainly because the processing time elapsed in the server side by PBWS also grows with the current state's out-degree number.

Although the tests are only preliminary ones, they show that PBB produces acceptable message sizes (35 KB) and round trip delays (less than 6 seconds) for a representative KBase.

VI. CONCLUSION

In this paper, we described a model for representing context information for remote monitoring of patients with chronic diseases. To illustrate the functionalities of the proposed model, we developed the PBB framework that can be used to instantiate applications for monitoring a variety of chronic diseases. We present a concrete application that can monitor patients with AF and evaluate it by executing time performance tests and measuring the size of the result partition of the KBase for each test.

Other contribution of this work is the set of ontologies written in OWL-DL that provides a vocabulary to describe a variety of details of a remote monitoring of patients with chronic diseases. All the ontologies provides logical restrictions (written in OWL-DL) that aids the user who wants to write new monitoring procedures to avoid customizing the ontologies in an inconsistent way.

The monitoring ontology, compared to the other ontologies, brings one different contribution for this work that is it's form of representing context information. The set of states model of this ontology was created with the main purpose of representing the monitoring process of a patient with atrial fibrillation. However, it was generalized to represent a monitoring process in general, because the domain of the monitored information is not represented in the monitoring ontology. Thus, we can see this model as a form of representing context information, that focus on the monitoring of some entity, that can be a person, an object or even group of them. When we say that the model of states focus on the monitoring, we say that because during a monitoring is necessary that the application knows how to behave towards the acquired context information and how to take different actions according to the context that the application is inserted, and both are characteristics that can be represented in the model.

Another contribution of this work is that it gives a practical implementation and a possible application to the theoretical framework purposed by [6]. This practical implementation proves that with only this set of logical operators and level of formality it is possible to implement an application that can remote monitor patient with chronic diseases that have the set of functionalities that we offer in PBB. The practical implementation of PBB does not contain the *existential* and the *for all* operators of the first order logic available for the user to customize and express his/her form of remote monitoring of patients.

Medical guidelines [9] may be used in future work for enhancing our medical knowledge base. Medical guidelines, (a.k.a clinical guideline, clinical protocol or clinical practice guideline) are documents that contain guiding procedures and criteria regarding diagnosis management and treatment in specific areas of healthcare. They do not contain protocols to be strictly followed, but their objective is just to guide the doctor on how to manage and treat each disease. Providing a set of ontologies that implement a specific guideline can be a solution for minimizing the effort of the doctor to insert the remote monitoring knowledge.

REFERENCES

[1] Google, "Nosso Planeta Mobile: Brasil", May 2012, Available in http://services.google.com/fh/files/blogs/our_mobile_planet_brazil_pt_BR.pdf [Accessed in March 2014]

[2] R. S. H. Istepanian, S. Laxminarayan, C. S. Pattichis, "M-Health: Emerging Mobile Health Systems". New York, NY, USA: Springer, 2006.

[3] A. Asmi, L. Ragavan, "Pervasive asthma monitoring system. pams. a health systems approach to remote monitoring of asthma." Georgia Institute of Technology. Gatech.edu, n.d. 2007, Available in www.hsi.gatech.edu/healthcaredesign/files/asthma_report.pdf [Accessed in April 2014]

[4] W. J. Clancey, E. H. Shortliffe, "Readings in medical artificial intelligence." Boston, MA, USA: Addison-Wesley Longman Publishing Company, 1984, pp. 4-14.

[5] V. Pinheiro, E. Haeusler and M. Endler, "Patient-buddy-build: Customized mobile monitoring for patients with chronic diseases." In: The 5th International Conference on eHealth, telemedicine, and social medicine, ISBN: 978-1-61208-252-3, PP. 121-124, Nice, France, February 2013.

[6] T. R. Gruber, Int. J. Hum.-Comput. Stud. "Toward principles for the design of ontologies used for knowledge sharing", journal, v.43, n.5-6, pp. 907–928, Dec. 1995.

[7] I. Cafezeiro, J. Viterbo, A. Rademaker, E. Haeusler, M. Endler, "Specifying ubiquitous systems through the algebra of contextualized ontologies", Knowledge Eng. Review 29(2): 2014, pp. 171-185.

[8] V. Pinheiro, M. Endler and E. Haeusler, "Patient-buddy-build: Customized mobile monitoring for patients with chronic diseases." MSc. dissertation, Pontifícia Universidade Católica do Rio de Janeiro, Brazil, 2013.

[9] R. A. Greenes, S. Tu, A. A. Boxwala, M. Peleg, E. H. Shortliffe, "Toward a Shared Representation of Clinical Trial Protocols: Application of the GLIF Guideline Modeling Framework" in *Cancer Informatics*, Health Informatics series, J. S. Silva, M. J. Ball, C. G. Chute, J. V. Douglas, C. P. Langlotz, J. C. Niland, W. L. Scherlis, Ed. New York: Springer, 2002, pp. 212-228.