Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Technology 5 (2012) 820 – 829

Procedia Technology

CENTERIS 2012 - Conference on ENTERprise Information Systems / HCIST 2012 - International Conference on Health and Social Care Information Systems and Technologies

A Methodology for an Architecture of Pervasive Systems to Homecare Environments

Leandro O. Freitas[a], Rafael T. Pereira[b], Henrique G.G. Pereira[a], Ricardo G. Martini[a], Bruno Mozzaquatro[a], Jeferson Kasper[a], Giovani Rubert Librelotto[a]*

[a]*Univeridade Federal de Santa Maria, Santa Maria, RS, 97105-900, Brazil*
[b]*Universidade do Minho, Campus de Gualtar, Braga, Portugal*

Abstract

These days, queues in hospitals grows due to, among others, the increasing of world population and delay in patient's attendance. One way of solving this problem is through the development of systems that ensure treatment directly in the house of patients. These systems help to decrease queues and improve attendance of those who goes to hospitals looking for assistance. In this paper we present the modeling of architecture for pervasive systems to homecare environments. Systems with this modeling aim to improve services provided by professionals during treatment of patients located in their houses. We used concepts of pervasive computing to provide access to information anytime and wherever the user is, once a homecare environment has a high level of dynamicity. The knowledge representation of the environment is done through ontologies due to the possibility of reuse of information stored, as well as the interoperability of information among different computational devices.

Keywords: pervasive computing; ontologies; homecare treatment; cloud computing.

* Corresponding author. Tel.: +55 55 9137 0380; fax: +55 55 3220 8523 (r 28).
E-mail address: librelotto@inf.ufsm.br.

doi:10.1016/j.protcy.2012.09.091

1. Introduction

One of the main problems faced nowadays by administrators of hospitals is to offer a quality service to their clients. Every year, the queues in hospitals grow due to a set of factors like, for example, high demand of patients, problems of infrastructure, poor internal communication and lack of professional commitment, which compromises the clinical treatment of patients [4].

One way to minimize the delay on the costumer service is to invert the way that the patient is treated today. Instead of the patient goes to the hospital to receive the treatment, the clinical services go to the patient's house through the implantation of medical applications. A homecare environment is characterized by the possibility of the patient to receive medical care directly in his house, whether by necessity like in cases where the patient is unable to move, or even by his own will, once he will stay more time with the family and friends [7].

A homecare environment may become very dynamic, once the patient can live and enter in it wherever he wants, walk through the rooms or receive visit of relatives and friends. Besides being at home, the patient should be monitored constantly, and any sudden worsening on his health state should be treated immediately. This way, a system with medical applications to homecare should support these frequent changes in the context of the environment [1].

In this scenario, pervasive computing appears as a solution to solve problems like mobility and dynamicity that characterize a homecare environment. Pervasive computing allows access to information anytime and anywhere and, thus, can provide quick and secure access to the medical record of patients to clinicians through applications even if they are not in their workplaces [2]. They can take faster decisions about which treatment apply to a specific patient whose health is worse.

To develop these kinds of applications we need a system that could use information from the context of the homecare environment. For this, a detailed representation of the knowledge existing in the domain is needed. One of the more appropriate ways to make this representation is with ontologies. They can be defined as a formal representation of a contextualization. Ontology is composed by entities and relations defined formally and with a detailed semantics. If these relations have names that identify their meanings, a human will be able to understand it as well as software could assume a semantic of a relation and act through it [9]. Ontologies ensure interoperability of environmental information that travels through the system so that they are correctly interpreted without ambiguity, independently of the platform of processing of the devices.

Using concepts of pervasive computing and ontologies, this paper has three main purposes connected each other. First, we intend to find a way to assist the health professionals in their daily tasks. With this, the workflow of the clinical treatment of the patients will be improved. With these two goals achieved, the last, which refers to the improvement of the attendance of patients in their houses, also will be achieved.

To do that, we present a methodology for architecture for pervasive systems with medical applications in homecare environments. From this architecture it will be possible to develop pervasive systems to assist professionals during the treatment of patients. These systems ensure a constant communication between the patient's house and the clinic that is offering the service. Thus, if a patient has sudden worsens; the system communicates immediately the professional on duty in the house of the patient, and also warns the hospital (or clinic) if a specialized backup is needed. This is possible because the homecare environment will have sensors monitoring the patient and sending information about his health state to the system.

Another problem solved with applications developed to homecare environments is the overcrowding of hospitals. Once exists the possibility of the patient to receive the treatment at home, with the same quality of the hospital, the queues tend to decrease considerably, and consequently, the patients that still decide to receive treatment in hospitals will receive a better attendance. Finally, with this methodology we intend to

improve the workflow of treatment of patients in their houses through the suggestion of possible next steps that professional might want to take during their routines.

This present is organized as follows: in section 2 we present the architecture for pervasive systems design to homecare environments, describing each of the modules that composed it (2.1) EHR, (2.2) Sensors, (2.3) Monitoring and distribution of information, (2.4) OntoHC, (2.5) Cloud processing and (2.6) Computational devices. Then, in section 3 we present the conclusions and future steps of the project.

2. Modeling of an architecture for pervasive homecare systems

The main goal of pervasive homecare systems is to make the job of healthcare professionals faster while treating their patients. The system, based on information captured from the context of the environment, is able to suggest possible actions to be taken by the professionals during their workflows. Thus, this section aims to describe architecture for homecare environments. We developed ontology with a formal representation of the existing knowledge of the domain to be used with the system. Figure 1 presents the architecture of a pervasive healthcare system focused on this kind of environment.

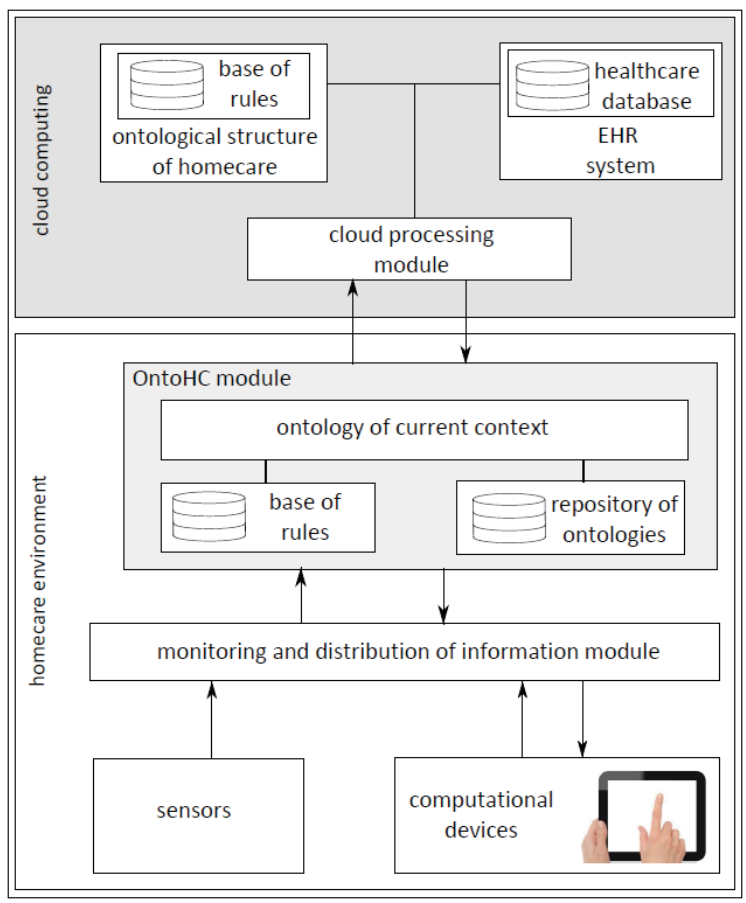

Fig. 1. Architecture of homecare system

There are two domains that must exchange information about the entities involved. The first domain is a computational cloud that stores information related to the pervasive hospital. In this cloud there is the ontology that represents the homecare environment, which was developed with Web Ontology Language - OWL [8], a recommendation from W3C Consortium for this kind of representation. The cloud has a database that stores inference rules written with Semantic Web Rule Language - SWRL [3] and queries written with Semantic Query-Enhanced Web Rule Language - SQWRL [6] to be performed on the ontology. We chose these languages because they can use all the expressivity of OWL. The Electronic Health Record (EHR) and its database are also located on the cloud. The ontology does not have any instances, i.e., it is only an ontological structure composed by entities and relations that may exist in a homecare environment. From this ontology, the system will create a new one with information only about the entities of a specific current context. Then, the system will be able to execute applications to help the users in their tasks.

The computational cloud a module to process information. In this module the system will create a new ontology that contains only information about a specific context. The sending and receiving of XML documents make the communication between the layers of the system. These documents contain relevant information about a specific situation. The *cloud-processing module* manipulates the XML documents that come to it with the help of a parser. This parser receives a XML document containing tags related to a specific situation (e.g. identifier codes of entities detected by the sensors) and processes this information in an appropriate manner (e.g. uses the identifiers to invoke a inference rule related to an entity). Besides that, this module has a system based on *Web*, which is used by the professionals that work directly at the patient's house.

The system has the following workflow: the homecare environment has sensors which perform frequent searches aiming to find changes in the context that, somehow, could affect the medical treatment of the patient. Among these changes, we could cite: the entering of a physician in the room for his daily round; changes in the vital signs of the patient; changing shift of the professionals in service; schedule of a specific medicine to be administrated to the patient; and the schedule of tasks that the patient may have to accomplish. When this kind of information is detected, they are inserted in a XML document and sent to the *monitoring and distribution of information module*.

The *monitoring and distribution of information module* is responsible for treating accordingly the information received from the sensors and send them to the *OntoHC module* or to the computational devices. First, the information is sent to the *OntoHC module* to verify if the current ontology stored there has the entities detected by the sensors. In affirmative case, this module searches on its database rules that could be executed to start applications for the users.

A specific context refers to a current situation of what is happening in the homecare environment, e.g., which entities are present and which relations exists between them. This way, the ontology of the current context refers to all classes and properties that represent the entities present in the homecare environment in the current moment.

Thus, if the entities detected by the sensors are not represented in the ontology, it will be sent to the repository of ontologies and a new one need to be created with the classes and relations referring to the detected entities. For this, the *OntoHC module* sends a XML document with information about the instances to the *cloud-processing module*. This module creates a new ontological structure formed by classes and relations referring to the entities detected and fill this structure with updated information from the EHR system.

At this point, the *cloud-processing module* sends to the *OntoHC module* an ontology that represents the current context of the environment. This module uses the inference rules and queries database to verify if exist applications that can be performed in order to be used by the environment people (physician, nurses, patient, cares). In affirmative case, the *monitoring and distribution of information module* is notified and sends an

appropriate application to one of the computational devices to interact with the user through interfaces. An example of this process happens when the *OntoHC module* verifies, searching the rules database, that exists a relationship between an individual of the class *Physician* and an individual of the class *Exam* through the property *visualized_by*, *which* is checked as *false*, i.e., exists an exam of the patient that the physician has not visualized yet. The *OntoHC module* informs this to the *monitoring and distribution of information module*, which sends a notification to the closest computational device from the user, asking him if he wants to analyze the exam in that moment.

The layers that compose the architecture of the homecare pervasive system are described separately as follows.

2.1. Electronic Health Record System of Patients - EHR

The cloud computing has a module with a Electronic Health Record System (EHR) with a database with general information about the health condition of the patients like, for example, his medical history and the medicines that are current being administrated to him, as well as the applications to manipulate these data. The information from the database is used to fill the structure of the ontology of current context, which are called individuals or instances. After use information of the EHR database to fill the structure, the *cloud processing module* sends the new ontology to be manipulated by the *OntoHC module*.

In this architecture, we propose the use of the EHR system in a cloud computing [5] due to the fact that hosting it in a cloud, not only one homecare environment will be able to use it, but any other that decides to do it. In other words, with the EHR system hosted in a cloud, any patient that has medical record stored in that database could receive clinical treatment in his house. Besides that, if the environment of treatment changes, e.g., patient leaves his house to be treated in a house of a relative; the EHR system does not suffer any changes.

More specifically about the proposed architecture, a meaningful gain of the use of the system hosted in a cloud computing is the possibility of reach the abstraction level proposed by the pervasive computing, where the users (physicians, nurses or cares) access their applications and the willing information anywhere and at anytime, without concerns related the computational infrastructure that maintain the system working, which stays "invisible" to the users. This way any device with Web access will be able to use the applications of the EHR through a browser.

The system does not need to be executed in a cloud computing to this centralization of information. It could be in a Web Server of the hospital. The majority of the applications of these systems do not require a large processing; they basically address input and output of data. This way, a normal Web server could be enough for the system to work. However, cloud computing also supports more robust applications like tools to assistance of diagnosis, prediction of diseases, among others. These kind of applications indeed require a larger processing and can use the information stored in the EHR system as source for its reasoning.

Besides that, adopting this approach we could use the patterns of cloud computing, like the maintenance of systems or processing elasticity according demand. In this case only information about patients are stored in cloud domain. The operational information of hospitals, like financial, for example, are stored in each hospital.

2.2. Sensors

In general, a pervasive environment should have sensors that monitor it searching for relevant changes that could interpose in the correct functioning of the system. The workflow in a homecare environment is the

same. The disposed sensors should be able to detect changes in the context and use them in prol of the users (patients, physicians, nurses or cares).

These sensors should be able of, e.g., detect movements of people, changing the position of objects or the time that the user performs a task, and transform these data in relevant information to be used appropriately during the workflow of the system. When a relevant change is detected, the sensors capture information about the entities involved and store them in a XML document to later send to the *monitoring and distribution of information module*.

This XML document if composed by three tags that store information related to the event detected. The first tag is called *id* and refers to the identification of the sensor. This tag is important because the environment in question can have more than one type of sensor and this should be considered for the correct functioning of the system. For example, an event detect by a sensor that monitors the vital signs of a patients should be treated in a specific way, since that a sudden change on it could represent a crises in the health state of the patient and, consequently, should be solved with urgency. A sensor that detects a entry and leaving of people in the environment not always represent a case of urgency.

The tag *info* refers to information collected like, e.g., the temperature of a patient. This tag stores information of an instance of the ontology. The last tag stores the type of information collected, e.g., the identifier of a person. It stores information referring to the attribute of a class of the ontology. This is related to the second tag. For example, if the system detects 37.5 degrees of temperature of a patient, the content of the XML document will have the value 37.5 degrees in the tag *info* and in the tag *type* the value *Temperature*.

We should clarify that to each entity detected in a context, a XML document correspondent will be generated. This means that if, for example, a nurse in a room is cared a patient, and in that moment a physician enters in the environment, the sensors will detect the change in the context and a XML document will be created only with information related to the physician. This happens because the other detected entities (instance of *Patient and Nurse*) are already mapped in the current ontology stored in the *OntoHC module*. This XML document is sent to the *monitoring and distribution of information module*.

2.3. Monitoring and distribution of information module

This has two main purposes: (a) check with the *OntoHC module* if the entities detected by the sensors are already mapped in the ontology; and (b) execute applications that interact with users through the computational devices assisting them in their tasks. It is responsible for controlling the information of the environment. Through it the information can travel between the different layers of the architecture.

The *monitoring and distribution of information module* has a parser that allows it to manipulate separately each piece of information of the XML documents received from the sensors. It uses information from the tags to verify with the *OntoHC module* if the structure of the ontology of current context stored there contains the entities detected by the sensors. If yes, the *OntoHC module* searches on its database rules that could be executed involving one or more entities and notifies the *monitoring and distribution of information module*. This module starts communication with the computational devices that will execute applications to assist users in their tasks. Sending a XML document containing information from the *OntoHC module* does this communication.

The document is composed by five elements. The first of them refers to an identifier code of the device. This information is needed for the system to know for which device it should send a specific application. The second element refers to the user that will interact with the system. In this tag is stored a identifier code of the user. The third element has the class of the ontology from which the instance stored in the tag *user* belongs to. The fourth element refers to a task that the user will perform using the computational device. The content of this tag represents an object property in the ontology, which connects two entities. The element *parameter*

will have other two tags, which can store one or more information needed to execute a task. The tag *type* represents the kind of information that is being passed as a parameter, which in the ontology is a datatype property (attribute). This tag can contain, for example, name of exam, symptom, or name of medicine. By last, the tag *info* represents an instance stored in the property of the tag *type*.

A scenario that describes the use of this XML document happens when the *OntoHC module* verifies that the temperature of the patient is 38.9 C, i.e., above a maximum secure limit pre-established. The *monitoring and distribution of information module* sends this XML document to the computational device closest to the physician. The tags *type* and *info* of the element *parameter* will have the information *temperature* and *38.9* respectively. Based on this information the physician can take the best decision for the patient's problem to be solved.

2.4. OntoHC module

This is responsible for manipulating the ontology of current context. It has a database with a set of inference rules and queries and a repository of old ontologies that were used in previous contexts. The inference rules can be executed when user is performing a task. For example, when a physician prescribes a new medicine, a new instance of the class *Medicine* is created and must be described in the ontology through the running of a rule. This process is managed automatically by the system. This database will also be used to search information in the ontology of current context and send them to the cloud computing to update the EHR database.

The *monitoring and distribution of information module* sends information about entities detected by the sensors to the *OntoHC module*. This module uses the SQWRL queries to check if such entities are already mapped in the ontology. If they are, it verifies which actions could be taken and communicates the *monitoring and distribution of information module*. However, if the classes are not described yet in the ontology of current context, a new ontology should be created. To do that, the *OntoHC module* should establish a communication with the cloud computing informing the *cloud-processing module* about which entities should compose the new ontology. The system sends a XML document created by the *OntoHC module* with information from the *monitoring and distribution of information module*.

This XML document is composed by a name of the class and the identifier of the individual (instance) detected by a sensor. The *cloud-processing module* receives only one XML document for each modified context. In the scenario where the physician enters in the room where the patient and nurse are, the document would have the classes *Physician, Patient and Nurse* and the identifier of each one. With this, the *cloud-processing module* is capable of creating an ontology containing only information related to these individuals (attributes, relations, restrictions, inference rules, and queries).

When we perform inference rules on the ontology of current context, perhaps some new information about the entities are generated. An example of that can be described when a physician prescribes a new medicine for the patient “John”. When this happens, a new instance of the class *Medicine* is generated, which will have relations with the individual “John” through the properties *precribed_for* and *takes* (*Medicine prescribed_for Patient and Patient takes Medicine*). This information is related to the treatment of the patient and must be inserted in his medical records. This way, before the *cloud processing module* creates a new ontology, the information that were represented in the ontology of the previous context should be sent to the EHR system. For this, along with the *cloud.xml* the system also sends another XML document containing information from each entity represented in the current ontology. This information is inserted in the EHR system. The detailed description of this document is in section 2.5.

At the same time that the *OntoHC module* sends the XML documents to the cloud, it also sends the ontology that was stored there to the repository of ontologies. Thus, when the *OntoHC module* receives the

ontology of current context already instantiated from the cloud, the pervasive system will be able to manipulate it properly, e.g., through the performing of inference rules, queries, or even wait for a new relevant event to happen for the treatment of the patient.

2.5. Cloud processing module

This module is responsible for all the processing in the cloud computing. It is able to update the EHR system and search for updated information on it and responsible for accessing the ontological structure stored there to create a new ontology only with classes referring to a specific context.

To perform this processing, the module receives the document *ehr.xml* sent by the *OntoHC module*. Then, a parser manipulates the information of the ontology separately and sent them to the EHR system. This ensures that each time that a new ontology is created and instantiated with information from the EHR system, it will have updated information about the entities involved.

Four tags with information of each represented entity compose this XML document. This way, the *OntoHC module* should send to the *cloud-processing module* a document referring to each entity detected by the sensors. The first tag refers to the class from which the information belongs to. It will identify the class in the ontology. The second tag represents a datatype property of the information that is being sent. The third tag stores the date of the change of context related to this entity. The *cloud-processing module* will compare its content with what is already stored in the EHR system and will update it only if the date is different. The last tag stores the instance of the entity, i.e., to which individual belongs the information of XML document in the ontology.

After update the patient's clinical record in the EHR system, the *cloud-processing module* creates a new ontological structure only with information of the current context through XSL processing. With XSL is possible to access the OWL document and select determined elements, creating a new ontology. It searches for classes and properties of the entities detected by the sensors, which characterizes a *current context*. The module performs queries in the database of the EHR system looking for information related to those entities to instantiate this new ontology. Then, the system sends the new ontology to the *OntoHC module* to be manipulated properly. At this point, the system has an ontology with information of the entities present in the current context stored in a module located in the homecare domain. Thus, to execute an application, the system does not need a communication with the computational cloud unless a new entity is detected in the environment.

2.6. Computational devices

This module represents computational devices that, somehow, ensure the interaction between the system and users. Among these devices we can cite touch screens, tablets, equipment to monitor vital signs, or any computational device with processing capability and memory to support applications available in the system.

The workflow of the computational devices is described through the example that is being used throughout this paper. When physician enters in the room where the patient and nurse are, the *OntoHC module* verifies that exists in the ontology a relation between instance of the classes *Exam* and *Physician* through the property *visualized_by* which is checked as *false*, i.e., the physician did not analyzed yet an exam that he had asked for. *OntoHC module* sends a notification about this to the *monitoring and distribution of information module*, which sends the document *compDev.xml* to the closest computational device from the physician. The device shows a suggestion asking if he wants to analyze the exam. He can accept or decline the suggestion. If he accepts, the computational device sends a XML document to the *monitoring and distribution of information*

module, which searches in the *OntoHC module* information about this specific exam and then shows them to the physician through the closest computational device.

This document has the same structure of *compDev.xml*, but in this case it has information needed to perform the task. Among the document's parameters, it will have the information about the decision of the physician in visualizing the exam. The first element of the document refers to an identification code of the device that is being used by the user. This code allows the system to verify if that kind of device supports the requests of the user. For example, if the physician is using his smartphone to access the system, he will not be able to analyze a X-Ray exam, since this image is not supported by this kind of device. The second element refers to the user. This tag stores a code that identifies which instance of the class *Person* is using the system. The third element refers to the class from which the instance belongs to. These two tags allow the system to check what kind of applications could be executed to each user. For example, if an individual of the class *TechnicalNurse* is using the system, he could not prescribe a medicine to the patient. The fourth element stores a string that identifies which task the user wants to request. If the physician wants to visualize an exam, this tag will have a string like *visualize_exam*. This string is compared to the element *RDFS:Label* in the ontology. The ontology has one element of this type associated to each task described on it.

The last element corresponds to a list of one or more parameters needed to perform a specific task. Two tags compose this parameter: *type* and *info*. The tag *type* refers to the name of the datatype property that is being passed as parameter. It can contain names like *name of medicine, posology, name of exam or type of treatment*. The content of this element can range according the task that the physician wants to perform. The second element corresponds to the information itself, i.e., the instance of the datatype property that was defined in the other tag. The content of it could be, e.g., *paracetamol* for *name_of_medicine*; *8h to 8h hours* for *posology*; *endoscopy* for *name_of_exam* or *bedrest* for *type_of_treatment*.

In some cases to perform a task it might be needed more than one set of this information. This means that the system will need more than one set of parameters. Cases like these are predicted and can be manipulated in the cardinality of the element parameter.

3. Conclusion

The development of pervasive applications for homecare environments appears as an incentive to more people receives medical treatment in their houses. This kind of service ensures a higher level of comfort for the patient, once they will stay at home, allowing them to spend more time with relatives. With this kind of treatment, the queues in hospitals tend to decrease and the service there also will be improved.

We presented a methodology for an architecture to be used as bases to develop systems for homecare environments through the concepts of pervasive computing applied to health care. We used ontology to represent the existing knowledge in the environment. The system can help users in different situations ranging from simple applications like reminders of daily activities of patients, to more complex scenarios, like when the system offer assistance to professionals during their workflow, suggesting possible actions to be taken.

We intend to continue this project focusing on issues not covered so far, like safety and privacy of information. At this point we assume that the homecare environment has a computational infrastructure that ensures the development of pervasive applications and also that it treat issues of safety and privacy properly. Finally, we will perform simulate tasks through the construction of a real homecare environment. With this we will be able to analyze better the functioning of the system, and then make it available to real users.

References

[1] Librelotto G, Augustin I, Gassen J, Kurtz G, Freitas LO, Martini R, Azevedo RP. OntoHealth: An Ontology Applied To Pervasive Hospital Environments. In: Handbook of Research on Mobility and Computing: Evolving Technologies and Ubiquitous Impacts. Ed. Maria Cruz-Cunha and Fernando Moreira. IGI Global p. 1077-1090, 2011.
[2] Chen H. An Intelligent Broker Architecture for Pervasive Context-Aware Systems. University of Maryland, Baltimore Count, 2004.
[3] Horrocks I, Patel-Schneider PF, Boley H, Tabet S, Grosof B, Dean M. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. DARPA DAML Program, 2004.
[4] Andrade LM, Martins EC, Caetano JA, Soares E, Beserra EP. Atendimento humanizado nos serviços de emergência hospitalar na percepção do acompanhante. Revista Eletrônica de Enfermagem. V. 1 p.151-157, 2009.
[5] Vaquero LM, Rodero-Merino L, Caceres J, Lindner M. A break in the clouds: towards a cloud definition. SIGCOMM Comput. Commun. Rev. V. 39 p. 50-55, 2008.
[6] O'Connor MJ, Das AK. SQWRL: A Query Language for OWL. In. OWLED. Ed. Rinke Hoekstra and Peter F. Patel-Schneider. V. 529, 2008.
[7] McGee-Lennon MR. Requirements engineering for home care technology. Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems. p. 1439-1442. CHI'08, Florence, Italy, 2008
[8] Bechhofer S, Harmelen FV, Hendler J, Horrocks I, McGuinness DL, Patel-Schneider PF, Stein LA. OWL Web Ontology Language Reference. W3C, 2004.
[9] Gruber TR. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. In Formal Ontology in Conceptual Analysis and Knowledge Representation, Kluwer Academic publishers, in press. Substantial Revision of Paper Presented at the International Workshop on Formal Ontology, 1993.