Are The Integrations Between Ontologies and Databases Really Opening the Closed World in Ubiquitous Computing?

Vinicius Maran, José Palazzo M. de Oliveira  
Institute of Informatics - Federal University of Rio Grande do Sul (UFRGS)  
Porto Alegre – RS – Brazil  
{vmaran, palazzo}@inf.ufrgs.br

Iara Augustin  
Technology Center - Federal University of Santa Maria (UFSM)  
Santa Maria – RS – Brazil  
august@inf.ufsm.br

Abstract: Currently, ontologies are widely used in ubiquitous systems for the representation of context and situation information. In addition, task ontologies are used to describe processes and daily situations that may occur in certain domains. In this way, it is necessary that there are strategies of integration between ontologies and databases, because ubiquitous systems generate large amounts of context data. In addition, it is necessary that exist ways to query and infer information on these persisted ontologies. Through the description of a motivating scenario, based on ClinicSpace ubiquitous architecture, and the study of requirements identified on recently proposed architectures, it was possible to list a set of important features to use persistence tools in ubiquitous systems. This paper presents an analysis of recently proposed solutions for the use of ontologies as data model for context representation and situation inference in ubiquitous systems. Through this analysis it was possible to list a set of requirements that the actual state of the art does not attend, such as the possibility of distributed inference about information in ontologies.

Keywords — Ontologies; Ubicomp; Databases; Context-Awareness; Pervasive Computing.

I. INTRODUCTION

The idea proposed by Mark Weiser [31], known as Ubiquitous Computing, defines a world composed by intelligent environments where users and devices of all kinds are fully integrated in a way that provides the invisibility of computing. In these spaces, computing should be transparent to the user, i.e., devices and systems should assist users in performing their daily tasks and it is desirable that users can not perceive the aid of computation involved in these processes. Thus, the main feature of ubiquitous systems is that they perform tasks centered on the end user and their daily activities according to the needs of these users and the context where they are inserted [14].

Context can be represented computationally in various ways. One of the most used ways is through ontologies, which allow the formal representation of a set of concepts and terms that represent domain knowledge [27]. Moreover, ontologies can be used to represent situations, and can be used for inference of new information based on the definitions of previously modeled contexts and situations. For the ubiquitous systems can complete the adaptation to the context as a result of situations that occur during their execution (context-awareness adaptation) is necessary that exists strategies to interconnect context information with data persistence tools. Recent researches propose the definition of methodologies for this interconnection, but there is still no consensus on the use of a particular data model.

In this paper we present a motivation scenario based on ClinicSpace ubiquitous healthcare system [17][11], which defined a ubiquitous system for customization of tasks and clinical workflow inside hospitals. From this scenario and studies carried out on ubiquitous architectures from different domains we defined a set of features for the integration of context modeling, persistence, recovery and context inference tools. Furthermore, we found that many of these requirements are common to other ubiquitous architectures, regardless of the field in which they are used.

From the definition of these requirements, we conducted a review in relation to state-of-the-art works that propose the integration between ontologies and databases in order to verify whether these solutions meet the requirements for their use in ubiquitous systems, and if these solutions are really integrating the concept of Open World (OW – present in ontologies) with databases – which uses the concept of Closed World (CW).

OW defines that information or structures that are not modeled in an ontology exist in the world, but are not modeled. Thus, a query about a not represented knowledge in the ontology may not result in a false statement; it must return a response stating that the knowledge is unknown [21][16]. CW in turn defines that everything, which is not modeled in the conceptual model of the database, is false. Thus, if a query involves data that do not exist in the database, the response will be false, if a similar query was made in an ontology, the answer would be unknown [21][16]. Additionally, databases use UNA (Unique Name Assumption), which states that every individual has a single and unique name. In ontologies, individuals may have more than one name.

Next, Section 2 records the concepts used for modeling context and situation information in ubiquitous systems. Section 3 describes a motivation scenario based on ClinicSpace architecture. Section 4 describes the state of the art of integration between ontologies and databases. Section 5 illustrates a comparative study of methods of integration between ontologies and databases and presents a discussion of results, which
indicate a range of implementation difficulties and research possibilities in the area. Section 6 presents the conclusions of this work.

II. CONTEXT AND SITUATION INFORMATION IN UBIQUITOUS SYSTEMS

Ubiquitous systems base their operation on information collected from the environment and on user interactions. The inputs data are collected by sensors abstracted, forming context data. The term context can have several definitions, because it is a broad term, which encompasses several areas.

According to Dey & Abowd [10], context can be defined as "any information that might be used to characterize the situation of entities (person, place or object) that are considered relevant to the interaction between a user and an application including the user and application". Bazire & Brézillon [5] define context based on two main settings: (i) the context acts as a set of constraints that influence the behavior of a system, embedded in a given task, and (ii) definition of context depends on the area of knowledge to which it belongs. [10] and [5] are the most widespread and used context definitions in recent studies of context awareness.

Context data can be represented in various ways. Recent studies [26][23][6] made comparisons between the forms of representation of context and the requirements that they meet. These comparisons were based on six key factors, named in the comparative table as: distributed composition (dc), partial validation (pv), information quality (qua), incomplete and ambiguous (inc), level of formality (for) applicability in existing environments (app) and interoperability between systems without the necessity of adaptation (int) [26]. Table 1 shows the compiled result of these comparisons.

TABLE I. COMPARISON BETWEEN FORMS OF CONTEXT REPRESENTATION. ADAPTATION OF [26][23]

<table>
<thead>
<tr>
<th>Ordered Approaches</th>
<th>dc</th>
<th>pv</th>
<th>qua</th>
<th>inc</th>
<th>for</th>
<th>app</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key-Value</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Markup Scheme</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Graphic</td>
<td>-</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Logic Based</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Object-Oriented</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ontology Based</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Thus, it was found in the comparisons that context modeling based on ontologies meets key requirements for the complete representation of contexts. Moreover, this form of representation allows interoperability between systems in different languages, a difficult task to be accomplished if we use other ways, for example, based on object oriented approach which depends on characteristics present in each programming language.

In addition to defining context, the definition and inference of situations in ubiquitous systems are necessary because they allow ubiquitous systems and services to determine what actions should be executed before a combination of situations and contexts [32]. Situation is an abstract concept that can be defined in various ways. Some authors [19][32] define it as an external abstraction (from the point of view of applications) for a set of relevant contexts on environmental entities and their semantic relationships. There are also proposals to add time variant concept in situations [7][3]. These proposals include the concepts of situations that must have a certain time of existence, and the lifetime of the situation must be situated in global time. For example, agitated_patient situation can happen during certain periods of time during a day in the field of clinicians care to elderly patients.

In literature, there are proposals to use methodologies for the representation and detection of situations. Ontologies can be used to specify situations. Moreover, logical rules can be created to perform the inference of situations and determine which operations must be performed based on the state of the environment [26]. From the modeling of context and situations based on ontologies, it is necessary to use language standards for queries and inferences in these definitions.

For queries and inferences to be made in information represented in OWL-DL ontologies, W3C [29] recommends the use of three languages: SWRL (Semantic Web Rule Language), used for the development of logical rules and consequently the inferences, SQWRL (Semantic Query-enhanced Web Rule Language), which adds functions to SWRL for queries in OWL-DL ontologies. SWRL is a language used to define rules in first order logic and is widely used in ubiquitous architectures. The code snippet shown below is an example of SWRL rule applied the definition of Lying action, “if an user has a tag, the tag is at 10 cm high from the floor, and it is staying for 60 seconds” [15].

\[\text{owns(?user, ?tag)} \land \text{hasHeight(?tag, ?height)} \land \text{swrlb:lessThan(?height, 10)} \land \text{stayFor(?tag, ?duration)} \land \text{swrlb:greaterThan(?duration, 60)} \rightarrow \text{hasMovement(?user, Lying)}\]

Fig. 1. SWRL definition of Lying situation [15]

III. MOTIVATION SCENARIO

To identify important requirements for persistence and use of context information in ubiquitous systems, we describe a usage scenario of ClinicSpace architecture. This scenario was created based on the descriptions provided by other studies [17][18]. Besides defining the scenario and rank a set of important requirements for persistence and use of context information, we relate these requirements with other recent works of ubiquitous architectures, to determine whether these architectures meet these requirements.

ClinicSpace ubiquitous architecture was designed to be used in hospitals, allowing doctors and clinicians to model their everyday activities in computer interfaces. This modeling of activities should be aware of the context, i.e., each one of the activities defined in the workflow of the user is performed according to the contexts of environments where the activity will be held. For this to occur, the ClinicSpace architecture uses the definition of contexts based on OWL-DL language, and stores these settings using the SemanticCouch architecture [18]. To illustrate some important requirements on the use of context, we describe a scenario where the ClinicSpace ubiquitous architecture is embedded.
"A neurologist wants to consult the history of his patient exams on his mobile device. ClinicSpace obtains the current context, identification of the patient, location of the physician, and patient, devices and sensors. All information is modeled in OWL-DL [35]. The physician receives the results of tests made recently by the patient who is being treated, concerning their clinical specialty. Moreover, the tests are presented to the physician in a manner adapted to his device - for the viewing capability of the display, the available bandwidth of the network, location where the doctor is and the people around him (if this information requires privacy or not).

From the definition of this usage scenario, we identify a set of requirements for persistence and retrieval of information used by the architecture. Moreover, some of these requirements have been also identified in other recent studies of ubiquitous architectures [28][13][12]. The following are the main requirements for the persistence of context and situation information in ubiquitous architectures:

- **Vertical and Horizontal Scalability.** The persistence tools that support semantic data should provide (i) vertical scalability - possibility of increasing the capacity of the database when it is used locally; and (ii) horizontal scalability – data replication in databases on multiple nodes of a network, allowing these nodes to access data in the same way as a local database. For that it can use a distributed infrastructure to store and query data, allowing the data persisted in this architecture to be found in several places by devices of all kinds. This feature is required by the use of large amounts of information sensed from the environment in ClinicSpace [18] (generated from sensors and inferences about the environment), also, the amount of data is a problem in other ubiquitous architectures [28][13][12], and the use of heterogeneous devices is required - two key characteristics of ubiquitous systems [33][18];

- **Standardized query and inference languages for OWL.** Preferentially, the methods of integration and persistence of ontologies tools should use query and inference standardized languages for OWL. Among these languages include the SWRL and SQWRL languages, since they are subject to standards W3C [29] and permit the inference and query data in the OWL language [32][28]. Inferences and queries can also be made using SPARQL [29], but this language was defined for use in conjunction with the RDF standard. All recent architectures that uses OWL-DL as context representation (including ClinicSpace) uses SWRL to infer about context [28][13][12];

- **Persistence only of the relevant data.** The constant conversion of formats and query languages makes that occurs a steady expansion (overhead) on the overall size of the data, resulting in considerable loss of performance depending on the size of the database and the number of queries. Furthermore, the space occupied by these data greatly increases with the use of these databases [4]. [18] and [4] presents recent studies of solutions to minimize the overhead of data, but these solutions do not implement solutions for reasoning over OWL-DL;

- **Using part of the ontology in memory.** The processing of inferences and queries should be done only with relevant data, mainly because the performance and memory consumption increases considerably with the increase of individuals in the ontology. Therefore, it is important that changes be processed on a limited data part - through a filter to use only part of the data in memory [4]. Also, the actual widely used inference engines over OWL-DL have high computational complexity, through the use of Tableau or Hypertableau algorithms. This level of complexity decreases the performance of data management, as shown in [35];

- **Support to queries, inference and modeling of temporal situations.** Situations should be modeled and should be stored according to their occurrence. The storage of situations allows ubiquitous architectures to perform processes proactively. For this to happen, it is necessary that the integration methodologies enable the modeling, query and inference of situations based on the time variation;

- **Integration with mobile platforms.** Ubiquitous systems must support heterogeneous devices in a distributed manner; this implies the use of mobile platforms in the integration of the system with the environment. In our vision, it is necessary that the strategies of integration between ontologies and databases allow architectures to use the partial replication of information and reasoning over data on mobile devices;

- **Support for OWL-DL standard.** The OWL-DL standard is commonly used to represent context information in ubiquitous systems. According to W3C, OWL-DL offers maximum expressiveness without losing computational completeness and decidability of inference engines. OWL 2 [29] offers a new set of profiles and tools, but OWL-DL still widely used to represent contexts and situations [34][22].

IV. INTEGRATION TOOLS BETWEEN ONTOLOGIES AND DATABASES

Strategies for integrating ontologies and models of databases have been proposed in two distinct lines: (i) integration of ontologies in relational databases [33], and (ii) integration between ontologies and NoSQL databases or distributed file systems. Semantic data are used in many areas of computing. Therefore, some solutions are defined for specific purposes, with representation languages and specific queries to the field where they will be applied. The information of techniques surveyed in this study are presented in Table 2.

A. Relational Databases

Many solutions related to the integration between databases and ontologies use relational model as basis for integration.

SciSPARQL [2] is an architecture that expands SPARQL language to address relevant issues to the scientific field - specifically in research groups that use information modeled in RDF format and have large amounts of generated information. This architecture allows scientific data to be stored in RDF triple format using the relational data model. To carry out the queries, the architecture allows the use of SciSPARQL, a query language that extends the functions of SPARQL query language [14] with external functions defined in Python language. The SciSPARQL does not support inferences in persisted ontologies. This feature is referred to as a topic of future work by the authors.
Shan [24] presents a methodology for integrating ontologies defined in RDF format and relational database MySQL or PostgreSQL. Through a mapping table created specifically for this architecture, the algorithm C-Store [24] defines the database based on the inserted ontology model. SPARQL language is used to query information persisted in the database.

Nyaya [8] is a persistence architecture of semantic data that allows large amounts of information to be stored. To do this, each imported RDF in architecture is converted into a structure called Semantic Data Kiosks with structures and modeled data according to the imported RDF. These structures also generate some extra data that allow queries to be made using a Datalog based language. When a query is performed in the language, the architecture converts the query into subqueries in SQL, and then accesses the data persisted in relational database.

ERMOS [25] is a project that implements the integration of OWL-DL and MySQL database. It implements the Rule Based Ontology Query Language (ROQL) language. Through the use of this language, developers can query (but can not do inferences), which are converted into SQL statements.

Based on the possibility of insertion of XML files without the necessity of conversion to other data model, OXDBS [20] proposed the extension of the functionalities of eXist database to the possibility of syntax checking of RDF and OWL-DL files, and allows the use of RacerPro inference engine to query and inference ontologies.

Db4OWL architecture [4] proposes the integration of OWL-DL ontologies and DB4O database, which is object-oriented. To accomplish this integration, the authors created a model in Java language that represents the structure found in OWL. From this structure, the version and the information contained in it were stored in the database. Queries and inferences are written in RuleML format [4], and subsequently converted to QBE (Query by Example) format, used by DB4O database.

SemantiCouch [18] proposes an architecture of integration between OWL-DL ontologies and CouchDB document-oriented database. Thus, the information could be replicated easily if compared to other approaches. To perform the queries, the architecture provides an API based on QBE.

Other approaches propose the integration of ontologies and distributed file systems, allowing the same ontological definition to be replicated in many places. JenaPro [30] presents an architecture of integration between Jena framework - for the use of ontologies in the Java language, and Hadoop architecture - used for the distribution of information.

Swarms [9] presents an architecture for the persistence of semantic data represented in RDF format. The architecture was modeled and implemented in a way that allows the distribution and replication of information across nodes in a network. This information distribution is done intelligently, through the analysis of paths between nodes, to choose the best route information distribution. To carry out the queries, the authors use ALC language [9].

OCCSS (Ontology Cloud Storage System) [1] presents an architecture and a set of algorithms of reading and writing ontologies modeled in RDF and OWL languages persisted in GFS (Google File System). This architecture allows a Cloud Computing infrastructure to be created for the storage and querying of ontologies.

### TABLE II. INFORMATION ABOUT INTEGRATION TOOLS BETWEEN ONTOLOGIES AND DATABASES USED IN THE ANALYSIS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Model</td>
<td>RDBMS</td>
<td>RDBMS</td>
<td>RDBMS</td>
<td>RDBMS</td>
<td>XML</td>
<td>Object Oriented</td>
<td>JSON docs</td>
<td>Hadoop files</td>
<td>Files</td>
<td>Files</td>
</tr>
<tr>
<td>Query Language</td>
<td>Sci</td>
<td>SPARQL</td>
<td>Sparql</td>
<td>Based on Datalog</td>
<td>RDF</td>
<td>nRDF</td>
<td>RuleML</td>
<td>QBE</td>
<td>Not mentioned</td>
<td>Alc</td>
</tr>
<tr>
<td>Inference</td>
<td>Not mentioned</td>
<td>Not mentioned</td>
<td>MySQL</td>
<td>eXist XML</td>
<td>Db4Owl</td>
<td>Not mentioned</td>
<td>Couch</td>
<td>Hadoop</td>
<td>Not mentioned</td>
<td>GFX</td>
</tr>
<tr>
<td>System</td>
<td>Postgre / MySQL</td>
<td>Not mentioned</td>
<td>MySQL</td>
<td>eXist XML</td>
<td>Database</td>
<td>Not mentioned</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

V. ANALYSIS AND DISCUSSION

Ontologies have been used as a form of contexts and situations representation in recent ubiquitous architectures, regardless of the areas where these architectures are inserted. Thus, it is necessary that exist forms to integrate ontologies and databases, to allow control of the information captured by the environment and allow this information to be consulted and new information be inferred. Methodologies for integrating ontologies and databases are being proposed, but there is no consensus among studies regarding in which model database is best to perform this integration.

The more used model is the relational, mainly by extensive use by its formal conceptual model and the number of tools associated with this model. However, more recent approaches have been proposed with the use of other models in order to overcome some shortcomings of this model.

From the analysis of the characteristics of the recently proposed methods, we performed a second analysis. This analysis aims to determine whether the methods proposed thus far meet the requirements listed in Section 3, based in the scenario of ClinicSpace use, for the use of integrated databases in ubiquitous systems ontologies.

The low overhead requirement in data conversion is contemplated in more recent studies. In the works [30][9][1] there is no file conversion, because the works proposed persistence architectures using distributed file systems. The work [20] also shows no file conversion, it uses XML as a data storage model. Other recent works [4][18] proposed the use of different storage strategies to decrease the creation of new data on the persistence of ontologies process. Specifically on these two works, object-oriented and JSON language were used. As we can see, the data overhead is well worked by recent researches.
About horizontal scalability, only 4 studies in the comparative offered any solution. The works [30][9][1] provide horizontal scalability through the use of distributed file systems. However, these do not use a database management system; thus, do not offer query languages or inference to ontologies.

Regarding query languages, most jobs do not allow the systems to query and inferences using languages based on Database. Defined by the W3C standard for querying RDF document in databases - in most of these works, the SPARQL language is utilized. But the SPARQL language does not provide the means to achieve inferences, supported by languages SWRL and SQWRL, so little support builders and important definitions found in standard OWL-DL.

Still regarding query languages, without the possibility of inferences on the data stored in databases, some jobs do not offer the ability to query and inference of situations based on timing issues. Most recent studies support the persistence of files in OWL-DL standard. This enables contexts to be represented and stored by ubiquitous architectures. However, none of the studies reviewed presented in the form of the integration testing with mobile platforms.

This requirement is important in ubiquitous architectures, since it reduces the necessity of existence of a central server with all context representations. Instead, mobile devices can contain and manipulate context definition according to the device, subsequently; the captured context can be replicated on these devices. Table 3 shows the result of the relationship between the requirements presented in Section 3 and state of the art methods of integration between ontologies and databases.

<table>
<thead>
<tr>
<th>Requirements / Work</th>
<th>[2][24][8][25][20][4][18][30][9][1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low data overhead</td>
<td>- - - - - - - - - - - - - - - - - -</td>
</tr>
<tr>
<td>Easily Horizontal Scalable</td>
<td>- - - - - - - - - - - - - - - - - -</td>
</tr>
<tr>
<td>Support of query and inference languages to OWL (SWRL, SQWRL or based on Datalog)</td>
<td>- - - - - - - - - - - - - - - - - -</td>
</tr>
<tr>
<td>Support inferences and temporal modeling of situations</td>
<td>- - - - - - - - - - - - - - - - - -</td>
</tr>
<tr>
<td>Integration with Mobile Platforms</td>
<td>- - - - - - - - - - - - - - - - - -</td>
</tr>
<tr>
<td>OWL-DL support</td>
<td>- - - - - - - - - - - - - - - - - -</td>
</tr>
</tbody>
</table>

As we can see, there is still a gap between the horizontal scalability of architectures of ontologies persistence and the support of inference and query in standard languages OWL-DL. This is primarily because the majority of inference and supporting the SWRL and SQWRL engines work patterns to form separate DBMSs that store ontologies.

Architectures for context management are directly related to bond with sensors and mobile devices. The processing and storage capacity of mobile devices may limit the action of architecture. Also, perform all operations on the context in one place (client-server) is not recommended because the environment is heterogeneous and distributed [36][37]. Historic analyses are directly related to the ability of ubiquitous systems persistence and recovery of context. If modeling is done using ontologies are generally used the SWRL language (for inferences) and SPARQL (for queries in RDF). However, these languages do not offer mechanisms to monitor the flow of information in real time, and have few resources compared to similar languages based on SQL for queries in relation to time/space [6][26]. Efficient inference and query that integrate ontologies are still a problem for ubiquitous systems [6][26][38]. Furthermore, these approaches do not meet the mobility requirements needed for ubiquitous systems.

Distributed persistence only offers the possibility to distribute and query the information, but the possibility to infer about ontologies in a distributed manner is a high complexity problem. In ubiquitous systems, the realization of these inferences should be made with portions of ontological representations. Figure 2 shows an adaptation of the scheme presented in [6].

![Pyramid of context representation according to level of abstraction and view of the world. Adaptation of [6]](image)

This schema of context management differs from other proposals, because it proposes the use of ontology as the top support to the whole process of representation (from the data representation to the representation of situations). Thus, architecture for context management could manage contexts and situations without constant conversions of formats, and the ability to make inferences at different levels (for there is the use of ontologies with different levels of abstraction and worldview).

For this to occur, it is necessary that exists: (a) A lightweight form of serialization of ontologies based on a W3C standardized language, such as JSON; (b) An inference engine that performs the inference of ontologies in levels, according to the worldwide represented in the ontology; and (c) An architecture management framework that provides those allied to a persistence model that prioritizes efficiency over the generation of extra data features.

VI. CONCLUSIONS

Ontologies have been used as a way of representation of contexts and situations in recent ubiquitous architectures, regardless of the areas where these architectures are inserted. Queries and inference efficiently has still demonstrated a major challenge in ubiquitous systems, this is primarily because the current methodologies for integrating ontologies and databases do not include all the features of the inference engines.

The performed study shows that there are still great opportunities for research in the area, especially in relation to ubiquitous systems. In this field there are still major challenges relat-
ed to integration of inference and managers of databases on heterogeneous devices (fixed and mobile) engines, and in relation to efficiency analysis of information, due to the large amount of generation of information occurs in ubiquitous systems.

**REFERENCES**


