Contemporary Engineering Sciences, Vol. 12, 2019, no. 3, 103 – 116
HIKARI Ltd, www.m-hikari.com
https://doi.org/10.12988/ces.2019.9515

Linked Data and Dialetheic Logic for Localization-Aware Applications

Carlos Jiménez, Marxjhony Jerez, Jose Aguilar

CEMISID, Facultad de Ingeniería
Universidad de Los Andes
Mérida, Venezuela

Rodrigo García

Departamento de Ingeniería de Sistemas
Universidad de Sinú
Montería, Colombia

Abstract

The number of applications for mobile devices that exists and that is created on a daily basis is enormous. These applications, called micro-location applications, are a type of context-aware application that provides services depending on the location. These applications need to model the context, typically using ontologies. This paper extends a contextual ontology, called CAMeOnto, with linked data and dialetheic logic paradigms, to enrich and manage ambiguities in this applications.

Keywords: micro-location applications, linked data, dialetheic logic

1 Introduction

Nowadays, technological advances in computing capacities, and especially in sensors and mobile devices, have allowed the development of applications with behaviors that depend on the context. These Context-Aware Applications (CAAs) constantly monitor and react in different ways, to adapt to the situation in which they are [1, 2]. Particularly, micro-localization applications have appeared as a type of CAA for mobile devices [2, 4]. They provide services depending on the location. In general, CAAs models the context, typically using ontologies [2, 7], to represent the most important characteristics of the environment: users, activities, devices, etc.

Moreover, currently there is a large amount of data on the web; if these data are semantically linked, we can get information and knowledge from them. In recent years, the amount of data linked that have been appearing on the web is huge [12, 13]. The generated ontologies for CAAs can be enriched with this immense amount of data on the Web, which is semantically linked [12]. However, these applications change of context very quick and deal with human beings, which can mean ambiguous situations in their ontologies that lead to inaccurate reasoning. For these reasons, the inferences must be extended with Dialetheic logic [5].

These domains, linked data, Dialetheic logic and micro-localization applications, need to be combined. In specific, the micro-localization applications need to model the context in which they are, using ontologies. If these ontologies are enriched with linked data, and a reasoning mechanism that manages the ambiguities, the micro-localization applications can increase the knowledge that they possess, improve the services they provide, and specifically, personalize their activities for each user.

In a previous work, we have proposed an ontology to model contexts, called CAMeOnto [2]. CAMeOnto is a two-level meta-ontology, the first layer is a general context ontology, and the second layer models the specific domain. The first layer is divided in three components: the *Internal context* about the user, the *External context* about the environment, and the border context with the interactions between these contexts. CAMeOnto is used by CARMiCLOC, a reflective middleware for context-aware applications [1]. This paper proposes to extend CAMeOn, with linked data and Dialetheic logic paradigms, to enrich the contextualization service, which manage the uncertainty, for micro-localization applications.

There are several works in the field of context modeling using ontologies [9, 10, 11]. Normally, the ontologies are described using the Ontology Web Language (OWL). Also, there are some works about context modeling using linked data, which try semantically to enrich the information about the context [12, 13]. On the other hand, there are no works about context modeling using dialetheic logic, or that mix ontologies with linked data and dialetheic logic. The works about micro-location applications aware of the context, normally are based on Beacons and their capabilities to use local information [14, 15]. The utilization of knowledge models (like ontologies) and their reasoning mechanisms are not considered, and less linked data or dialetheic logic. Thus, this paper has two challenges: to propose a context modeling framework that combines ontologies, linked data and dialetheic logic; and to support micro-location applications aware of the context based on this framework.

2 CARMiCLOC and CAMeOnto

CARMiCLOC is a reflective Middleware [1], which incorporates the autonomic computing for the self-adaptation, and the cloud computing paradigm for a SaaS (Software as a service) behavior, to provide services and functionalities that allow managing the context for CAA. This middleware is divided into two levels: the *base level* corresponds to the users and applications that are in the shared context. At this level, the middleware monitors the interactions in the context and their changes.

The *meta level* offers the reflexive capabilities, from the observations of the base level. At this level, the services of CARMiCLOC are customized and provided. The architecture includes an autonomous manager that implements an intelligent context management loop to automate the self-configuration of the applications, based on the current context. CARMiCLOC has seven services to perform the context management, which are: i) *Context acquisition* (Se_1)*:* the context discovery is carried out in this service. It is based on the data extraction from sensors and their pre-processing; ii) *Context Modeling* (Se_2)*:* This service represents the context as knowledge, in order to infer information. CARMiCLOC uses an ontology for the modeling of the knowledge, called CAMeOnto [2]; iii) *Context reasoning* (Se_3)*:* This service applies reasoning techniques to infer information. Also, it analyzes the uncertainty in the context; iv) *Context distribution* (Se_4)*:* it is responsible for delivering the context to the users; v) *Context verification* (Se_5)*:* It is responsible for detecting inconsistencies in the context and resolve them; vi) *Context Security* (Se_6)*:* it is responsible for the security of the information about the context (privacy and data integrity); vii) *Context Service Manager* (Se_7)*:* it is an orchestrator of the services of CARMiCLOC. For more details about CARMiCLOC, see [1, 7].

CAMeOnto is an ontology to model the context of any domain, and captures the concepts of high level of a context. In addition, CAMeOnto provides mechanisms, to be extended with context-specific information [2]. CAMeOnto is used by CARMICLOC in its services of context Modeling, context reasoning, and context distribution. CAMeOnto is characterized by two hierarchical levels; the *first level*, a general ontology independent of the domain; and the *second level* allows incorporating ontologies of specific domains. In addition, the first level categorizes the context into three contextual categories: a) *Internal context:* it describes the user information, such as its profile, preferences and role; b) *External context:* it describes the environment of the users, using the following contextual categories: i) Activity: it describes the different activities that can be developed in the context; ii) Time: it describes the notion of time in the context; iii) Location: It describes the location of the context; iv) Environment: it describes the conditions and the properties of the user's location, c) *Border Context:* it describes the entities that perform the interactions between the users and the environment, and is described by the following contextual categories: i) Device: it describes the software and hardware devices in the context; ii) Services: it describes the existing services in the context. In addition, CAMeOnto incorporates other classes: the role, the service profile, and the activity domain. The service profile describes the services in more detail, and associates a role to a service. The role is also associated with an activity domain. For more details about CAMeOnto, see [2].

3 System Design

3.1 System Architecture

The system must perform the next actions: i) Obtain the context using CAMeOnto; ii) Make inferences with the data that possesses CAMeOnto (First Order Logic);

iii) Enrich semantically CAMeOnto (Linked Data); iv) Manage ambiguities in CAMeOnto (Dialetheic Logic). The system architecture is shown in Figure 1, which has three main components: i) *Semantic Enriching*, it links the CAMeOnto ontology with other data. Particularly, the data sources are the domain ontologies, and the Semantic Web. With all that information, the Semantic Enriching extend CAMeOnto using the linked data paradigm. It is connected to the services that get the context in CARMiCLOC; ii) *Semantic Reasoner*, it receives CAMeOnto enriched by the linked data paradigm, and, using axioms, it makes inferences. It needs two components. The first is a component for the inferences using First-Order Logic. However, as there are ambiguities in the context, it uses a second component based on the dialetheic logic; iii) *Output Generator*, it uses a Query Handler, to provide the information to who invoked the service. It takes the inferred data, and formats them, to deliver them to the applications that have invoked them.

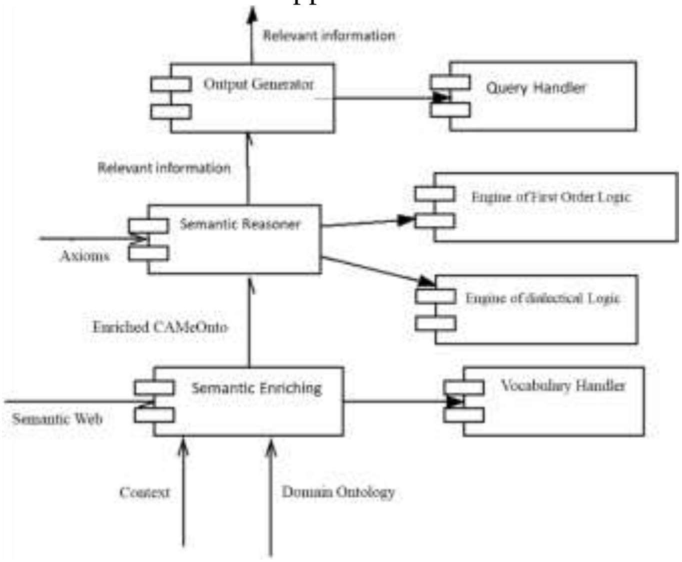


Figure 1: System architecture.

3.2 Axioms of the system

The *Semantic Reasoner* component is composed by a group of axioms. The set of proposed axioms is generic, based on the classes of CAMeOnto. In Table 1 are shown the axioms, with a corresponding example, and the type of logic to which they correspond. In Table 1, there are axioms in both *First Order Logic* and *Dialetheic Logic*. The axiom 4 involves two terms that generate ambiguity. The first one is the *time*. Depending on the user; an hour of waiting can be a long time, but for another user it might appear to be short. It can also depend on the situation in which the user is located, since, in an airport an hour of scale is short, while in a bank queue an hour of waiting is long. The second is the *distance*, in the same way as in the previous case, this term depends on the user and its situation, while for some users to move a certain distant may seem a lot, for others it can seem short. The next axiom that involves Dialetheic logic is the axiom 7, which falls in two cases of Dialetheic logic [5], the first of vagueness of natural language. In the

Example 1 of this axiom, it is exposed that Oscar is in an auditorium, so this is open, however, this is not necessarily the case, and a possible reason would be that there is a special event for a certain group of people. The second case is a contingent statement on the future. Seeing the example 2, a possible case is that just at this time the restaurant is open, however, it is almost the closing time, so in minutes it will be closed. The other axioms involve First-Order Logic, without ambiguity.

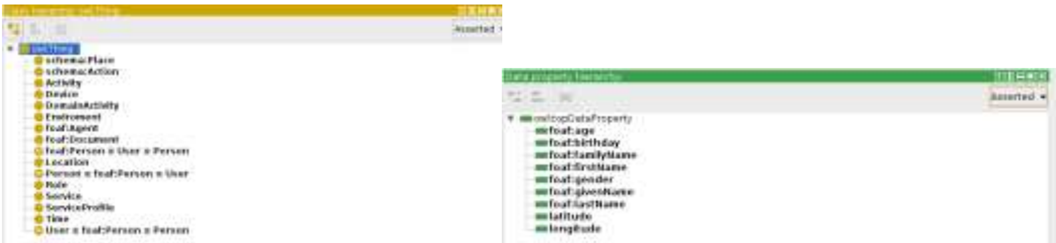
Table Nº1. Axioms of the system

N	Axiom	Example	Logic Type
1	If a role is developed in a location, then the activities that have a role-related activity domain can be done at that location.	If you are a student and you are in the library, then you can read, research, study, in the library.	First-Order Logic.
2	If a user uses a device in a location, and the device provides a service, then the location has the device.	If Carlos researches using a computer in a library, then the library has a computer.	First-Order Logic.
3	If a user is in a location, and uses a device that provides a service, then the service is provided by the location.	If Maria surfs the Internet in a library, then the library provides the Internet.	First-Order Logic.
4	If localization A has an activity in time B and user is located in location C in close time to B, and A is near B, then the user can go to the activity.	**Example:** If there is a concert in an auditorium at 5:00pm and John is near at 4:45pm, then John can go. **Example 2:** If Carlos is at home at 3:00pm and there is a play at 4:15pm in a theater, then Carlos can go.	Dialetheic logic. an ambiguity: vagueness due to natural language in the terms of distance and time.
5	If a user has a role and is in a location, then the role can be developed on the location.	Andrea is a student and she is at the Faculty of Engineering, then Andrea can be a student of the Faculty	First-Order Logic.
6	If a user uses a device that provides a service, then the user can consume the service.	If Peter uses a computer that provides information from books in a library, then Peter can use the information from the library books.	First-Order Logic.
7	If a user performs an activity in a location in time A, then the location is open on time A.	**Example 1:** If Oscar is in a concert in an auditorium, then it is open. **Example 2:** If Gerardo is dining in a restaurant, then it is open.	Dialetheic logic. Two ambiguities: *vagueness* and *Contingent statements* about the future.
8	If a user does an activity in a time A, then the user cannot do more activities in time A.	If Valentina has an exam, then she can't have any more tests at that time.	First-Order Logic

4 System Implementation

4.1 CAMeOnto Implementation

CAMeOnto has been implemented with *Protégé* 5.2.0 [4]. In addition to the CAMeOnto classes, CAMeOnto has been enriched with classes of known vocabularies, such as *Friend of a Friend* (known as FOAF [3]) (see Figure 2.a). The properties of the classes of CAMeOnto are divided into two types: the data properties define the properties of a class (see Figure 2.b); and the object properties define relationships between classes (see Figure 3). These are the basic elements of the *Semantic Enriching* component, which extends CAMeOnto through the use of the semantic Web based on the linked data paradigm.



a) CAMeOnto classes in Protégé b) CAMeOnto data Properties in Protégé

Figure 2: Extensions to CAMeOnto classes

Figure 3: CAMeOn object Properties in Protégé

For data properties, some properties defined by the FOAF ontology were included. On the other hand, the object properties (see Figure 3) were included from the original CAMeOnto properties, and other aggregates to allow the linked data.

Later, using Protégé again, the axioms based on First Order Logic were defined (see Table 1), using the Semantic Web Rule Language (SWRL) tool. The SWRL is a language based on OWL which allows defining rules. The antecedents and consequents are sets of atoms joined together by conjunctions, where atoms can indicate: a variable of a class; a relationship between variables; equivalence of variables; etc. An example of the first axiom defined using SWRL, is:

Role(?r) ^ developsIn(?r, ?l) ^ Activity(?a) ^ hasDomainActivity(?a, ?da) ^relatedToDomainActivity(?r, ?da) => canBeDone(?a, ?l)

4.2 Implementation of the Dialetheic Logic

The *JGXYZ* inference engine was used to implement the Dialetheic logic [6]. The axioms generated that imply dialetheic logic were written in the syntax accepted by the *JGXYZ* system. An example of the input to *JGXYZ*, corresponding to the Axiom 7 (see Table 1), is presented below:

```
fof(locationIsOpen, axiom,(? [User, Activity, Location, Time]: ((user(User) & activity(Activity) & location(Location) &
            time(Time)  & locatedIn(User, Location) & doing(User, Activity) & isTime(User, Time) )
            => open(Location, true) ) )).
```

4.3 System Interface

The interface is based on the Java programming language. A *Prototype* class was created for the interface implementation, which has private methods:

1. readContext(String):void,
2. replacePrefix(String):String,
3. executeQuery(String):Model,
4. getDialecticLogicAnswer(Buffe redReader, BufferedReader): String.

The first method reads the file in the direction indicated by the parameter, and assigns the content of the ontology to the model that is the attribute of the prototype class. The second method facilitates the exit of the models, replacing the spaces of name that exist in the URIs by their prefix. The third method executes a query in the context according to its parameters, and returns the respective response. Finally, getDialecticLogicAnswer invokes the *JGXYZ* system.

In addition, the prototype class has a large number of public methods, which are used to specify the queries behind our axioms: i) queryContext(String):ResultSet: This method allows the predicate to be queried in the context; ii) readInitialContext():void: it allows the context to be read from a default path. It uses the ReadContext(String) method; iii) readInitialContext(String):void: it allows the context to be read from a path, which is indicated by the parameter, iv) activitiesThatCanBeDoneAtLocation():Model: it uses a query in SPARQL to query the context. This query invokes the Axiom 1. It uses the executeQuery(String)method; v) usersWhoCanConsumeServices():Model: it uses a query in SPARQL to query the context. This query invokes the Axiom 6 (see Table 1). It uses the executeQuery(String) method; vi) devicesLocatedInPlaces():Model: it uses a query in SPARQL to query the context. This query invokes the Axiom 2. It uses the executeQuery(String) method; vii) locationProvidesService():Model: it uses a query in SPARQL to query the context. It invokes the Axiom 3 and uses the ExecuteQuery(String) method; viii) roleDevelopsInLocation():Model: it uses a query in SPARQL to query the context. It invokes the Axiom 5 (see Table 1); ix) activitiesThatCantBeConsumedByUsers():Model: it uses a query in SPARQL to query the context. It invokes the Axiom 8. It uses the ExecuteQuery(String) method; x) userCanGoToActivityNearHisLocation(String): String: It uses the JGXYZ system to infer, using the file indicated as a parameter, and it returns the response of the system JGXYZ. It invokes the Axiom 4 (see Table 1); xi) locationIsOpen(String): String: it uses the JGXYZ system to infer, using the file indicated as a parameter, and it returns the response of the JGXYZ system. It invokes the Axiom 7; xii) modelToDOT(Model, String):void: This method saves a DOT file with the data model supplied as a parameter that describes the graph model; xiii) outputInformation(String, String, Model...): Model: This method takes the data models sent as parameters and joins them in a single one, which then saves it in the indicated format, and in the indicated file.

5 Experimentation

5.1 Description of the Test Scenarios

Test scenario using first-order logic and Linked Data

The following situation was proposed: A group of users wants to play soccer, so they invite other friends (User10, User11, User12 and User13) with our system, to ensure that the guests have the right clothes to play with. Also, with our system, they check that places provide a soccer ball to play with.

Test Scenario using Dialetheic Logic

To test the axioms that make use of the Dialetheic logic, are instantiated the two axioms using JGXYZ system. For example, for Axiom 4 (see Table 1), we use the next context: there is a concert at a certain time, a user is in a nearby location, and there is little time for the concert.

5.2 Experiments with the Test Scenarios

Test Scenario using First-Order Logic and Linked Data

For the test scenario, a class is created that performs the queries (see Figure 4). This class is an example of how an algorithm that would like to use the services of our system should invoke it. In this experiment were created instances of Users (User1, User2, User3, User10,..., User14), activities (CenaFamiliar and juegoDeFutbol), the devices (balonDeFutbol, chanchaDeFutbol, and ropaDeportiva), and location (canchasDeFutbol), of a Service (hacerDeporte).

```
public class Case2 {     public static void main(String[] args) {
      org.apache.log4j.BasicConfigurator.configure(new NullAppender());
      System.out.println("Segunda parte del caso de estudio");
      Prototype caso2 = new Prototype("../ontology/pruebas/futbol.owl");
      Model canPlay = caso2.usersWhoCanConsumeServices();
      Prototype.modelToDOT(canPlay, "./resources/pruebas/caso2/canPlay");
      Model whoCantPlay = caso2.activitiesThatCantBeConsumedByUsers();
      Prototype.modelToDOT(whoCantPlay, "./resources/pruebas/caso2/whoCantPlay");
      Model whereIsABall = caso2.devicesLocatedInPlaces();
      Prototype.modelToDOT(whereIsABall, "./resources/pruebas/caso2/whereIsABall");
      Model whoLendsFields = caso2.locationProvidesService();
      Prototype.modelToDOT(whoLendsFields, "./resources/pruebas/caso2/whoLendsFields");
      Model information = Prototype.outputInformation("./resources/pruebas/caso2/information.owl",
            "N-TRIPLES", canPlay, whoCantPlay, whereIsABall);    }}
```

Figure 4: Programmed class for the test.

In this case, first, the users want to know if their guests have the clothing to be able to play. This is done using the method usersWhoCanConsumeServices, which generates the Figure 5. In the yellow circle in this figure are the users that can play, with a labeled arc: CanConsume, with the node: hacerDeporte. Finally, users want rent a ball to play. For that, they use the DevicesLocatedInPlaces method, which generates the Figure 6 that indicates who rents a ball (see yellow area).

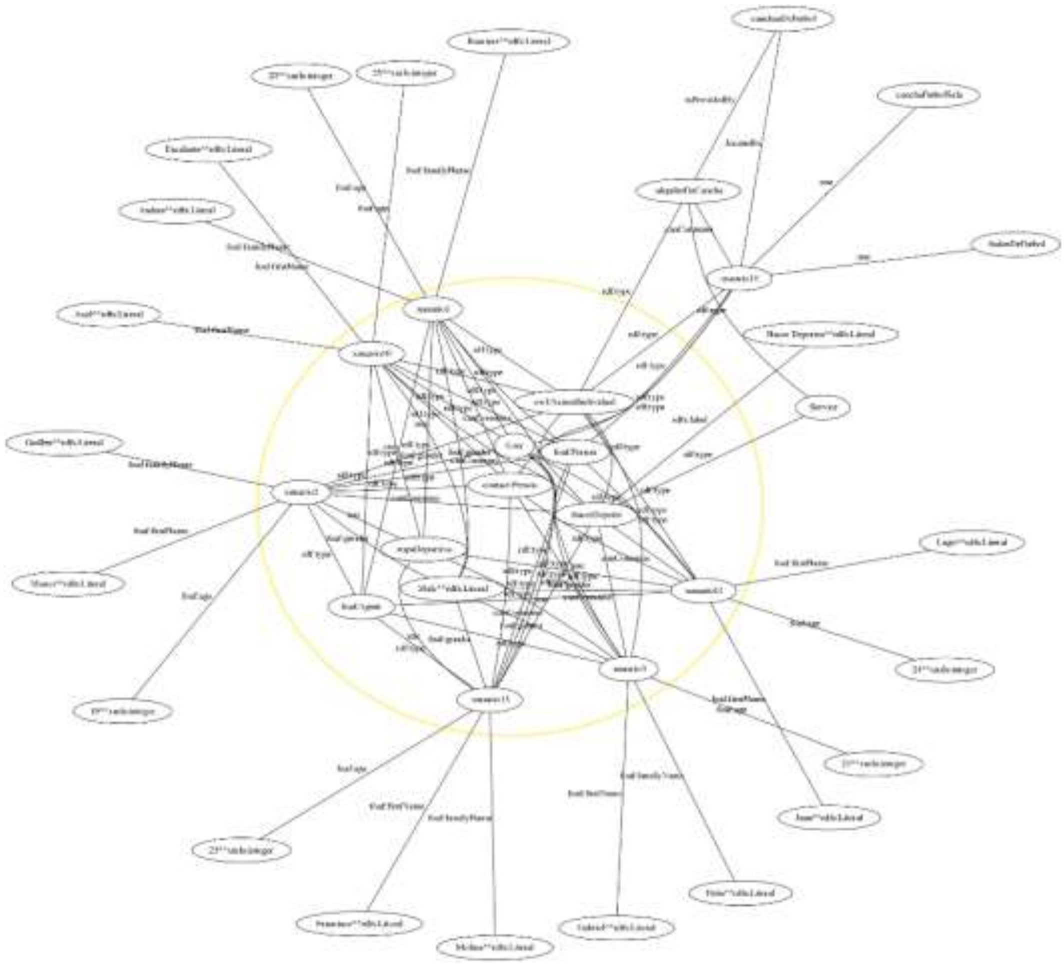


Figure 5: Result of the UsersWhoCanConsumeServices method.

Test Scenario Using Dialetheic Logic

For the case study of the Axiom 4, a simple class was developed, using the UserCanGoToActivityNearHisLocation method (see Figure 7). The file that describes the context in which the Axiom 4 will be evaluated, in the format accepted by the JGXYZ system, can be seen in Figure 7.

In the context of JGXYZ, the term “axiom” defines statements. Figure 8 contains in the axioms 1, 2, 3, 4, 5, and 6, the statements of instances of the CAMeOnto classes, which represent the facts (individuals) that describe the context in that Axiom 4 will be evaluated (instantiated). In axioms 7, 8, 9, 10, 11, and 12 are the relationships between the instances, required by Axiom 4 to describe the context.

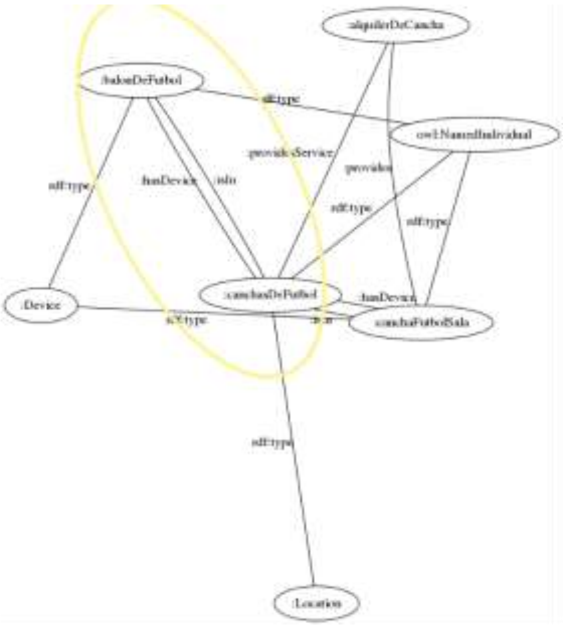


Figure 6: Result of the DevicesLocatedInPlaces method

```
public class Case3 {    public static void main(String[] args) {
        org.apache.log4j.BasicConfigurator.configure(new NullAppender());
        Prototype caso3 = new Prototype();
        System.out.println("Resultado de evaluar el axioma 4:");
        System.out.println(caso3.
              userCanGoToActivityNearHisLocation("./resources/ld/problemas/pruebaAxioma4.p"));    } }
```

Figure 7: Class programmed for the first test using Dialetheic logic.

```
fof(1, axiom, activity(concierto) ).
fof(2, axiom, location(ubicacionDelConcierto) ).
fof(3, axiom, location(ubicacionCercana) ).
fof(4, axiom, time(horaDelConcierto) ).
fof(5, axiom, time(horaActual) ).
fof(6, axiom, user(usuario) ).
fof(7, axiom, locatedIn(usuario, ubicacionCercana) ).
fof(8, axiom, starts(concierto, horaDelConcierto) ).
fof(9, axiom, isIn(concierto, ubicacionDelConcierto) ).
fof(10, axiom, soon(horaActual, horaDelConcierto) ).
fof(11, axiom, close(ubicacionCercana, ubicacionDelConcierto) ).
fof(12, axiom, isTime(usuario, horaActual) ).
fof(userCanGoToActivityNearHisLocation, axiom,(
   ?[User, Location2, Time1, Activity, Location1, Time2]:
   (( user(User) & location(Location2) & time(Time1) & activity(Activity)  & location(Location1) & time(Time2)
   & locatedIn(User, Location2) & starts(Activity, Time1) & isIn(Activity, Location1) & soon(Time2, Time1)
   & close(Location2, Location1) & isTime(User, Time2) )  => goes(User, Activity) ) )).
fof(conjecture, conjecture, goes(usuario, concierto)).
```

Figure 8: Instance of the axiom 4 (see Table 1).

5.3 Result analysis

First-Order Logic and Linked Data Results

As a measure of the quality of the context ontologies that are obtained, we will use the number of nodes and the amount of edges that are obtained by using only first-order logic, to be compared against using first-order logic with linked data. The number of nodes and the number of edges represent knowledge, so the greater the number of edges and nodes added, then more knowledge is added, as long as the information on the nodes and edges is relevant and not redundant. The results of the case study are shown in Table 2.

The results with first-order were obtained using the SWRLs previously described. According to Table 2, the amount of knowledge (nodes and edges) is always greater in cases where linked data are used. The nodes and edges generated represent knowledge, because nodes and edges alone are data only, but when a pair of nodes is joined with an edge, knowledge is generated. Linked data enriches the information using the semantic web. Let's take Figure 6 as an example. If we only take the information obtained with first order logic, we obtain the information shown in Figure 9. Looking at Figure 9, we can see that the information obtained is very simple, the location CanchasDeFutbol has BalonDeFutbol and CanchaFutbolSala. On the other hand, in Figure 6, with the enriched data, we can see that in addition, this location offers the service of ball rental, services required by the users.

Table 2: Comparison of results for the second part of the case study.

Methods	First Order Logic		With Linked Data	
	Nodes	**Edges**	**Nodes**	**Edges**
usersWhoCanConsumeServices	9	7	39	72
devicesLocatedInPlaces	3	2	7	12
locationProvidesService	2	1	7	8
activitiesThatCantBeConsumedByUsers	2	1	14	15

Figure 9: Information with the DevicesLocatedInPlaces method using first-order logic.

Results with Dialetheic Logic

For the axiom, the response generated by our system was CounterSatisfiable. The documentation of the system *JGXYZ* indicates that the system receives as input a problem F of the form $Ax \Rightarrow C$, where Ax are a set of logical formulas that must be fulfilled, so that C is inferred, and C is a formula of output [6]. *JGXYZ* evaluates that *Ax* entry in a set of facts (instances) to determine the veracity of the F problem. Later, the documentation of the JGXYZ system indicates that this status (CounterSatisfiable), is generated when "some of the axioms Ax allow corroborating or not the conjecture C". Within the framework of JGXYZ, F may be invalid, or satisfiable in the context where it is being instantiated. In our case, the two examples of instantiation of each axiom gave undeniable. This result indicates that was found a counter example of the conjectures. For example, in the case of the Axiom 4: goes(user, concert).

6 Conclusions

The semantic Web is a powerful tool, which has incredible amounts of information. Linked Data helps exploit this information. On the other hand, in a world where more and more data are generated, technology needs to use this data to create knowledge, which can be used to adapt the applications. Context-aware applications are based on this principle. In turn, it is common to have ambiguities in the real world, where in many cases the truth is only absolute for each individual. In these cases, there is not a single correct answer, and this is what the Dialetheic Logic approach seeks to model. This paper extends CAMeOn with these principles.

At the beginning, this ontology was extended with some known vocabularies of the Semantic Web, thus allowing the association of its classes with the existing classes in the Semantic Web. In this way, it was extended with linked data, allowing the creation of more information with machine-readable semantic meaning. Also, generic axioms have been proposed, based on First Order Logic. However, there are cases in which First-Order Logic is not enough to describe the situation with certainty, and another approach was considered for the ambiguous situations. This approach is based on Dialetheic Logic that produces inference that is no longer true in all cases, but only in the context of some of the users. Finally, the system makes available the extracted knowledge as a service, to any location-aware application that requires it. There is not studies that include all these areas together, and exploit the great potentials of each area.

This paper is based on two previous works, the CARMiCLOC Middleware [1] and the CAMeOn ontology [2]. Our proposed system adds new characteristics: linked data, and the handling of ambiguities using Dialetheic Logic, which extend the data management, to improve the services of analysis of context that provides CARMiCLOC. Our results show that the extensions to CAMeOn improve its capabilities of inference about the context. Particularly, the extension with linked data allows a semantic enrichment, which generates a more exact information about the context, and mainly, inferring more contextual information. With respect to the dialetheic logic, it allows managing the ambiguous information about the context during the reasoning processes. Both extensions are very useful for micro-location applications aware of the context, because linked data can include additional information (for example, if a location offers the service of rental of the ball, in our case study) that is used for a better decision-making process (for example, to select a place with these service); and dialetheic logic can solve conjecture in runtime, to cover all ambiguity situations.

Acknowledgements. We want to thank the professor Geoff Sutcliffe, of the University of Miami, for his comments and revisions to improve the paper.

References

[1] Aguilar, J., Jerez, M., Exposito, E., and Villemur, T., CARMiCLOC: context awareness middleware in cloud computing, in: *Latin American Computing Conference* (CLEI), (2015), 532-541. https://doi.org/10.1109/clei.2015.7360013

[2] Aguilar, J., Jerez, M., Rodriguez, T., and Exposito, E., CAMeOnto: Context awareness meta ontology modeling, *Applied Computing and Informatics*, **14**(2) (2018), 202-213. https://doi.org/10.1016/j.aci.2017.08.001

[3] Brickley, D. and Miller, L. Friend of a friend, 2000. http://xmlns.com/foaf/spec/.

[4] Musen, M., The protege project: A look back and a look forward, *AI Matters*, **1**(4) (2015), 4-12. https://doi.org/10.1145/2757001.2757003

[5] Pelletier, F., Sutcliffe, G., Hazen, A., Automated Reasoning for the Dialetheic Logic RM3, in: *13th International Florida Artificial Intelligence Research Society Conference,* (2017), 110-115.

[6] Sutcliffe, G., The TPTP Problem Library and Associated Infrastructure, From CNF to TH0, TPTP v6.4.0, *Journal of Automated Reasoning*, **59**(4) (2017), 483-502. https://doi.org/10.1007/s10817-017-9407-7

[7] Aguilar, J., Jerez M., Sánchez M., Mendonca M., Emergencia Ontológica Basada en Análisis de Contexto, como servicio para Ambientes Inteligentes, *DYNA*, **84**(200) (2017), 28-37. https://doi.org/10.15446/dyna.v84n200.59062

[8] Vizcarrondo J., Aguilar, J., Exposito E., Subias A, ARMISCOM: Autonomic Reflective MIddleware for management Service Composition, in: *4th Global Information Infrastructure and Networking Symposium* (GIIS), IEEE Communication Society, 2012. https://doi.org/10.1109/giis.2012.6466760

[9] Alirezaie M., Renoux J., Köckemann U., Kristoffersson A., Karlsson L., Blomqvist E., Tsiftes N., Voigt T., Loutfi A., An Ontology-based Context-aware System for Smart Homes: E-care@home, *Sensors.* **17**(7) (2017), 1586. https://doi.org/10.3390/s17071586

[10] Bourougaa-Tria S., Seridi-Bouchelaghem H., Mokhati F., An Ontology-Based Context Model to manage Users Preferences and Conflicts, *Informatica*, **40**(1) (2016), 71–94.

[11] Giustozzi F., Saunier J., Zanni-Merk C., Context Modeling for Industry 4.0: an Ontology-Based Proposal, *Procedia Computer Science*, **126** (2018), 675–684. https://doi.org/10.1016/j.procs.2018.08.001

[12] Catania B., Guerrini G., Yama B., Context Aware Source Selection for Linked, in: *23rd Italian Symposium on Advanced Database Systems*, 2018.

[13] Taneja K., Zhu Q., Duggan D., Tung T., Linked Enterprise Data Model and Its Use in Real Time Analytics and Context-Driven Data Discovery, *In IEEE International Conference on Mobile Services*, (2015), 277-283

[14] Zafari F., Papapanagiotou I., Christidis K., Microlocation for Internet-of-Things-Equipped Smart Buildings, *IEEE Internet of Things Journal*, **3**(1) (2016), 96-112. https://doi.org/10.1109/jiot.2015.2442956

[15] Pešić S., Radovanović M., Tošić M., Ivanović M., Iković O., Bošković D., Bluetooth low energy microlocation asset tracking (BLEMAT) in a context-aware fog computing system, in: *International Conference on Web Intelligence, Mining and Semantics*, 2018. https://doi.org/10.1145/3227609.3227652

Received: May 21, 2019; Published: June 14, 2019