IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 6, 2018, accepted September 19, 2018, date of publication October 1, 2018, date of current version October 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2872812

Software Experience for an Ontology-Based Approach for the Definition of Alarms in Geographical Sensor Systems

EVELIO GONZÁLEZ, ROBERTO MARICHAL, AND ALBERTO HAMILTON
Departamento de Ingeniería Informática y de Sistemas, Universidad de La Laguna, CP 38200 San Cristóbal de La Laguna, Spain
Corresponding author: Evelio González (ejgonzal@ull.es)

ABSTRACT This paper presents a system based on ontologies for the definition of alarms in sensor systems. Although we consider that the ontology and the system are interesting themselves, we have a special impact on their experience with the software tools used and the technology related to the Semantic Web. This approach is intended to illustrate the process for those readers not familiar with this field and who can consider the use of ontologies for their next developments. An ontology has been designed in the OWL language, complemented with Semantic Web Rule Language rules and search elements in the Semantic Query-enhanced Web Rule Language. The set has offered satisfactory results in simulation. Other interesting contributions of this paper are: a survey of the available literature in the field of the use of Semantic Web technologies and ontologies for the detection of events from data obtained from sensors and a study on the tools and vocabularies to be used to create a system that interfaces with ontologies and an educational method—reporting our experience—to help university students understand this topic.

INDEX TERMS Applications, decision support, education, ontology, semantic sensor network, software experience.

I. INTRODUCTION

When working with sensors, one of the interesting aspects to put in place is a system of alarms in the face of anomalous or potentially dangerous situations detected, both for the device itself or for the population in general. Managing those alarms must be done in a suitable way to be effective. This paper presents the development of a general application of alarms in sensor systems. For this, the authors will use an ontology and several technologies associated with what is called the Semantic Web. The authors believe that these tools can be very useful in sensor-based systems since the very functioning of these systems implies, by their very nature, a large amount of information and that the domain of knowledge is well structured.

First of all, the authors consider that they should briefly explain the concept of ontology to the reader not familiar with the term and thus ease the understanding of the article. Today's world needs that existing data are really understandable by many applications (related or not to the concept of Semantic Web). For this, semantic interoperability is necessary. The current syntactic interoperability is simply the correct processing of the data. So, semantic interoperability requires an analysis of the content. This translates into the need for formal and explicit specifications of the domain models that define the terms used and their relationships in a conceptual structure that stores the knowledge representation of a domain.

Several authors have proposed different definitions of the term. The best known in this respect is that of Gruber [1], which defines an ontology as "*a specification of a conceptualization*", understanding conceptualization as a vision of the world with respect to a certain domain and that is independent of language and tools. This definition is usually considered quite ambiguous and other authors have proposed their own definitions. In this sense, the authors will mention those of Guarino [2] ("*an ontology is a representation of a conceptual system by means of a logical theory*") and Swartout, Patil, Knight and Russ [3] ("*an ontology is a hierarchically structured set of terms that describe a domain, which can be used as an initial skeleton for a knowledge base*"). These alternative definitions tend to emphasize several aspects of ontology [4]: explicit representation of knowledge, the

underlying idea of associated logical theory (usually takes the form of a first-order logic where vocabulary words appear as names of unitary or binary predicates, respectively called concepts and relations), agreement on the meaning of the terms of the domain among the users of the ontology and, from this agreement, the reuse for the design of new applications.

In the context of this article, the authors prefer to highlight the definitions proposed by Blomqvist and Öhgren [5] which indicate that an ontology is ''a hierarchically structured set of concepts describing a specific domain of knowledge that can be used to create a knowledge base'' and that of Kashyap *et al.* [6] who define the ontology as ''a set of terms of interest in a particular information domain and the relationship between them''. These two definitions are more adjusted, in the authors' criterion, to work presented below. In any case, it is undeniable that it is encompassed within the development of new forms, more and more complete, of expression of knowledge, which allow collecting the semantics of the domain in its cognitive model. The main core of an ontology is the vocabulary that covers knowledge about the application domain. However, this vocabulary must be complemented with other elements that reflect the underlying conceptualization of the application domain so that the ontology is really useful: a hierarchy of the elements, relationships between the concepts, axioms that are valid in the domain, among others. As result, an ontology is usually formed by a set of concepts (entities, processes, attributes, etc.) together with their definitions and interrelationships [7].

The benefits offered by ontologies have been widely described in the literature on this matter [8]: formalization of knowledge in a way understandable by computers, deduction of new knowledge from existing statements, common expression of the domain for various actors in the pursuit of a common goal, etc. In this sense, the main aim of the paper is not to justify the use of ontologies in systems with sensors. Regarding this issue, there are already numerous successful applications in the literature.

Although the authors consider that the implementation of this ontological alarm system is interesting in itself, the article also focuses on describing its experience in the process of integrating existing ontologies related to different fields of development, which can illustrate the reader in a future choice of ontologies as a tool in their applications. In this sense, the authors miss work with this orientation in the literature on the subject. The document also reflects a pilot experience based on the development of the application based on ontologies. This experience has been carried out with a small number of students in a postgraduate course.

The remainder of the paper is as follows. Section II presents a series of recent related work on the use of ontologies and systems with sensors. Most of these references are intentionally general since they seek to illustrate a non-exhaustive analysis of the trends in the use of tools in the field. Section III focuses on the analysis of the references found, the extraction of procedures and useful resources for the implementation of the application presented in this paper. Section IV will present the results obtained and their performance in aspects such as the scalability of their use. Finally, Section V will briefly present those highlights of the integration of the method presented in teaching.

II. RELATED WORK

There are several references in the literature about the use of ontologies in the context of sensors. As a sign of the interest towards the use of ontologies in systems with sensors, the results of searches in three scientific databases are shown in Fig. 1: Web of Science, Scopus and Google Scholar. In Web of Science and Scopus (Fig. 1a), the search was made with the words ''ontology'' and ''sensor'' in the title, abstract and keywords of the item. In Google Scholar the search was throughout the entire document. Without ignoring the possible serious limitations of this procedure, the graphs show a trend of growth in research that uses ontologies in systems with sensors, although in Fig. 1a a slight decrease can be deduced as of 2015.

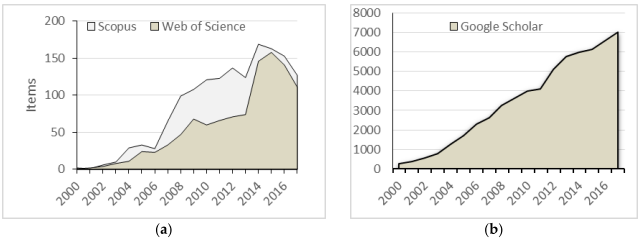

FIGURE 1. Ontology and sensors items provided by Scopus and Web of Science (April 2018); (b) Items provided by Google Scholar (April 2018).

An in-depth analysis of the search results is outside the scope of this paper. However, a non-exhaustive summary of recent related work is shown in the following, where significant articles have been selected (related articles published since 2016). This selection allows analyzing aspects related to this work.

Salguero *et al.* [9] use ontologies to automatically generate inputs for supervised learning algorithms to produce a classification model in the recognition of activities of daily living. These inputs are class expressions generated by combining the entities of the designed ontology. For this purpose, Description Logic (DL) operators are used with the purpose of combining both concepts and properties from the ontology. Xu *et al.* [10] propose an ontology-based method for fault diagnosis of loaders. This method includes an ontology-based fault diagnosis model, Case-based Reasoning for an effective and accurate fault diagnoses and ontology based RBR (rule-based reasoning) using Semantic Web Rule Language (SWRL) rules. Jin and Kim [11] have designed an ontology for e-Health data for a system based on Semantic Sensor Network (SSN) using IETF YANG. The resulting prototype comprises several sensor types: body temperature, blood pressure, electromyography, and galvanic skin response. The model includes a YANG-to-JSON translator.

Alirezaie *et al*. [12] have developed an ontology-based reasoning framework for querying satellite images for disaster monitoring. The designed framework is in charge of transform satellite imagery data into an interactive map ready to be queried. For this purpose, they implement an extension of an existing ontology called GeoSPARQL. Based on this ontology, the system is able to automatically query the classified regions based on previously defined criteria. Maleki *et al*. [13] propose a tailored ontology supporting sensor implementation for the maintenance of industrial machines. This ontology can be used for the query and classification of a wide range of sensor types: radiation detectors, chemical sensors, etc. Li *et al*. [14] have implemented an ontology for the cooperation of underwater robots, since in this context a common language for to exchanging information with unambiguous meaning is needed. These authors propose a networked ontology to address information heterogeneity, and that is based on ontology constructs defined in the PR-OWL ontology. The terms of the ontology are related to ''the mission and planning, the robotic vehicles, the environment recognition and sensing, and the communication and network domains.''

Villalonga *et al*. [15] have designed an ontology-based high-level context inference for human behavior identification. This method combines cross-domain low-level primitives of behavior (activity, locations and emotions) in order to extract high-level context information. Moreover, the authors describe a framework built on the developed ontology and reasoning models. Fernandez *et al*. [16] describe an ontology-based architecture for intelligent transportation systems using a traffic sensor network. In this work, the ontology acts as a facilitator in the task of information exchange and interoperability between the different applications of the system (where sensors play a key role).

Huang *et al*. [17] present an ontology for smart home management. The authors implement a prototype where a reasoner can make decisions about risk and safety from the semantic knowledge (including structure and state information) stored in the ontology and the definition of semantic rules. Moreover, the authors claim that the use of an ontology has implied that the system is more open, since entities can be added or removed at any time, according to the needs of the smart home. Crispim-Junior *et al*. [18] have designed a framework for the recognition of daily activities of senior people in assisted living scenarios, combining an ontology to model daily living activities with a system for people detection and tracking based on color-depth signals. Alti *et al*. [19] develop an autonomic context-aware platform for mobile applications in pervasive environments based on semantic web technologies.

Apart from these references on the use of ontologies in the management of information obtained through sensors, the authors will include in this section of related work, two more specific references, and thus closer to the area of interest of the proposal presented in this paper, although there are more in the literature. These two references, which the authors consider to be very significant, focus on alarm management systems using ontologies.

Xu *et al*. [20] define an ontology for a rollover monitoring and decision support for the rollover risk of engineering vehicles. As in the case that will be seen in this paper, the ontology is enriched through a system of semantic searches and the inclusion of rules that are triggered. In this case, there are two simple semantic rules that set the alarm level in case of an incident, since the relationship between cause and alarm level is previously defined in the ontology. This reference, and others that follow the same methodology, although valuable, focus on specific aspects, creating most of the time ad hoc ontologies (missing the potential of the reuse of existing ontologies).

Wang *et al*. [21] develop a web ontology of hydrological sensor with a method closer to the one followed in this paper. They reuse two ontologies (one on sensors and another ontology on geographic elements) and establish reasoning rules. These rules are related to the detection of possible floods from the monitoring of the water level in rivers and other sources.

In this paper, a more general system of alarms is intended from the information provided by sensors, integrating previously defined ontologies. In addition, as indicated above, an important aspect of the paper is to highlight the software experience in the application development process.

III. MATERIALS AND TOOLS

This section will be divided into two subsections that will justify the choice of tools and methodologies for implementing the application. The authors will start with the general elements for work with ontologies and will continue with the more specific existing elements for the integration of systems based on sensors. It is emphasized that this analysis and implementation are included in the classical methodologies of work with ontologies such as the one indicated in [7]. The elements implemented in the system will be included in subsequent subsections.

A. TOOLS FOR ONTOLOGIES

As a starting point, the authors have analyzed the references cited in section 2 on specific aspects of implementation of ontology + sensor systems: domain of the resulting application, ontology implementation tool, coding language, integration or not with ontologies pre-existing and other technologies / software used in the implementation. Table 1 collects information on these data, from which useful information can be extracted for the experience that is sought in this paper.

The first conclusion from this analysis is that Protégé [22] is the tool used in almost all cases for the development of ontologies. Although there are other ontology editors (the authors will cite the cases of KAON2 [23], Hozo [24], FluentEditor [25], OWLGrEd [26], OntoStudio [27], TopBraid Composer [28], Swoop [29], Vitro [30]) none of them is so popular among developers and researchers. As a significant fact, it can be shown in Table 2 the refer-

TABLE 1. Analysis of references cited in Section 2 (ontology+sensors papers 2016-2018).

Authors	Subject	Ontology Tool	Ontology Language	Previous ontologies	Other ontology-related technologies
Huang et al. (2016)	Smart Homes	Protégé	OWL	SSN	SWRL SWRLJessTab Jess
Fernández et al. (2016)	Intelligent Transportation	Protégé	OWL-RDF	SSN	Pellet SPARQL
Villalonga et al. (2016)	Human Behavior Identification	Protégé	OWL-2		SPARQL Pellet
Li et al. (2017)	Cooperation of Underwater Robots	Protégé	OWL	PR-OWL	SWRL
Maleki et al. (2017)	Maintenance of Industrial Machines	Protégé	OWL	SSN	DL Query
Alirezaie et al. (2017)	Satellite Images for Disaster Monitoring	Protégé	N/A	GeoSPARQL DOLCE+DnS Ultralite (DUL)	
Crispim-Junior et al. (2017)	Recognition of Daily Activities	N/A	N/A	N/A	
Alti et al. (2017)	Mobile Applications	Protégé	OWL DL		SWRL, Jess
Salguero et al. (2018)	Recognition of Activities of Daily Living	N/A	OWL		OWLExpand OWLVectorize
Xu et al. (2018)	Fault Diagnosis of Loaders	Protégé	OWL DL		SWRL SPARQL Jena
Jin and Kim (2018)	e-Health	Libyang	YANG	SSN	

ences found by each of the editors in Scopus (retrieved on April 2018) and the year in which the most recent document was published. It is also considered significant of Protégé relevance that several of the other editors are promoted in their respective web pages indicating that they have a communication interface with Protégé. Nevertheless, an in-depth analysis of the ontology editors is out of the focus of this article. The interested reader can consult other references in this regard.

TABLE 2. References found in Scopus regarding ontology editors (Title AND Abstract AND Keywords) (April 2018).

Ontology Editor	Documents	Year of the last document
Protégé	1,137	2018
Hozo	39	2018
KAON2	20	2016
OWLGrEd	17	2017
Swoop	13	2017
OntoStudio	5	2014
TopBraid Composer	5	2017
FluentEditor	1	2013
Vitro	1	2011

Protégé is a free and open-source platform for the construction of domain models and knowledge-based applications with ontologies. Its first version, Protégé-2000, was developed by the Medical Informatics Group of Stanford University to support the acquisition of knowledge for medical expert systems, although currently, it has become very popular for many other purposes. In fact, one of its strongest points is its high number of users (more than 320,000 - April 2018) and its very active community. The reader can access a detailed list of features of Protégé in [31], although in the context of this paper the authors will highlight the following: OWL API for OWL 2.0, loading of multiple ontologies into a single workspace, handling disjoints / different instances, various transforms on restrictions (including covering), DL Query tab for testing arbitrary class expressions, OWL description parsing, direct interface to ontology reasoners (HermiT [32], FaCT ++ [33], Pellet [34], RACER [35], among others), SWRL rules editing and a highly pluggable architecture. Its interface to apply semantic reasoners can be used, among other objectives, to check the consistency of the designed ontology. Protégé 3.4.1 should be enough for the requirements of the proposed problem and it has a more friendly SWRL rule editor (as seen in Fig. 2) than in later versions. However, last version (5.2, updated in March 2017) has been used in the presented work since it is able to treat more effectively the import of ontologies, handle ontologies stored with Turtle syntax and it provides more direct interaction with other powerful APIs. At this point, the authors want, from their experience, to discourage the use of several different versions of the tool during the implementation of the ontology, since this behavior frequently produces undesired effects.

OWL (Ontology Web Language), as in the case of other ontology languages (for example, DAML + OIL), is an extension of the RDF Schema, whose structure definition dates from 2003 [36]–[38]. The purpose of this language is similar to that of RDF Schema: to provide an XML vocabulary for the definition of classes, their properties and the relationships between classes. However, it allows the user to express relationships of greater semantic richness, thus

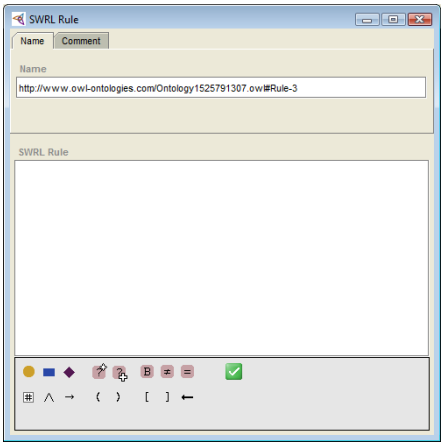

FIGURE 2. SWRL Editor in Protégé 3.x.

providing a greater capacity for inference to the processing of a document. In this sense, it is the most advanced step towards the Semantic Web. In a scheme of evolution of the languages of marks [39]:

- XML provides a superficial syntax for structured documents, but does not impose any semantic restrictions on the meaning of these documents.

- XML Schema is a language to restrict the structure of XML documents.

- RDF is a data model for objects (resources) and relationships between them, providing simple semantics for this model and being able to be represented with XML syntax.

- RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for hierarchies of generalization of said properties and classes.

- OWL supposes an even greater enrichment of the vocabulary, including for example the definitions of the symmetrical properties and functional inverse.

W3C provides three OWL sublanguages: OWL Full (totally compatible with RDF but undecidable), OWL DL (that offers the greatest expressiveness without losing computational completeness and decidability) and OWL Lite (with a limited set of constructors) [40]. In addition, there are languages to perform searches on an ontology, two of which are the most used. On the one hand, SPARQL (recursive acronym for SPARQL Protocol and RDF Query Language), a subset of RDF [41]. This language presents the problem (as stated in [42]) an incomplete understanding of OWL semantics, although it offers acceptable results in most cases when the ontology is expressed in RDF serialization. The other language is SQWRL (Semantic Query-enhanced Web Rule Language) [43], a specific search language in OWL. These searches are carried out using built-ins, with which the syntax is more compact and simple to use.

Once the ontology has been designed, if users want to take advantage of it, it is necessary to process it in order to integrate it within real applications. In this sense, the authors state that this integration can be done independently of the plugins offered by the Protégé tool or not. In this regard, there are tools and resources that facilitate this integration work. The authors have used the following resources in the system presented in this paper, which, from their experience, are very suitable for this purpose. However, the reader can find alternative resources that can fulfill the same purpose or give new use to the tools described below.

The inclusion of ontological deduction rules can be a powerful method for developed applications. W3C proposes the Semantic Web Rule Language (SWRL) for that purpose. SWRL [44] is based on a combination of subsets of the OWL and RuleML (Rule Markup Language) [45] languages in order to include the Horn-like rules interaction with statements expressed in OWL. There are two circumstances to consider when working with SWRL that differentiate it from classical logic programming: the open world assumption (a statement may be true irrespective of whether it is explicitly declared to be true) and the monotonic inference (a SWRL rule cannot modify / include / remove any existing statement in the ontology). At the implementation level there are several alternatives to work with SWRL: SweetRules [46], ROWL [47], but the preferred one in this paper is the definition directly in Protégé through the SWRLTab. In this way, it is integrated with the tool chosen for the definition of the ontology and the support of the Protégé user community is used. As it is verified in the literature on the subject [43], [48] and it will be seen later, this procedure adapts to the type of applications that the authors try to achieve in this paper.

With the sole purpose of facilitating the reading of the following sections, the authors will show a simple example of a SWRL rule. An expression like the following

$$\text{Class}(?x) \wedge \text{hasProperty}(?x,?y) -> \text{hasOtherProperty}\ (?x,?z)$$

should be interpreted (without going into other possible details) in the following way. If ?x is an instance of the Class class and also has a value ?y for the hasProperty property, then ?x has a value ?z for the hasOtherProperty property. A typical example is the following [49], since an OWL ontology cannot express directly the hasUncle property as a combination of hasParent and hasBrother properties:

$$\text{hasParent}(?x,?y) \wedge \text{hasBrother}(?y,?z) - > \text{hasUncle}(?x,?z)$$

The nature of the problem posed, the possibility of using ontologies and SWRL and its temporal response needs, naturally raise the possibility of using an implementation based on facts and rules, such as the use of CLIPS, where the ontology is mapped to facts and the SWRL rules to rules. In this respect, a powerful tool is Jess (Java Expert System Shell [50]). Although not the employee in the implementation of the Alarm System, the authors consider it useful to detail some aspects of it, according to the line of explaining its software experience. In this way, relevant aspects applicable to the tools finally chosen are also presented.

Jess is a Java-based rule engine and scripting environment that can be used in order to ''reason'' from knowledge expressed in the form of declarative rules, with a syntax

much like CLIPS (in fact, Jess language is considered as a superset of the CLIPS programming language). For that purpose, it uses a version of the Rete algorithm to process those declarative rules. A key aspect to keep in mind for the use of Jess is that it is closed-source although it is available at no cost for academic use. Jess cannot directly employ the ontology designed in OWL nor the rules in SWRL, but these must be translated into facts and declarative rules. A direct option is the use of SWRLJessBridge, one of the plug-ins offered by Protégé. Another option for the developer is that, fortunately, the facts that OWL is a markup language based on XML (formally speaking, an XML vocabulary) and that Protégé already stores SWRL rules in markup language format allow for easy transformation using parsers. In this sense, the authors also recommend two XSL transformation templates: OWL2Jess and SWRL2Jess [51]. However, these XSL templates do not manage the use of SWRL built-ins (terms defined in SWRL with the aim of preparing further extensions and which refer, among others, to comparisons and mathematical expressions), for example for the use of mathematical expressions within the rules. This aspect is very important in the development of applications such as those presented in this paper. The authors have extended SWRL2Jess template including some extra processing code. As an example, the following code refers to the built-ins swrlb:greaterThanOrEqual (which uses test order in Jess when dealing with a comparison) and swrlb:substract (which uses instead bind). These modifications are general, independently of the application, and that therefore can be used in other works. The authors have implemented the comparison built-ins [52] and mathematical expressions in the XSL template. As it can be seen, previously defined templates have been used, which allows an implementation in relatively little time and effort. However, this procedure is limited to those built-ins that have direct correspondence in the Jess language.

```
< xsl:template match = ‘‘swrl:BuiltinAtom′′
< xsl:variable name = ‘‘operation′′ >
< xsl:apply-templates select = ‘‘swrl:builtin′′/ >
< /xsl:variable >
< xsl:variable name = ‘‘operationWithoutNamespace′′
>
<xsl:value-of select=‘‘substring-after($operation,′#′)′′/>
< /xsl:variable >
< xsl:if test = ‘‘$operacionSinNamespace
=′ greaterThanOrEqual′′′ >
(test (>=
< xsl:apply-templates select
= ‘‘swrl:arguments/rdf:List′′ / >
< xsl:text / >
< xsl:apply-templates select
= ‘‘swrl:arguments/rdf:List/rdf:rest/rdf:List′′/ >
))
< /xsl:if >
< xsl:if test = ‘‘$operacionSinNamespace =′ subtract′′′ >
(bind
< xsl:apply-templates select
= ‘‘swrl:arguments/rdf:List′′/ >
(−
< xsl:apply-templates select
= ‘‘swrl:arguments/rdf:List/rdf:rest/rdf:List′′/ >
< xsl:text / >
< xsl:apply-templates select = ‘‘swrl:arguments/rdf:List
/rdf:rest/rdf:List/rdf:rest/rdf:List′′ / >
))
< /xsl:if >
<! − − rest of built-ins − − >
< /xsl:template >
```

Although the weight of the resulting application will fall on the Jess rule engine, from the application it will be necessary to directly process the ontology, for example, to manage the list of available resources or visualize the current state of the system. A recommended tool for this purpose is Jena [53], a Java framework for building Semantic Web applications, that provides many Java libraries for RDF, RDFS, RDFa, OWL and SPARQL languages. As stated in its web site ‘‘*Jena includes a rule-based inference engine to perform reasoning based on OWL and RDFS ontologies*’’ that could perform a role similar to that of Jess [54]. However, the use of this inference engine instead of the procedure described above with Jess implies their own syntax for the rules. Thus, regardless of the efficiency of the reasoner, the implementation would need to translate the SWRL rules to that new syntax or define them in the Jena's own language. It is important to note that the simultaneous use of Jena and Jess (for example with the use of SWRLJessTab) will be useful to detect situations in which the knowledge provided by the rules may conflict with that stored in the ontology (since non-monotonic features sometimes cannot be mapped with classical logic, and combining OWL DL and rules could lead to undecidability) [55]. Regarding this issue, the developer has to take into account that Jena's reasoners offer an incomplete implementation of the OWL Lite subset.

The Jess-SWRLJessBridge-Jena combination is acceptable on most occasions, especially when working with the RDF serialization. However, they can present unwanted behaviors if you want to take advantage of the logical syntax offered by OWL or simply use ontologies with OWL serialization. Moreover, last versions of SWRLTab plugin does not support Jess as inference engine. For this reason, in the development of the application, alternative tools have been used, integrated in the SWRLAPI project [42], [43]. Installation instructions can be found in [56]. It is based on

OWLAPI, an OWL-centric API that allows to manage the ontology as a set of axioms instead of a set of triples that encode the semantic information of the ontology.

SWRLAPI is a set of applications for complete management of SWRL rules. It consists of ten modules of which the SWRL Rule Engine API, SQWRL API, SWRL Built-in Libraries, SWRL Built-in Bridge and SWRL Rule Engine Bridge have been used in this work. Its structure of operation is similar to that explained for Jess and Jena, so we will not detail it at this point. However, the authors wish to note that SWRLAPI provides an interface that employs the rule engine called Drools. This is a business rule management system (BRMS) that, like Jess, uses an enhanced implementation of the Rete algorithm.

Finally, it is necessary to briefly analyze the tools that allow visualizing the results (in our case, the geographical position of the alarms produced). If the search for these results has been done in SPARQL, a good alternative is Sgvizler [57] It is a useful JavaScript library which renders the result of SPARQL queries into elements easily included in web pages. Moreover, it supports all the major chart types offered by the Google Visualization API, and it includes an exhaustive set of functions for drawing graphs. However, SWRLAPI offers powerful support for the use of SQWRL, which simplifies the search syntax. In the implementation of the system, therefore, searches have been defined in SQWRL. For visualization, the system transforms those results to KML (Keyhole Markup Language) and they are easily displayed through a web browser. For this, Leaflet [58], an open-source Javascript library for interactive maps, and the KML.js plugin are used [59].

B. SPECIFIC ONTOLOGIES FOR THE DEVELOPMENT

Once the general tools have been seen, the authors will focus on elements that are more oriented to the field of sensors. The main one is the integration of existing ontologies (as indicated previously, one of the main advantages of the use of ontologies is reusability) by extending them. This provides, as is logical to deduce, a considerable decrease in the time necessary for the development of the application at the same time as getting more robustness and consistency. However, and this is a general aspect in the field of ontologies, there is a proliferation of ontologies (sometimes even several defined by academic organizations of recognized prestige) referring to the same field. The field of sensors is not different in this regard [60], [61]. The choice of ontology to integrate/ extend may depend on several factors, including some subjective ones. As indicated in [62] there is no definitive quantitative method to establish if one ontology is better than the other as long as it responds to the needs of the developer, although in the opinion of the authors it is better to choose an ontology endorsed by a broad community.

For the application it will be considered that there are three dimensions involved: sensors, geographic elements and alarm system (this dimension will include the elements that help to integrate the previous elements). Existing ontologies will be used for the first two dimensions and new elements will be implemented for the last one.

1) ONTOLOGIES ABOUT SENSORS

At this point, an argument for the decision is to return to Table 1. Of the ten papers analyzed, in six a pre-existing ontology is used, and of those six cases, the Semantic Sensor Network Ontology by W3C is used in four [63].

This is an ontology "for describing sensors and their observations, the involved procedures, the studied features of interest, the samples used to do so, and the observed properties, as well as actuators." SSN is complemented by a core ontology called SOSA (Sensor, Observation, Sample, and Actuator) for its elementary classes and properties. To do this, eight conceptual modules are established in which each of the defined concepts is assigned. To illustrate the ontology, Table 3 reflects this distribution, taking into account that there is overlap between some of the modules (overlapped classes are in bold, and the other module is described in brackets). The properties and detailed relationships between the concepts can be found in [63].

W3C proposes two other conceptual modules (System Property and Condition), although there are no concepts directly from the SSN and SOSA cores that fall within them. By means of what W3C denominates horizontal segmenta-

TABLE 3. SSN and SOSA conceptual modules and related classes.

Conceptual module	SSN/SOSA Classes
Deployment	ssn:Deployment, sosa:Platform,
System	ssn:System
Procedure	ssn:Input, ssn:Output, sosa:Procedure
Feature	**sosa:FeatureOfInterest**(Obs./Act/Samp.), ssn:Property
Observation/Actuation/Sampling	sosa:ActuatableProperty, **sosa:Actuation**(Result), sosa:Actuator, **sosa:FeatureOfInterest** (Feature), sosa:ObservableProperty, **sosa:Observation** (Result), sosa:Sample, sosa:Sampler, **sosa:Sampling** (Result), sosa:Sensor, ssn:Stimulus
Result	**sosa:Actuation** (Obs./Act/Samp.), **sosa:Observation**(Obs./Act/Samp.), sosa:Result, **sosa:Sampling** (Obs./Act/Samp.)

TABLE 4. Example of implemented classes and properties.

Authors	Subject	Examples of implemented classes and properties
Huang et al. (2016)	Smart Homes	monitored_by, Electronic_device, acting, Home_context, Status, Device_status_door_closed…
Fernández et al. (2016)	Intelligent Transportation	Car_flow, Car_flow_value, SourceMapping, SensorAnnotation…
Maleki et al. (2017)	Maintenance of Industrial Machines	hasSpecification, SensorSpecification, MeasurementSpecification, hasWorkingCondition…
Jin and Kim (2018)	e-Health	e-Health_Device, BloodPressureResURI, BodyTemperatureResURI…

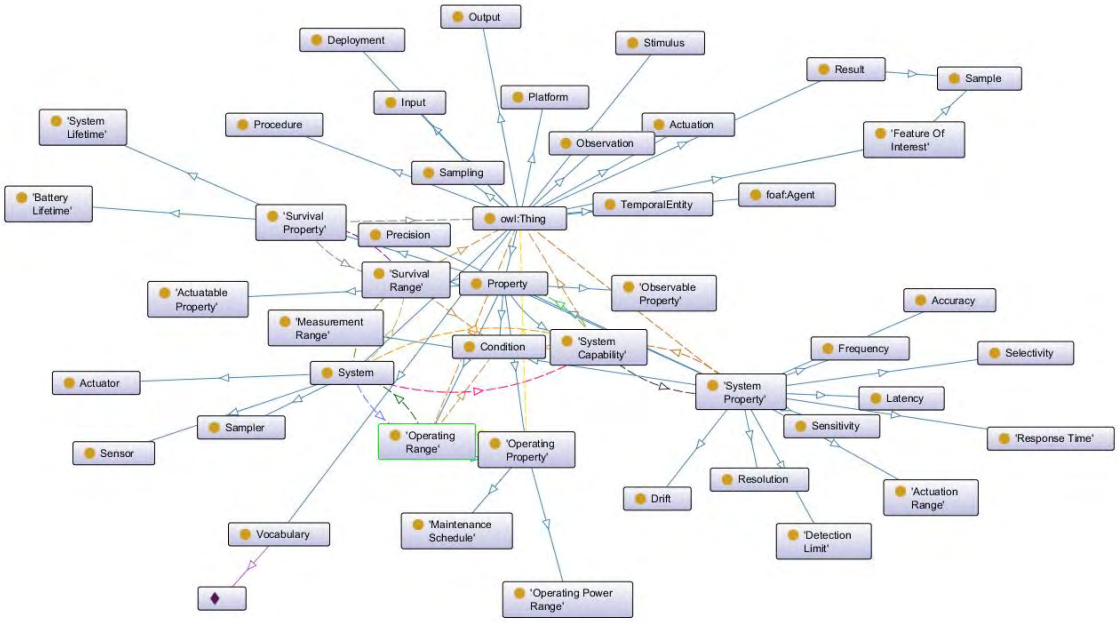

FIGURE 3. Overview of the elements of the SSN ontology with the System Capabilities Module (the authors have used OntoGraf tab for Protégé 5.0).

tion, extensions of the original cores refer for example to the system capabilities or sample relationships.

To finish this description of the SSN, the authors will analyze very briefly some examples that can illustrate the capacity of extension and integration in more complex systems. For this, Table 4 shows examples of implemented classes and properties in the works included in Table 1 that make use of SSN. As it can be deduced, the structure of the ontology allows the definition from it of elements of the domains of the application. Other interesting works using SSN are [21] and [64].

From the above, it seems that the SSN is a good candidate to be employed in the system, although possible details of implementation remain to be defined. SSN is used with assiduity in the development of applications for the management of sensors through the ontological approach.

For the development of the application, the SSN ontology with the System Capabilities Module has been selected. This module is one of those proposed by W3C within the horizontal segmentation of the SSN. This module is related to modeling "system capabilities, operating systems, and the range of tests, under given conditions" [63]. The resulting class scheme and the relationships between them are shown in Fig. 3.

2) NTOLOGIES ABOUT GEOGRAPHIC ELEMENTS

The problem posed requires that the ontology includes geographic elements that allow locating both the available sensors and the elements that trigger the possible alarms. This is because there are sensors (for example, seismographs, sensors in drones, etc.) that inform about events not located in their geographical position.

Regarding this point, it is possible to apply what is indicated in the previous section, that is, the existence of several available ontologies and the suggestion of choosing an ontology endorsed by an important community. Taking into account these circumstances, the GeoSPARQL defined by Open Geospatial Consortium (OGC) [65] will be used. Another analogous and simple ontologies are the Geo

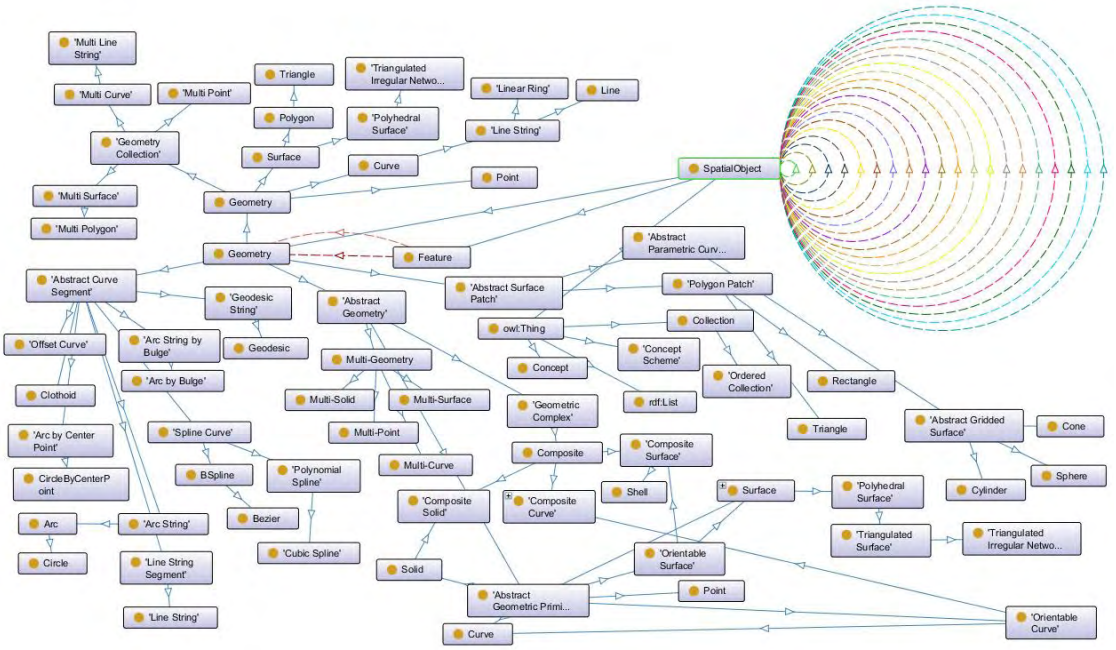

FIGURE 4. Overview of the elements of the GeoSPARQL ontology (the authors have used OntoGraf tab for Protégé 5.0).

OWL [66] and WGS84 Geo Positioning [67], both defined by W3C. Actually, for the first version of the system that includes only a geographical point expressed by its latitude and longitude, the concepts defined in the WGS84 Geo Positioning are sufficient. In future versions it is intended to use other geographic elements (as in the case of areas) provided by GeoSPARQL.

This ontology describes concepts for representation of geospatial properties and constitutes a standard for the exchange of information about these properties. This exchange of information would be done through RDF and allows reasoning and searches through SPARQL. The integration of GeoSPARQL with SSN is possible to implement as demonstrated in works such as [21]. Fig. 4 shows the schema of the classes of the ontology and the relationships between the defined terms.

3) IMPLEMENTED ELEMENTS

Before detailing the elements implemented in the system proposed in this paper, the authors will make a recapitulation that allows the reader to be situated. From the above, the scheme resulting from the implemented system is shown in Fig. 5. The proposed software tools are indicated in the gray boxes. The resulting ontology and the defined SWRL rules are loaded by the OWLAPI classes, creating an OWLOntology. Through the SWRLAPI (in particular the SWRLRuleEngine class) this model is transferred to the Drools engine, which infers new knowledge that is included back into the OWLOntology. Periodically, the results are visualized through the Laeflet software from SQWRL queries.

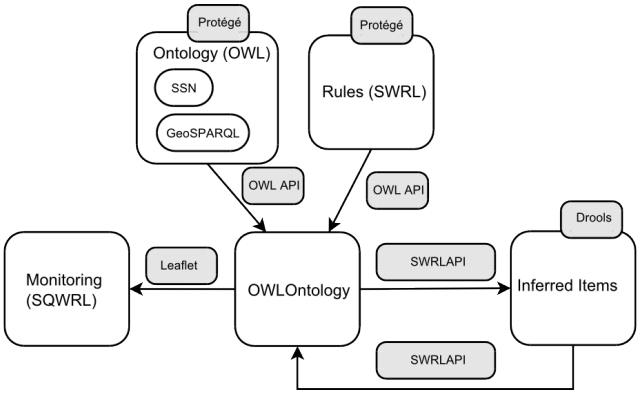

FIGURE 5. Flow in the implemented system.

For the reader interested in other options, an alternative scheme is shown in Fig. 6, where Jena, SWRLJessBridge, Jess, SPARQL amd Sgvizler are used. GeoSPARQL can be replaced by Geo OWL or WGS84 Geo Positioning ontology.

From the experience of the authors, an important advice for the use of all the indicated tools is to know the limitations of each of them, possible bugs detected and even the differences in functionality between each version. This can save a lot of time in the development of the application. A good starting point is the Protege Project forum [68].

Table 5 shows the mapping between prefixes and namespaces used by the ontology. This table will allow to distinguish the different vocabularies involved. For reasons of simplification of the ontology and its processing, the inclusion of the Dublin Core vocabulary for metadata, or that of

TABLE 5. Prefixes and namespaces for the implemented ontology.

Prefix	Namespace	Comment
Custom Namespaces for the implementation		
alarms	http://www.semanticweb.org/evelio/ontologies/2018/5/Alarms	Namespace for the implemented ontology
geo	http://www.w3.org/2003/11/geo#	Namespace for implemented custom SWRL built-ins
Namespaces for XML, XML Schema, RDF, RDFSchema and OWL vocabularies		
xml	http://www.w3.org/XML/1998/namespace	XML namespace
xsd	http://www.w3.org/2001/XMLSchema#	XML Schema namespace
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF namespace
rdfs	http://www.w3.org/2000/01/rdf-schema#	RDF Schema namespace
owl	http://www.w3.org/2002/07/owl#	OWL namespace
Namespaces for SSN/SOSA ontologies		
ssn	http://www.w3.org/ns/ssn/	SSN vocabulary
sosa	http://www.w3.org/ns/sosa/	SOSA vocabulary
systems	http://www.w3.org/ns/ssn/systems/	Namespace for System Capabilities Module (horizontal segmentation)
Namespaces related to GEOSPARQL		
geosparql	http://www.opengis.net/ont/geosparql#	GEOSPARQL namespace
gml	http://www.opengis.net/ont/gml#	Geography Markup Language (GML) namespace
sf	http://www.opengis.net/ont/sf#	GEOSPARQL Specification for simple feature geometries
Other namespaces		
core	http://www.w3.org/2004/02/skos/core#	Simple Knowledge Organization System (SKOS)

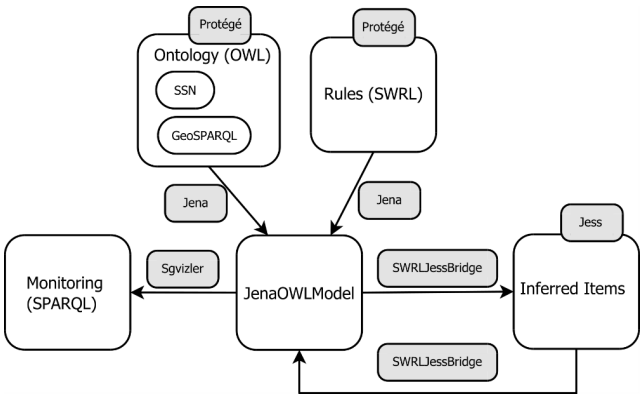

FIGURE 6. Alternative Flow in the implemented system.

TABLE 6. Defined classes and their direct superclasses.

Class	Subclass of
alarms:Alarm	owl:Thing
alarms:CombinedAlarm	alarms:Alarm
alarms:NotCombinedAlarm	alarms:Alarm
alarms:AlarmRange	owl:Thing
alarms:InfluenceRadius	owl:Thing
alarms:DistanceSensor	sosa:Sensor
alarms:InSituSensor	sosa:Sensor

TABLE 7. Defined object properties (with their range and inverse property when available).

Object Property	Range	Inverse of
alarms:hasAlarmRange	alarms:AlarmRange	alarms:isAlarmRangeOf
alarms:hasLocation	sf:Point	alarms:isLocationOf
alarms:triggersAlarm	alarms:Alarm	

TABLE 8. Defined data properties (with their range).

Data Property	Range
alarms:hasActualInfluence	xsd:boolean
alarms:hasLowerLimit	xsd:double
alarms:hasNumericValue	xsd:double
alarms:hasPriority	{0,1,2,3,4}
alarms:hasTemporalInfluenceinHours	xsd:integer
alarms:hasUpperLimit	xsd:double

TABLE 9. Defined individuals in the ontology. Prefix alarms: has not been included for the sake of simplicity.

Individual	Class
NoAlarm, Level_1_Alarm, Level_2_Alarm, Level_3_Alarm, Level_4_Alarm	alarms:NotCombinedAlarm
CombinedAlarm1	alarms:CombinedAlarm
InfluenceRadius1	alarms:InfluenceRadius

QUDT ontology to specifying quantities, units, and dimensions has not been contemplated [69].

The main elements defined in the implemented ontology are summarized in Tables 6 (classes), 7 (object properties), 8 (data properties) and 9 (individuals). The core class on which the rest of the ontology will turn is alarms:Alarm, which indicates the type of alarm produced by the sosa:Result value of an sosa:Observation. It is a subclass of owl: Thing, the root of the OWL taxonomic tree. In terms of the SHOIN(D) descriptive language:

$$\text{alarms:Alarm}^I \subseteq \Delta^I, . \tag{1}$$

This class has two subclasses, alarms:CombinedAlarm (alarm produced by the combination of the results of two different observations) and alarms:NotCombinedAlarm (generated by the result of an isolated observation). The use of alarms:CombinedAlarm is considered useful for observations made by different sensors on the same feature of interest.

An example can be the activity observations in volcanoes. In this case, it may be interesting to combine the results of different measurements (seismographs, gas emission meters, etc.).

These two subclasses are disjoint to each other and define a covering axiom on the alarms:Alarm class.

$$\begin{aligned} \text{alarms:CombinedAlarm}^I &\subseteq \text{Alarm}^I, \\ \text{alarms:NotCombinedAlarm}^I &\subseteq \text{Alarm}^I, \\ \text{alarms:CombinedAlarm}^I &\cap \text{alarms:} \\ \text{NotCombinedAlarm}^I &= \emptyset, \\ \text{alarms:Alarm}^I &= \text{alarms:CombinedAlarm}^I \\ &\cup \text{alarms:NotCombinedAlarm}^I. \end{aligned} \tag{2}$$

In a similar way, two subclasses of sosa:Sensor are defined alarms:InSituSensor (for sensors whose observations have the same physical location as the sensor) and alarms:DistanceSensor (sensors that are capable of making measurements referred to distant locations of their physical location, as in the case of seismographs). A covering axiom is also defined with these subclasses.

The ontology also includes the alarms:AlarmRange class that sets the interval of the observed magnitude that triggers a certain level of alarm. Finally, the alarms:InfluenceRadius class is defined that indicates the distance threshold to consider that an alarm:CombinedAlarm occurs. As can be seen, if the proposed methodology is followed, the definition of only seven classes is sufficient to implement the desired behavior.

Regarding object properties, first, the authors will mention alarms:triggersAlarm. This object property relates a sosa:Observation or an alarms:AlarmRange (its domain) with the level (or levels) of the alarm that triggers according to the value of its sosa:Result (property range).

$$\begin{aligned} \text{alarms:triggersAlarm}^I \subseteq (&\text{sosa:Observation}^I \\ &\cup \text{alarms:AlarmRange}^I) \text{x alarms:Alarm}^I. \end{aligned} \tag{3}$$

Another object property is alarms:hasLocation that serves to relate sosa:Observation and sosa:Sensor with its location. The rank of the property is sf:Point (consequence of the importation of the GeoSPARQL ontology). Its reverse property alarms:isLocationOf is also defined. The alarms property: hasLocation is defined as functional.

$$\text{alarms:hasLocation}^I = (\text{alarms:isLocationOf}^I)^-. \tag{4}$$

The ontology also includes the object property alarms: hasAlarmRange, which sets the alarms:Range for a sosa:ObservableProperty. The reverse property is alarms: isAlarmRangeOf.

The ontology makes use of the data properties shown in Table 8. The data properties alarms:hasLowerLimit and alarms:hasUpperLimit set the lower and upper limits for their corresponding alarms:AlarmRange. Moreover, alarms:hasNumericValue is used to express the sosa:Result of the soda:Observation.

A data property (alarms:hasPriority) is included to indicate the priority that each alarm level. It is set as an integer value between 0 (when the result of the observation does not produce an alarm to be answered) and 4 (maximum priority).

The two remaining data properties are related to the posterior temporal influence of a sosa:Observation. This reflects the fact that an observation has an influence on the analysis of the observable magnitude during a certain time. For example, a small earthquake associated with volcanic activity should be taken into account if another related abnormality occurs over a period of time. This time interval is indicated by alarms:hasTemporalInfluenceinHours. This value is related to the data property alarms:hasActualInfluence that indicates if a certain sosa:Observation has influence at the current time (in other words, if the time indicated by alarms: hasTemporalInfluenceinHours has already passed).

Finally, Table 9 shows the individuals in the ontology. Here alarms:InfluenceRadius indicates the spatial threshold that determines if two observations cause a combined alarm. Alarm levels are also defined. These individuals have been defined thinking about the subsequent incorporation of SWRL rules in the ontology. Although the rules can create individuals, this creation is discouraged by the problems they can cause at the level of reasoning and decidability in the ontology.

The ontology is completed with axioms on cardinality and universal restrictions. For example, alarms:AlarmRange has cardinality 1 for the alarms:triggersAlarm property (with alarms:NotCombinedAlarm as object).

$$\begin{aligned} \text{alarms:triggersAlarm}^I \subseteq &\text{ alarms:AlarmRange}^I \\ &\text{x NotCombinedAlarm, ,} \\ \{\text{a} \in \text{alarms:AlarmRange}^I &\,||\{\text{b}|(\text{a,b}) \\ &\in \text{alarms:NotCombinedAlarm}^I\}| = 1\} \end{aligned} \tag{5}$$

Once implemented the elements of the ontology, four SWRL rules (shown in Table 10) were defined for the treatment of the information contained. This information is updated dynamically with the appearance of new observations and the result of the application of these rules.

- Rule S1 assign an alarms:Alarm to a sosa:Observation in case the numeric value of the result (sosa:Result) is within the range defined in an alarms:AlarmRange corresponding to that sosa:Observation.
- Rule S2 indicates that a sosa:Observation made by an instance sensor of the alarms: InSituSensor class has the same location (including its object property sf:Point) as the sensor.
- Rule S3 is the most complex and defines when an alarms:CombinedAlarm is added to the ontology. Indicates that there are two sosa:Observation (with temporary influence at present) that trigger an alarm (not individuals of the alarms:CombinedAlarm class, with priority greater than 0) and with locations separated by less than a threshold (set by the individual alarms: InfluenceRadius1), the first sosa:Observation (with the

TABLE 10. SWRL rules for the implemented system.

Rule ID	SWRL rule
S1	sosa:Observation(?x) ^ sosa:Result(?y) ^ sosa:hasResult(?x, ?y) ^ alarms:hasNumericValue(?y, ?z) ^ sosa:observedProperty(?x, ?a) ^ alarms:hasAlarmRange(?a, ?b) ^ alarms:hasUpperLimit(?b, ?c) ^ alarms:hasLowerLimit(?b, ?d) ^ alarms:triggersAlarm(?b, ?e) ^ swrlb:lessThan(?z, ?c) ^ swrlb:greaterThanOrEqual(?z, ?d) -> alarms:triggersAlarm(?x, ?e)
S2	alarms:InSituSensor(?x) ^ alarms:hasLocation(?x, ?y) ^ sosa:Observation(?z) ^ sosa:madeBySensor(?z, ?x) -> alarms:hasLocation(?z, ?y)
S3	**geo:hasInfluenceNow**(?bool1, ?t1, ?z1) ^ swrlb:equal(?bool1, true) ^ alarms:hasNumericValue(alarms:InfluenceRadius1, ?radius) ^ alarms:hasTemporalInfluenceInHours(?y, ?z1) ^ sosa:resultTime(?y, ?t1) ^ sosa:Observation(?x) ^ sosa:Observation(?y) ^ differentFrom(?x, ?y) ^ **geo:hasInfluenceNow**(?bool, ?t, ?z) ^ swrlb:equal(?bool, true) ^ alarms:hasTemporalInfluenceInHours(?x, ?z) ^ sosa:resultTime(?x, ?t) ^ alarms:hasLocation(?x, ?l1) ^ alarms:hasLocation(?y, ?l2) ^ geosparql:asWKT(?l1, ?wkt1) ^ geosparql:asWKT(?l2, ?wkt2) ^ **geo:haversine**(?distance, ?wkt1, ?wkt2) ^ swrlb:lessThan(?distance, ?radius) ^ swrlb:greaterThan(?distance, 0) ^ alarms:triggersAlarm(?x, ?alarm1) ^ alarms:triggersAlarm(?y, ?alarm2) ^ alarms:hasPriority(?alarm1, ?prior1) ^ alarms:hasPriority(?alarm2, ?prior2) ^ alarms:NotCombinedAlarm(?alarm1) ^ alarms:NotCombinedAlarm(?alarm2) ^ swrlb:greaterThan(?prior1, 0) ^ swrlb:greaterThan(?prior2, 0) -> alarms:triggersAlarm(?x, alarms:CombinedAlarm1)
S4	sosa:Observation(?x) ^ **geo:hasInfluenceNow**(?bool, ?t, ?z) ^ swrlb:equal(?bool, false) ^ alarms:hasTemporalInfluenceInHours(?x, ?z) ^ sosa:resultTime(?x, ?t) -> alarms:hasActualInfluence(?x, false)

TABLE 11. SQWRL searches for the implemented system.

Search ID	SQWRL expression
Q1	sosa:Observation(?x) alarms:hasTemporalInfluenceInHours(?x, ?z) sosa:resultTime(?x, ?t) geo:hasInfluenceNow(?bool, ?t, ?z) swrlb:equal(?bool, true) alarms:hasLocation(?x, ?l1) geosparql:asWKT(?l1, ?wkt1) alarms:triggersAlarm(?x, ?alarm1) alarms:hasPriority(?alarm1, ?prior1) alarms:NotCombinedAlarm(?alarm1) swrlb:greaterThan(?prior1, 0) -> sqwrl:selectDistinct(?x, ?prior1, ?wkt1)
Q2	sosa:Observation(?x) alarms:hasTemporalInfluenceInHours(?x, ?z) sosa:resultTime(?x, ?t) geo:hasInfluenceNow(?bool, ?t, ?z) swrlb:equal(?bool, true) alarms:hasLocation(?x, ?l1) geosparql:asWKT(?l1, ?wkt1) alarms:triggersAlarm(?x, ?alarm1) alarms:CombinedAlarm(?alarm1) -> sqwrl:selectDistinct(?x, ?wkt1)

argument ?x) also triggers an individual alarm of the alarms:CombinedAlarm class. It is noted that the rule engine algorithm will include the same information in the other soda: Observation, since the rule will fire again when being symmetric for the arguments ?x and ?y.

- Rule S4 assigns a false value to the property data geo:hasInfluenceNow in case the temporary threshold set by alarms:hasTemporalInfluenceInHours has been exceeded. Note that does this property does not change from true to false at any time (it is set to false when the preset time is passed) since, as stated above, SWRL is not-monotonic. This rule can be used to directly locate in the ontology those sosa:Observations that can be removed.

Rules S3 and S4 require two custom built-ins, indicated in bold in Table 10 and with the custom geo: prefix. The authors have implemented said built-ins using the corresponding SWRLAPI module and the instructions indicated in [56]. This possibility of extending the expressive power of the SWRL is, according to the authors, a very important feature offered by SWRLAPI.

- geo:hasInfluence (?bool, ?t, ?z) returns a boolean ?bool that indicates if the resultTime ?t is at the current time in the temporary influence interval marked by ?z (in hours).
- geo:haversine (?distance, ?wkt1, ?wkt2) returns the distance ?distance between the locations expressed as WKT ?wkt1 and ?wkt2 calculated using the Haversine formula.

The third aspect to be detailed in this section is that related to the searches of entities in the ontology for its later visualization. The power of the SQWRL makes these searches compact and can use the SWRL built-ins used in the rules. In the case of the implemented system, two SQWRL expressions have been defined, shown in Table 11. The rules return the sosa:Observation that triggers alarms with influence in the current moment. In the case of Q1 are alarms:NotCombinedAlarms with alarms:priority greater than 0. In the case of Q2 are the alarms:CombinedAlarm.

IV. RESULTS

The authors have carried out several simulation tests of the system implemented in successive phases. First, we have generated sets of test observations with random values of results (values of alarms:hasNumericValue of individuals sosa:Result) and localization (values of geosparql: asWKT of individuals sf:Point, which in turn determine the location). This procedure allowed debugging some bugs both in the ontology and in the definition of the rules. As an example, the need to include the differentFrom(?x,?y) atom in rule S3 was verified, in order to distinguish two different sosa:Observation and to fix that axiom when new observations are included in the ontology. The reason is justified by the non-assumption of a unique name for individuals by the SWRL and by its open world assumption. After fixing those bugs, the system showed the expected behavior. Fig. 7 shows the visualization on a map of the markers

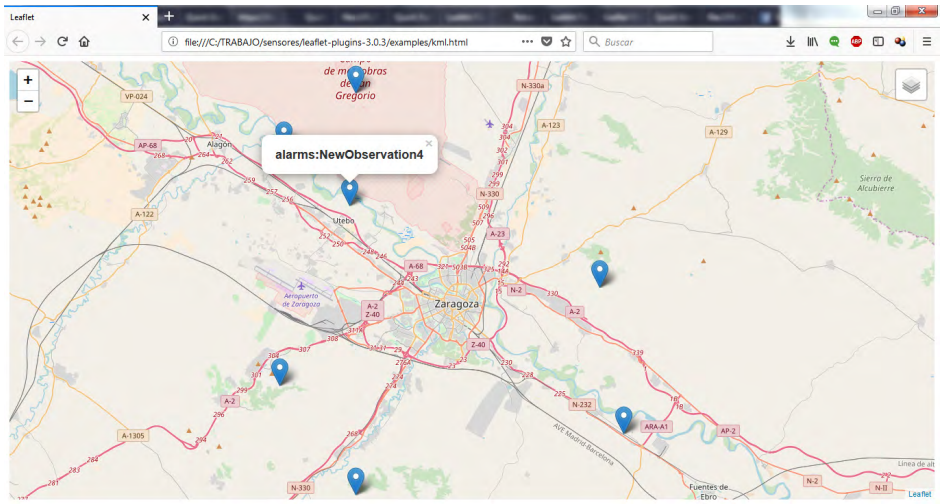

FIGURE 7. Monitoring of sosa:Observations locations in a web browser.

generated by sosa:Observation that triggers alarms:Alarm (alarms:CombinedAlarm and alarms: NotCombinedAlarm with priority greater than 0), in a simulated environment centered on the city of Zaragoza. As you can see, the implemented system takes advantage of the possibility of the pop-up to indicate the user additional information.

Simultaneously to this debugging on functionality and completeness, the Pellet reasoner has been used to detect possible inconsistencies of the ontology. The reasoner detected small inconsistencies derived from the use of OWL datatypes. Once these bugs are repaired, the reasoner indicates that the ontology is consistent.

The tests were carried out using an Intel®Core™i7-2600 CPU @ 3,40 GHz processor, 8GB RAM.

The following test of the use of the implemented system is with the use of real data to verify (also in simulation) its behavior. A series of real data accessible on the Internet has been used: water level data from the Ebro River on its way through Zaragoza (available on the website of the Ebro Hydrographic Confederation [70]), data on air quality in Madrid (available on the Madrid City Council website [71]) and on earthquakes in Spain (available on the page of the National Geographic Institute of Spain [72]). As an example of the use of the system, it has been able to reproduce warning and warning alerts referring to the rise in water level of the Ebro River produced in April 2018 (warning level: 4 m; emergency level: 4.5 m). For this purpose, a value of 30 hours was assigned for alarms: hasTemporalInfluenceInHours. Fig. 8 shows that evolution.

One of the analyses carried out with the ontology are the processing times of its use. First of all, an analysis was made of the time it takes for the rule engine to process a new set of observations in the ontology. The method has been as follows, random observations have been created (following the method explained above), and included in the ontology. After that, the time used by the rule engine is measured. The experiment is repeated three times to reduce the effect of

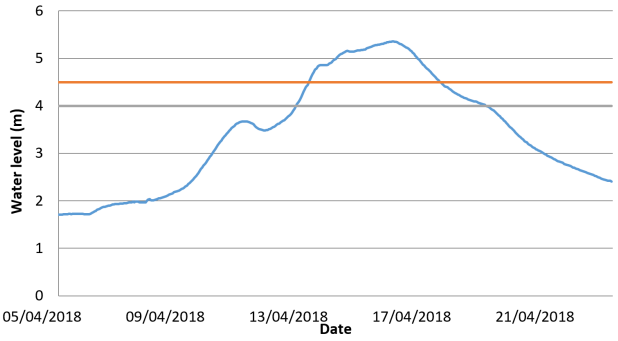

FIGURE 8. Evolution of water level in the Ebro River passing through Zaragoza in April 2018 ((warning level: 4 m; emergency level: 4.5 m). Data source: Ebro Hydrographic Confederation.

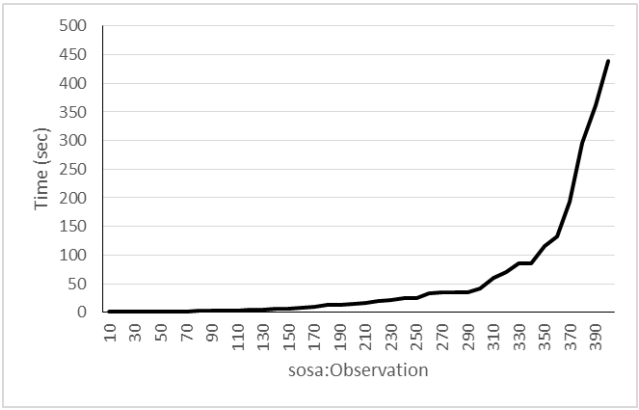

FIGURE 9. Evolution of time processing of a set of new sosa:Observations.

randomness and the mean of the times is calculated. Fig. 9 shows the evolution of time according to the number of sosa:Observation. It is verified that for a number of observations less than 110 the rule engine takes less than 3 seconds to trigger all the SWRL rules.

Later, the evolution of the processing time by the rule engine is analyzed when a new sosa:Observation is loaded according to the number of observations already existing in

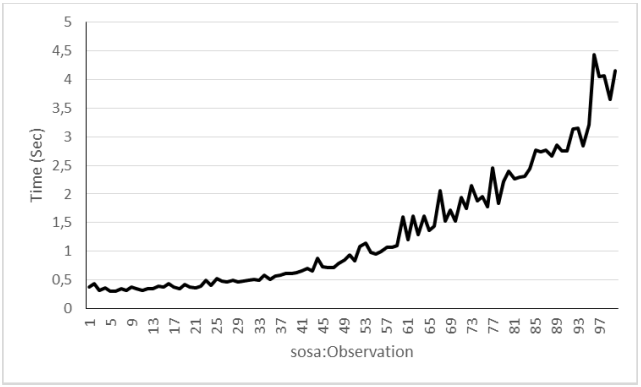

FIGURE 10. Evolution of time processing of a new sosa:Observation according to the number of observations already existing in the ontology.

the ontology. As in the previous case, the observation data are generated randomly and an average of three tests is done. The data obtained are shown in Fig. 10. It should be noted that the inclusion of a new sosa:Observation in the ontology involves adding 12 OWL axioms (or triples in the RDF terminology). In addition the definition of differentFrom axioms of the n-th sosa:Observation involves n-1 additional axioms. As it can be seen, the time is less than 1 second for 50 sosa:Observations in the ontology.

From the two experiments on the processing times, it can be concluded that the system presents a quite acceptable behavior for a not excessive number of sosa:Observations. It is noted that the observable magnitudes of the data with which it has been experimented can assume these processing times. However, when the number of observations stored in the ontology increases, it is advisable to eliminate those that have no temporal influence at the current time. In this sense, rule S4 can be useful.

V. AN EDUCATIONAL EXPERIENCE

In this section, the authors will briefly describe an educational experience using the development of the application presented in this paper. The authors believe that, although it is not the main focus of this work, this experience is significant to show the process of discovery of the ontological approach in the field of sensors and thereby achieve an improvement in the training of future engineers. With the proposed methodology, students are expected to acquire a deeper knowledge of both the technologies related to ontologies and the efficient management of sensor-based systems.

In recent years, practical teaching with sensor-based systems has been established as a cross-cutting element for the teaching of concepts related to several STEM disciplines, since it allows easy introduction into the learning process competencies such as design, innovation, skills and technical skills, problem solving and teamwork from a multidisciplinary perspective beyond the traditional acquisition of knowledge. In this sense, the introduction in the curriculum of an ontology-based approach supposes a plus of innovation and development with respect to other traditional approaches [37]. In this way, a constructivist process is defined that seeks to bridge the pedagogical gap that exists on many occasions between the students and the content that is taught.

A first (and incomplete) version of the resulting design has been applied to a blended postgraduate course ''Web Technologies'', which has a small number of students (10 students, 9 male and 1 female). Therefore, no significant validation has been obtained (which remains as an open line for future research), beyond the positive comments that have been made to the teaching staff of the subject. Even so, the authors consider that its mere description in this paper can constitute a point of interest. The profile of the students is mainly that of Computer Engineer. The subject has fifteen weeks of duration with four hours per week of teaching. In this environment, the field of sensors is an excellent base on which to focus the development of the subject.

The temporal design of the subject fits with the sequence of steps described in the previous sections. Of the 15 weeks of the course, the first eight weeks are used for the application. Table 12 indicates in detail the activities to be carried out during these weeks. The student will have available in the virtual classroom of the subject prior to the practice the material with which they will work: software tool, manuals, theory used. . ., in such a way that they have a prior knowledge of the subject, in accordance with the approach of Ausubel regarding the learning process [73]. For the course, a very didactic educational resource regarding the whole process

TABLE 12. Summary of the activities to be developed.

Week	Topics	Chapter(s) of [74] recommended	Proposed activities
1	Ontologies (Introduction)	1,4	Analysis of the material from virtual classroom/Presentation of applications
2	Protégé editor	5	Protégé editor
3	SSN, GeoSPARQL, Jena framework, OWL API	13-14	OWL basic activities (insert, remove, etc.)/ Puzzle learning
4, 5	Development of the ontology		Enrichment of presented ontologies/Work in couples
6	SWRL, SWRLJessBridge, SWRLAPI		Definition of students' own rules
7	SPARQL, SQWRL	6	Definition of students' own searches
8	Sgvizler/KML output		Integration of previous developed sets

is [74]. Table 12 also shows those recommended chapters for each topic.

Due to the limit of weeks of planning, many of the activities are descriptive of the process or the students are given a more or less advanced outline of the result, especially the employment of the SWRLJessBridge. In this case, access to the tools and libraries to be used is facilitated. It should be remembered that the objective sought is not for students to develop a complete application from scratch, but to present the tools of the ontological approach, that students work with them and apply them to the field of sensors. The reader might think that eight weeks is not enough time to present all these activities. However, students demonstrate a high work rate, acquired in their previous studies.

As it can be seen, with this weekly planning, the cognitive field is intended to be located in the student's proximal development zone. On several occasions, students already know the theoretical concepts of the topics to be developed, but they are totally lacking in the practice in their management. In this constructivist sense, the professors will act as a support of union between the learning objectives and the previous knowledge of the students.

During the first week, the concept of ontology is introduced and a theoretical repertoire of applications is presented, focusing on the field of sensors. Various bibliographical references are used, including those detailed in Table 2. The second week is used for students to become familiar with the Protégé editor. It is proposed to follow a complete tutorial [75], which helps them to consolidate the acquired concepts about ontologies. The following sessions are used to describe the two previously designed ontologies (SSN and GeoSPARQL) that will be used as the basis of the application to be developed. By having the Protégé editor, it is not necessary for students to access the OWL code directly. In this week, Jena and OWLAPI framework are also presented. Students implement very basic activities such as creating OWL models, inserting and removing statements, filtering elements and saving the model. For this case, a puzzle learning activity is carried out, where two groups are formed. Each group works with a framework and afterwards a sharing activity is carried out.

The development of ontologies (weeks 4-5) for beginners involves a significant amount of operations and sometimes complex edition of the ontology if concepts have not been previously assimilated. In these two weeks, students are asked to enrich the mentioned ontologies in order to be able to deal with the problem posed in class. To this end, the class is organized in work couples, who collaborate face to face in front of the screen to achieve a common goal. In this type of methodology, the students are forced to use their collaborative and social skills to build their knowledge: suggest ideas, design strategies for solving problems, offer positive feedback to the efforts made by the other partner, perseverance for complete the task, debate, reflection, argumentation and oral expression of new knowledge [76]. In the case that is detailed in this section, teachers will act as mediator in the possible conflicts between the members of the group within the constructivist process [77] and will promote a satisfactory shared understanding of the object of learning. The work of the groups is supervised and enriched by classical techniques of contingent teaching and metacognitive scaffolding [78]. In this experience, they have detected especially those failures referred to make all information explicit, mistaken use of universal restrictions rather than existential ones as the default and open world reasoning. This last case is, in the opinion of the authors, the one that supposes a greater effort to the students since they are accustomed to reason on traditional languages of programming.

SWRL is seen during week 6. Students will define their own rules about the defined ontology and see its effects using SWRLJessBridge and SWRLAPI. For this purpose, they are provided with some generic code. The next stage in the development (week 7) is about SPARQL and SQWRL search languages. In the case of SPARQL, a simple but complete tutorial [79] will be analyzed, which will make it easier to define students' own searches in the ontology. The knowledge acquired during that week is used in the following, integrating the set developed with the Sgvizler tool (for those students that worked previously with Jena) or Leaflet (for those students that worked with OWL API).

Finally, and in parallel with the delivery of the second part of the subject, a series of seminars are held in the subject with the aim of strengthening and complementing the skills acquired by the students. During these weeks the teachers will continue to apply the scaffolding techniques so that the students improve their skills and take control of their learning.

VI. CONCLUSIONS

This paper proposes an ontology for an alarm system in geographic sensor systems. Ontology imports the Semantic Sensor Network and GeoSPARQL ontologies. The authors establish a semantic approach to integrate the information of systems that require alarms. This method is fully extensible for the specific domain knowledge, which will enrich future applications without recompiling any code, just adding axioms to the ontology. The benefits of the system are acceptable for systems of this type, although a management of old observations can even improve it. This paper includes a detailed analysis of the software tools used and the Semantic Web, with which the reader can have a global vision of the field. The authors have also included this system within a pedagogical approach for the teaching of the Semantic Web to postgraduate students.

Thus, the most interesting contributions of the work are: a survey of the available literature in the field of the use of Semantic Web technologies and ontologies for the detection of events from data obtained from sensors; a study on the tools and vocabularies to be used to create a system that interfaces with ontologies; an educational method - reporting the authors' experience - to help university students understand this topic, with a rule-based system for the identification of events.

There are several future lines of development in this research. The first is to define an efficient system for eliminating information that is no longer relevant in the system. The second is to use other geographical configurations provided by the GeoSPARQL ontology and which the KML can also represent, as is the case of polygons. Once it is done, it can be tested with guarantees in real-world systems.

REFERENCES

[1] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[2] N. Guarino, "Formal ontology, conceptual analysis and knowledge representation," *Int. J. Hum.-Comput. Stud.*, vol. 43, nos. 5–6, pp. 625–640, 1995.

[3] B. Swartout, R. Patil, K. Knight, and T. Russ, "Toward distributed use of large-scale ontologies," in *Proc. Ontol. Eng. Symp. AAAI Spring*, 1997, pp. 138–148.

[4] I. G. Rodríguez, "Modelos de conocimiento basados en ontologías para la construcción de software en el dominio de la ingeniería de control," Ph.D. dissertation, Dept. Ingeniería Eléctrica Sistemas Automática, Univ. León, León, Spain, 2012.

[5] E. Blomqvist and A. Öhgren, "Constructing an enterprise ontology for an automotive supplier," *Eng. Appl. Artif. Intell.*, vol. 21, no. 3, pp. 386–397, 2008.

[6] V. Kashyap, C. Bussler, and M. Moran, *The Semantic Web: Semantics for Data and Services on the Web*, 1st ed. Berlin, Germany: Springer-Verlag, 2008.

[7] M. Uschold and M. Gruninger, "Ontologies: Principles, methods and applications," *Knowl. Eng. Rev.*, vol. 11, no. 2, pp. 93–136, 1996.

[8] H. J. Happel and S. Seedorf, "Applications of ontologies in software engineering," presented at the 2nd Int. Workshop Semantic Web Enabled Softw. Eng. (SWESE), Athens, GA, USA, Nov. 2006.

[9] A. G. Salguero, M. Espinilla, P. Delatorre, and J. Medina, "Using ontologies for the online recognition of activities of daily living," *Sensors*, vol. 18, no. 4, p. 1202, 2018.

[10] F. Xu, X. Liu, W. Chen, C. Zhou, and B. Cao, "Ontology-based method for fault diagnosis of loaders," *Sensors*, vol. 18, no. 3, p. 729, 2018.

[11] W. Jin and D. H. Kim, "Design and implementation of e-health system based on semantic sensor network using IETF YANG," *Sensors*, vol. 18, no. 2, p. 629, 2018.

[12] M. Alirezaie *et al.*, "An ontology-based context-aware system for smart homes: E-carehome," *Sensors*, vol. 17, no. 7, p. 1586, 2017.

[13] E. Maleki, F. Belkadi, M. Ritou, and A. Bernard, "A tailored ontology supporting sensor implementation for the maintenance of industrial machines," *Sensors*, vol. 17, no. 9, p. 2063, 2017.

[14] X. Li, S. Bilbao, T. Martín-Wanton, J. Bastos, and J. Rodriguez, "SWARMs ontology: A common information model for the cooperation of underwater robots," *Sensors*, vol. 17, no. 3, p. 569, 2017.

[15] C. Villalonga *et al.*, "Ontology-based high-level context inference for human behavior identification," *Sensors*, vol. 16, no. 10, p. 1617, 2016.

[16] S. Fernandez, R. Hadfi, T. Ito, I. Marsa-Maestre, and J. R. Velasco, "Ontology-based architecture for intelligent transportation systems using a traffic sensor network," *Sensors*, vol. 16, no. 8, p. 1287, 2016.

[17] X. Huang, J. Yi, X. Zhu, and S. Chen, "A semantic approach with decision support for safety service in smart home management," *Sensors*, vol. 16, no. 8, p. 1224, 2016.

[18] C. F. Crispim-Junior *et al.*, "Online recognition of daily activities by color-depth sensing and knowledge models," *Sensors*, vol. 17, no. 7, p. 1528, 2017.

[19] A. Alti, A. Lakehal, S. Laborie, and P. Roose, "Autonomic semantic-based context-aware platform for mobile applications in pervasive environments," *Future Internet*, vol. 8, no. 4, p. 48, 2016.

[20] F. Xu, X. Liu, and C. Zhou, "Developing an ontology-based rollover monitoring and decision support system for engineering vehicles," *Information*, vol. 9, no. 5, p. 112, 2018.

[21] C. Wang, N. Chen, W. Wang, and Z. Chen, "A hydrological sensor Web ontology based on the SSN ontology: A case study for a flood," *ISPRS Int. J. Geo-Inf.*, vol. 7, no. 1, p. 2, 2018.

[22] H. Knublauch *et al.*, "The protégé OWL experience," in *Proc. OWL Exper. Directions Workshop*, 2005, pp. 1–11.

[23] *Kaon2 Web Page*. Accessed: Feb. 5, 2018. [Online]. Available: http://kaon2.semanticweb.org

[24] N. Khozouie, F. Fotouhi-Ghazvini, and B. Minaei-Bidgoli, "Ontological MobiHealth system," *Indonesian J. Elect. Eng. Comput. Sci.*, vol. 10, no. 1, pp. 309–319, 2018.

[25] A. Wróblewska, P. Kapłański, P. Zarzycki, and I. Ługowska, "Semantic rules representation in controlled natural language in FluentEditor," in *Proc. 6th Int. Conf. Hum. Syst. Interact. (HSI)*, Jun. 2013, pp. 90–96.

[26] J. Bārzdiņš, G. Bārzdiņš, K. Cerans, R. Liepiņš, and A. Sprogis, "OWLGrEd: A UML style graphical editor for OWL," in *Proc. CEUR Workshop*, 2010, pp. 23–28.

[27] M. Weiten, "Ontostudio as a ontology engineering environment," in *Semantic Knowledge Management Integrating Ontology Management, Knowledge Discovery, and Human Language Technologies*, J. F. Davies, M. Grobelnik, and D. Mladenic, Eds. Berlin, Germany: Springer-Verlag, 2009, pp. 51–60.

[28] *TopBraid Composer*. Accessed: May 4, 2018. [Online]. Available: https://www.topquadrant.com/tools/modeling-topbraid-composer-standard-edition/

[29] A. Kalyanpur, B. Caruso, N. Cappadona, M. Worthington, S. Mitchell, and J. Corson-Rikert, "Swoop: A Web ontology editing browser," *Web Semantics, Sci., Services Agents World Wide Web*, vol. 4, no. 2, pp. 144–153, 2006.

[30] B. Lowe *et al.*, "The vitro integrated ontology editor and semantic Web application," in *Proc. CEUR Workshop*, 2011, pp. 296–297.

[31] *Protégé Wiki*. Accessed: May 4, 2018. [Online]. Available: https://protegewiki.stanford.edu/wiki/Protege4Features

[32] B. Glimm *et al.*, "HermiT: An OWL 2 reasoner," *J. Autom. Reason.*, vol. 53, no. 3, pp. 245–269, 2014.

[33] D. Tsarkov and I. Horrocks, "FaCT++ description logic reasoner: System description," in *Automated Reasoning* (Lecture Notes in Computer Science), vol. 4130. Berlin, Germany: Springer-Verlag, 2006, pp. 292–297.

[34] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical OWL-DL reasoner," *Web Semantics, Sci., Services Agents World Wide Web*, vol. 5, no. 2, pp. 51–53, 2007.

[35] V. Haarslev and R. Möller, "RACER system description," in *Automated Reasoning* (Lecture Notes in Computer Science), vol. 2083. Berlin, Germany: Springer-Verlag, 2001, pp. 701–705.

[36] *OWL Web Ontology Language Guide*. Accessed: Jul. 5, 2018. [Online]. Available: http://www.w3.org/TR/owl-guide/

[37] A. Rector *et al.*, "OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns," in *Engineering Knowledge in the Age of the Semantic Web* (Lecture Notes in Computer Science). Berlin, Germany: Springer-Verlag, 2004, pp. 63–81.

[38] D. McGuinness and F. van Harmelen, Eds., *OWL Web Ontology Language Overview*. Accessed: May 18, 2018. [Online]. Available: https://www.w3.org/TR/owl-features/

[39] E. J. G. González, "Diseño e implementación de una arquitectura multipropósito basada en agentes inteligentes: Aplicación a la planificación automática de agendas y al control de procesos," Ph.D. dissertation, Dept. Fís. Fund. Exper. Electr. Sistemas, Univ. La Laguna, San Cristóbal de La Laguna, Spain, 2004.

[40] *OWL Web Ontology Language Reference*. Accessed: May 4, 2018. [Online]. Available: https://www.w3.org/TR/owl-ref/

[41] J. Lee *et al.*, "Processing SPARQL queries with regular expressions in RDF databases," *BMC Bioinf.*, vol. 12, no. 2, p. S6, 2011.

[42] M. O'Connor, C. Nyulas, R. Shankar, A. Das, and M. Musen, "Connor, "The SWRLAPI: A development environment for working with SWRL rules," in *Proc. 4th Int. OWL Workshop Exper. Directions (OWLED)*, Washington, DC, USA, 2008, pp. 1–4.

[43] M. O'Connor, S. Tu, C. Nyulas, A. Das, and M. Musen, "Querying the semantic Web with SWRL," in *Advances in Rule Interchange and Applications* (Lecture Notes in Computer Science), vol. 4824. Berlin, Germany: Springer-Verlag, 2007, pp. 155–159.

[44] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A semantic Web rule language combining OWL and RuleML," Tech. Rep. W3C, Feb. 2004.

[45] G. Wagner, G. Antoniou, S. Tabet, and H. Boley, "The abstract syntax of RuleML—Towards a general Web rule language framework," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI)*, Sep. 2004, pp. 628–631.

[46] S. Bhansali and B. N. Grosof, "Extending the SweetDeal approach for e-procurement using SweetRules and RuleML," in *Rules and Rule Markup Languages for the Semantic Web* (Lecture Notes in Computer Science), vol. 3791. Berlin, Germany: Springer-Verlag, 2005, pp. 113–129.

[47] M. K. Sarker, D. Carral, A. A. Krisnadhi, and P. Hitzler, "Modeling OWL with rules: The ROWL protégé plugin," in *Proc. CEUR Workshop*, 2016, pp. 1–4.
[48] E. Wang and Y. S. Kim, "A teaching strategies engine using translation from SWRL to Jess," in *Intelligent Tutoring Systems* (Lecture Notes in Computer Science), vol. 4053. Berlin, Germany: Springer-Verlag, 2006, pp. 51–60.
[49] DAML. *Examples SWRL*. Accessed: May 16, 2018. [Online]. Available: http://www.daml.org/2003/11/swrl/examples.html
[50] J. Bak, C. Jedrzejek, and M. Falkowski, "Usage of the Jess engine, rules and ontology to query a relational database," in *Rule Interchange and Applications* (Lecture Notes in Computer Science), vol. 5858. Berlin, Germany: Springer-Verlag, 2009, pp. 216–230.
[51] J. Mei, E. P. Bontas, and Z. Lin, "OWL2Jess: A transformational implementation of the OWL semantics," in *Parallel and Distributed Processing and Applications* (Lecture Notes in Computer Science), vol. 3759. Berlin, Germany: Springer-Verlag, 2005, pp. 599–608.
[52] *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. Accessed: May 17, 2018. [Online]. Available: https://www.w3.org/Submission/SWRL/
[53] B. McBride, "Jena: Implementing the RDF model and syntax specification," in *Proc. 2nd Int. Workshop Semantic Web (SemWeb)*, 2001, pp. 23–28.
[54] *Jena Web Site*. Accessed: May 4, 2018. [Online]. Available: https://jena.apache.org/
[55] V. Tsetsos, V. Papataxiarhis, and S. Hadjiefthymiades, "Personalizing pedestrian location services through ontologies and rules," in *Personalization of Interactive Multimedia Services: A Research and Development Perspective*, J. J. Pazos-Arias, C. D. Kloos, and M. L. Nores, Eds. Commack, NY, USA: Nova, 2009, pp. 333–349.
[56] *SWRL API Project*. Accessed: Jun. 16, 2018. [Online]. Available: https://github.com/protegeproject/swrlapi-project
[57] M. G. Skjæveland, "Sgvizler: A JavaScript wrapper for easy visualization of SPARQL result sets," in *The Semantic Web: ESWC 2012 Satellite Events* (Lecture Notes in Computer Science), vol. 7540. Berlin, Germany: Springer-Verlag, 2015, pp. 361–365.
[58] *Leaflet Web Site*. Accessed: Jun. 16, 2018. [Online]. Available: https://leafletjs.com/
[59] *KML.js Web Site*. Accessed: Jun. 16, 2018. [Online]. Available: https://github.com/shramov/leaflet-plugins/blob/master/examples/kml.html
[60] C. Schlenoff, T. Hong, C. Liu, R. D. Eastman, and S. Foufou, "A literature review of sensor ontologies for manufacturing applications," in *Proc. IEEE Int. Symp. Robot. Sensors Environ. (ROSE)*, Oct. 2013, pp. 96–101.
[61] R. D. Eastman, C. I. Schlenoff, S. B. Balakirsky, and T. H. Hong, "A sensor ontology literature review," U.S. Dept. Commerce, Washington, DC, USA, Tech. Rep. NISTIR-7908, 2013.
[62] M. Fernández, C. Overbeeke, M. Sabou, and E. Motta, "What makes a good ontology? A case-study in fine-grained knowledge reuse," in *Proc. 4th Asian Conf. Semantic Web*, 2009, pp. 61–75.
[63] *Semantic Sensor Network Ontology*. Accessed: May 4, 2018. [Online]. Available: https://www.w3.org/TR/vocab-ssn/
[64] A. Agresta *et al.*, "An ontology framework for flooding forecasting," in *Computational Science and Its Applications—ICCSA* (Lecture Notes in Computer Science), vol. 8582. Berlin, Germany: Springer-Verlag, 2014, pp. 417–428.
[65] *GeoSPARQL—A Geographic Query Language for RDF Data*. Accessed: May 16, 2018. [Online]. Available: http://www.opengeospatial.org/standards/geosparql
[66] *W3C Geospatial Vocabulary*. Accessed: May 8, 2018. [Online]. Available: https://www.w3.org/2005/Incubator/geo/XGR-geo-20071023/
[67] *WGS84 Geo Positioning: An RDF Vocabulary*. Accessed: May 18, 2018. [Online]. Available: https://www.w3.org/2003/01/geo/wgs84_pos
[68] *Protégé Project Forum*. Accessed: Jun. 16, 2018. [Online]. Available: http://protege-project.136.n4.nabble.com/
[69] *QUDT Overview*. Accessed: Jun. 16, 2018. [Online]. Available: http://www.qudt.org/pages/QUDToverviewPage.html
[70] *Hydrographic Confederation of the Ebro*. Accessed: Jun. 16, 2018. [Online]. Available: http://www.chebro.es/
[71] *Madrid Council Web Site*. Accessed: Jun. 16, 2018. [Online]. Available: http://www.mambiente.munimadrid.es/sica/scripts/index.php
[72] *Instituto Geográfico Nacional de España Web Site*. Accessed: Jun. 16, 2018. [Online]. Available: http://www.ign.es/web/ign/portal/sis-area-sismicidad
[73] D. P. Ausubel, *Educational Psychology: A Cognitive View*. New York, NY, USA: Holt, 1968.
[74] L. Yu, *A Developer's Guide to the Semantic Web*. Berlin, Germany: Springer-Verlag, 2011.
[75] M. Horridge, S. Jupp, G. Moulton, A. Rector, R. Stevens, and C. Wroe, "A practical guide to building owl ontologies using Protégé 4.0 and co-ode tools edition 1.3," Univ. Manchester, Manchester, U.K., Tech. Rep., 2011. [Online]. Available: http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf
[76] C. Angeli and A. Tsaggari, "Examining the effects of learning in dyads with computer-based multimedia on third-grade students' performance in history," *Comput. Educ.*, vols. 92–93, pp. 171–180, Jan. 2016.
[77] K. Lovelace, D. L. Shapiro, and L. R. Weingart, "Maximizing cross-functional new product teams' innovativeness and constraint adherence: A conflict communications perspective," *Acad. Manage. J.*, vol. 44, no. 4, pp. 779–793, 2001.
[78] R. Reinglod, R. Rimor, and A. Kalay, "Instructor's scaffolding in support of student's metacognition through a teacher education online course: A case study," *J. Interact. Online Learn.*, vol. 7, no. 2, pp. 139–151, 2008.
[79] L. Feigenbaum. *SPARQL by Examples Tutorial*. Accessed: May 18, 2018. [Online]. Available: https://www.cambridgesemantics.com/blog/semantic-university/learn-sparql/sparql-by-example/

EVELIO GONZÁLEZ received the M.S. degree in applied physics and the Ph.D. degree from the University of La Laguna, Tenerife, Spain, in 1998 and 2004, respectively. He is a Professor with the Universidad de La Laguna. His areas of interest include ontologies, artificial intelligence, and intelligent agents.

ROBERTO MARICHAL received the M.S. degree in applied physics and the Ph.D. degree in computer science from the University of La Laguna, Canary Islands, Spain, in 1996 and 2003, respectively. From 1996 to 2000, he was a Research Student with the Department of Applied Physics, Electronics and Systems, University of La Laguna. He was a Visitor with the University of Technical University of Denmark, Denmark, in 2000, and the University of Sevilla, Spain, in 2004. He is currently a Professor with the Department of Computer and Systems Sciences, University of La Laguna. His current research interests include process neural network, nonlinear control, and bifurcation theory. He received the National Award for Academic Excellence. He currently serves as a reviewer in different journals.

ALBERTO HAMILTON received the degree in physics from the Complutense University, Madrid, Spain in 1991, and the Ph.D. degree in computer engineering from the University of La Laguna, Canary Islands, Spain, in 1995. He has been a Lecturer with the Department of Computer Engineering and Systems, University of La Laguna, since 1997. He is author of around 40 international papers and he has nearly 50 communications in international conferences. He has participated in eight national financed competitive projects. His research interests include artificial agent systems, mobile robotics, and Internet of Things.