The Role of Ontologies in Intelligent Information Systems

Eduardo Mena

Dept. of Computer Science & System Engineering
University of Zaragoza
50018 Zaragoza, Spain
Email: emena@unizar.es

Abstract: *The development of Information Systems has arisen as one of the most important areas in computing systems. Since their very beginning in the middle of the twentieth century, computers have been mainly used to store, manage, and retrieve data and they have become priceless tools for this task. However, when the amount of data managed by information systems becomes huge, the use of semantic techniques to ease the access of people to big amounts of data is mandatory.*
In this paper we review, along the last 30 years the role of ontologies and the reasoners that manage them, as needed techniques for developing intelligent information systems. Some of the advantages that we can obtain when using them nowadays, both in static and mobile information systems, are presented and illustrated with existing systems.

Keywords: *Semantic Web, Mobile Computing* ***Conference topic:*** *Information Technology*

1. Introduction

The development of Information Systems has arisen as one of the most important areas in computing systems. Since their very beginning, in the middle of the twentieth century, computers has been mainly used to store, manage, and retrieve data, and they have become priceless tools for this task, and that is the reason for the big and successful industry of database systems .

However, after some decades of increasing storage of data, in the middle 90's we realized that the huge and exponentially growing amount of data being stored and managed by databases would be useless for people unless we could obtain some help from the system: the problem is not storing or searching data but making explicit what we store, what we want to search, and what is the meaning of the retrieved answer. This was the beginning of a new era in Computing Science: We do not want to store data only but *information*, i.e., both data and their meanings. In other words, we needed *Information Systems* rather than database systems (which in fact will store all the data within an Information System; since then a database became a tool to manage data rather than the final user application used to retrieve the wanted information).

But then a problem arose: We had very good software to manage data (DataBase Management Systems)… but there was a lack of software to manage information. We were experts in managing data but not their meanings; how to manage their underlying semantics? Fortunately, a branch of Artificial Intelligence community has been working for years in how to express, store, and manage meanings, *knowledge*. We are talking of the research area of Knowledge Representation, which by using different formalisms (frames, semantic networks, knowledge bases, reasoners, etc.) had the most sophisticated existing tools (at least, much more advanced than those used by the database community) to describe the meaning of data. It was kind of ironic: Database people had been worried only about how to store, manage, and retrieve huge amounts of data very quickly, but they did not care about the meaning of those data, and the Knowledge Representation people had been worried only about how to represent knowledge, being not interested in how to manage and reason with a huge amount of knowledge. Thus, by middle 90's, both research communities found out in each other what they both needed: Finally, knowledge representation met databases, and vice versa. Thus, for the first time, we had the main bricks needed to build and manage huge and user-centered information systems.

Knowledge bases, lately renamed as ontologies when used to describe data sources, and the knowledge management systems to deal with them, have been helping us to provide semantics management to information systems since then. In a world like ours, where the access to the wanted information is so crucial, ontologies allow software designers to relieve users from having to deal with huge amounts of data. Instead, ontologies allow us to see (huge) information systems from the point of view we want to see them.

In this paper we review, along the last 30 years, the role of ontologies, and the reasoners to manage them, as unvaluable techniques to develop intelligent information systems. The different applications of ontologies

when accessing to data will be presented using research systems that we have been developing during the last years. So we will see some of the advantages that we can obtain when using ontologies nowadays, both in static and mobile information systems.

In Section 2, we will talk about the past, showing the first uses of ontologies as semantic descriptions of distributed database systems and how they became the key of global information systems when the Web was born, and its main role in the so called Semantic Web. In Section 3, we review the use of ontologies in some current static or mobile information systems. Finally, conclusions and open issues are presented in Section 4.

2. Ontology-based Systems in the Past

In this section we summarize the role of ontologies in the first kinds of information systems that took care not only of managing data but also describing what those data represent.

2.1. Database Technology meets Knowledge Representation in the 80's

During the 80's, the database technology was applied to the arising computer networks that were gaining importance in the design of information systems. The result was a new generation of *distributed* database systems, where a (global) database was supported by different (local) databases residing on different computers connected by a (local or wide area) network. Although these new architectures implied new problems that had to be faced, the flexibility and scalability of such systems showed the way to follow in the future. Three main distributed database architectures were defined:

1. Distributed databases, which follow a top-down approach: a global database is designed and then its schema is split into different local schemas, each one becoming a local database. Thus the data and the (global) database itself are distributed into several nodes. A query to the global schema is split into queries to the different local databases (whose data intersection can be empty or not), and finally the partial local answers are correlated into an answer to the global query.
2. Interoperable databases, where different preexisting databases are linked through a network using some kind of distributed SQL. Each of them can continue working independently although they take part of a global database. The problem is that a global schema does not exist so users of that system must be aware of distribution of data (they must know where each piece of data resides).
3. Federated Databases, which follow a bottom-up approach: the schemas of different preexisting databases are translated into a canonical model and then integrated into a global schema that can be used by federated users to query the whole information space; at the same time, local databases are independent enough to be accessed directly by their local users.

Federated databases are the most flexible and scalable, as their architecture of five levels (see Figure 1) hides (syntactic and semantic) heterogeneity from users. We can observe that the existence of a global schema is the key of such a federated architecture: the global schema describes what kind of information can be found in the federated system *before* accessing the data. It plays the role of a semantic catalog: if you are looking for some information first you consult the global schema and, only if you found something interesting, then you access to the underlying data. The advantage is that is much better to consult tens or hundreds of terms in the global schema than thousands or millions of tuples in the underlying databases.

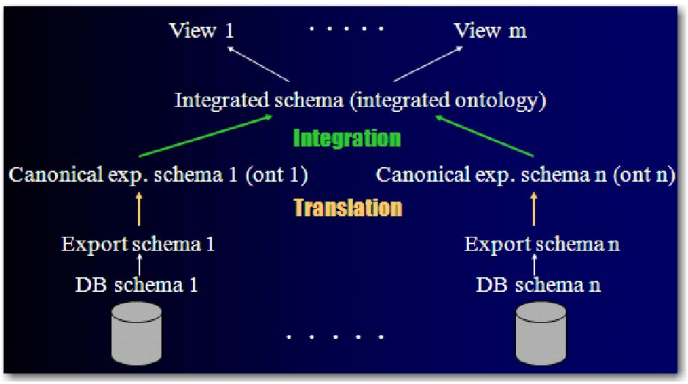

Figure 1: Five-level architecture of federated database systems

As explained in the introduction, while the database community was developing these visionary and flexible database architectures that distinguish between the data and their descriptions, the knowledge representation community (by then an area completely isolated from the area of databases) had developed different theoretical and practical tools to describe and manage certain kind of knowledge. Knowledge bases were created using knowledge representation languages where expressivity (what you can state) was more important than efficiency. *Ontologies*, the name given to knowledge bases when used in information systems[1], were defined as *"an explicit specification of a conceptualization"* [12]. Ontologies allow to model and capture the semantics of different knowledge domains, providing a means to share definitions and reach an implicit agreement on the meaning of the published information. Ontologies represent the vocabulary of some domain from a certain point of view and, when described using some formal language, can be managed easily by computers.

Although different formalisms exist to describe ontologies, we would like to stress Description Logics (DLs) [1], a family of logics for representing structured knowledge. They are a well-known formalism providing a good trade-off between expressivity of the representation and efficiency of the reasoning (to deduce implicit knowledge, i.e., logical consequences of the knowledge in an ontology). The most typical reasoning services in ontologies are designed to check: 1) *Consistency*: An ontology O is consistent iff it has a model, i.e., there is an interpretation satisfying every axiom in O; 2) *Entailment*: O entails an axiom τ iff every model of O satisfies τ; 3) *Concept satisfiability*: A concept C is satisfiable w.r.t. O iff it is not interpreted as the empty set in some model of O; 4) *Concept subsumption*: A concept D subsumes a concept C w.r.t. O iff C is interpreted as a subset of D in every model of O; and 5) *Classification*: The classification of an ontology O consists of computing a hierarchy of concepts based on their subsumption relation.

Thus, in DL-based ontologies, the basic ontological representation primitives (also called ontology elements) are individuals (or instances, which are the objects of the world), concepts (or classes, which are sets of individuals), properties (also called roles or relations, which define interactions between pairs of individuals of the domain), datatypes (or concrete domains such as numbers, strings, etc.), and axioms (formal conditions to be verified by the elements). Different DL reasoners, the software that implement all these theoretical models, were implemented and made available to be used by that time.

2.2. Global Information Systems in the 90's

At the beginning of the new decade, a new generation of systems arose. They aimed to manage a high number of data sources as it was clear that distributed systems would grow and grow in the future. In Figure 2 we show a classification of the main approaches that were proposed by different research groups and organizations; specific systems appear in dark blue within its category. We can observe that some of them were based on keyword search while others defined specific software agents to search the data, some of them relying on a global ontology that describes all the information system (which did not scale up well for many underlying data sources), and the most complex ones advocated accessing different information systems, each one described by one or more ontologies; in that case, interontology relationships were defined to semantically integrate different ontologies whenever needed. Now the main effort was in the management of data semantics rather than in accessing to the data, which has been the main goal of information systems until then.

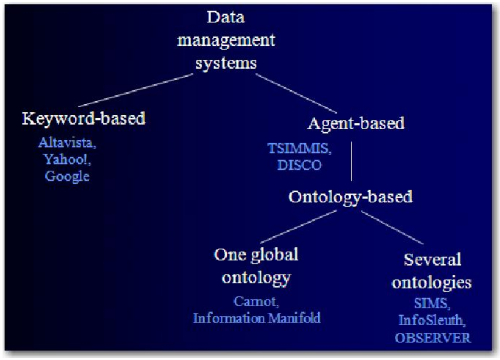

Figure 2: Approaches for global information systems in the 90's

[1] Ontologies, derived from the Greek word "ontos" (the essence of being), describe what kind of data *exist* in underlying data sources. They explicitly say what can and cannot be found in underlying data sources before accessing to them.

An important point to stress is that some of these systems, in addition to foresee the need of managing really huge information spaces, were flexible enough to allow the access to any kind of underlying data source, i.e., underlying data could be stored in any data organization (file systems, databases, even remote data sources of unknown data organization). The main goal was to find the needed information with the help of ontologies in the role of semantic catalogs of underlying data. Some of these systems, like OBSERVER [14], even provided imprecise answers whenever precise answers could not be obtained, estimating the *loss of information* by comparing the semantics of the user query and the semantics of the answer retrieved. Hence, all these approaches advocated using DL-based ontologies to take advantage of their DL reasoners in order to easily obtain inferences and semantic checking on ontologies, which would have been very costly to be developed from scratch.

However, in 1995 the World Wide Web entered the game. From the point of view of information management systems, the Web came to show that integrating two, five, or 20 databases, which has been the goal of distributed databases in the 80's, was not going to be enough: on the Web hundreds, thousands, (and later millions) of data sources were added continuously, and users did (and do) want to see all of them as a whole, as a huge but unique information system that they can query. Fortunately most of the integration techniques used by the different proposals of global information systems from the 90's, already shown in Figure 2, were still valid to access to websites… but the highly dynamic nature of the Web made their static integration techniques (with human intervention) obsolete. We needed new dynamic on-the-fly ontology integration techniques.

2.3. The Beginning of a New Century: The Semantic Web

Just a few years after the Web became very popular and huge in size, its own creator defined what he called the *Semantic Web* [3]. Its main goals, which pursue to fix some of the already well known problems of the Web, are: 1) to manage data and their descriptions, separately, 2) to be useful not only to humans (like the Web) but also to computer software (some services need to access to the Web), and 3) to allow semantic searches (as opposite to syntactic searches on the Web), where users specify what they want and the system retrieves exactly that. From the Semantic Web point of view, ontologies play a key role to achieve those goals: semantics of data must be expressed by ontologies that can be consulted, integrated, etc. by software systems to perform intelligent information retrieval. OWL became the language that is the current W3C standard for representing ontologies in the Semantic Web, which has Description Logics as the underlying formalism. Definitely, the ontology-based approaches from the 90's had hit the target when advocated the use of these semantic technologies.

3. Ontology-Based Systems in Current Information Systems

Today, although some of the goals of the Semantic Web have not been achieved yet, many advances have taken place. Knowledge representation languages to describe ontologies have evolved to the current standard Web Ontology Language (OWL 2) [13]. Currently many software applications (reasoners) implement ontology reasoning services, such as JFact[2], HermiT [10], or Pellet [15]. RDF and SPARQL are an alternative to those information system developers who do not need a high expressive power to describe their data sources but need a very efficient and quick ontology-based information retrieval system. Linked Data [2] is another important initiative, very popular nowadays, to store or publish data in RDF format so that data are linked to ontologies in a way easily processed by software. Thus, currently we have a good bunch of semanrtic tools and background knowledge to perform the dynamic ontology integration and data search that we needed since the Web shook the world of information management.

In this section, we summarize some of the different semantic-based applications and projects that we have developed in the Distributed Information Systems (SID) research group[3] during the last ten years. All of them benefit from semantic technologies, and provide a good example of how the addition of semantics broadens the capabilities of an information system. We summarize here each system (a more detailed description can be found in [6]):

- We have applied the use of semantic techniques to the field of search. Using disambiguation techniques which exploit the knowledge stored in ontologies [11], we have developed two different approaches to keyword search over different information systems: *QueryGen* [4], and *Doctopush* [17]. The former one is oriented to perform semantic keyword-based search over heterogeneous information systems, proposing a generalized semantic keyword interpretation process, while the latter aims at performing semantic data retrieval over the Web using the semantics of keywords to cluster relevant sources of information.

[2] http://jfact.sourceforge.net
[3] http://sid.cps.unizar.es

- We have also devised a framework to enhance semantically different tasks regarding information extraction, such as automatic text classification, semantic search, and summarization of text sources, among others. This framework, called *GENIE*, is aimed at supplying a set of libraries designed to assist developers in projects of this nature, adopting a semantic approach to all modules, thus taking advantage of the latest advances made in ontological engineering and semantics.
- Regarding the standard formalism used for ontologies, we have studied an extension of the semantics of Description Logics to embrace Fuzzy Logic. This has led to the implementation of the fuzzy ontology editor *Fuzzy OWL 2* [8], and the fuzzy DL reasoners *DeLorean* and *fuzzyDL,* combining the expressivity of classical DL languages (which use crisp logic) with the flexibility of fuzzy logics for imprecise knowledge representation.
- Finally, we have studied the relationship between locations and semantics. On the one hand, we have applied the know-how acquired using ontological formalisms to study how we can model locations using different granularities while keeping and exploiting their semantics. On the other hand, we are currently working on enhancing the use of Location-Based Services (LBSs) adding semantics, which is crystallizing in *SHERLOCK* [18], a system that searches and shares up-to-date knowledge from nearby devices to provide users with interesting LBSs.

In fact, all of these systems are not isolated, but help each other to perform different tasks. In Figure 3, we can see how they could be coordinated. SHERLOCK uses our different semantic models of locations to give meaning to the locations of the users, and infer further information out from them. Also, SHERLOCK might provide a keyword interface and could use QueryGen to construct a formal specification of the requested service out from the user's input keywords. Doctopush could be integrated as well and used as a particular service under the umbrella of SHERLOCK.

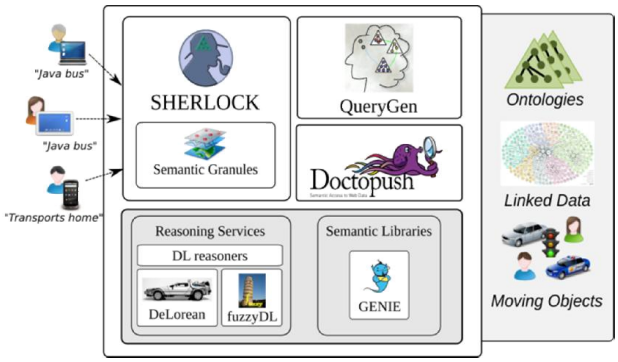

Figure 3: Overview of our semantic-based systems

All of these systems use several reasoning and semantic services, which could be abstracted in a separated layer. This layer would expose the services provided by classical DL reasoners, as well as the reasoning services of our fuzzy extension for DLs represented in the figure by DeLorean and fuzzyDL reasoners.

In the rest of this section we provide more details of two of our ontology-based information systems: QueryGen and SHERLOCK.

3.1. Semantic Keyword Seach: QueryGen

In this section we provide a more detailed summary of QueryGen, our approach to semantically deal with a keyword-based search. The use of keyword queries, although it is very easy for anyone, introduces a semantic gap between the user intention and the queries as, in fact, keyword queries are simplifications of the queries that really express the user's information need. For that reason, for instance, when talking about Web searches, users usually have to browse the returned web pages looking for the needed information manually.

Our objective is to discover and solve the user's information need taking as starting point a set of input keywords. We divide this task into three main steps (see [4] for more details):

- Discovery of Keyword Senses**:** To discover the exact meaning of each input keyword, our search starts by discovering and building their senses (the precise meaning of a keyword in a context). In particular, this process is divided into three substeps (see Figure 2): 1) Extraction of keyword senses by consulting many third-party ontologies, 2) enrichment of discovered senses and removal of redundancy, and 3) disambiguation of the different possible senses for each keyword.

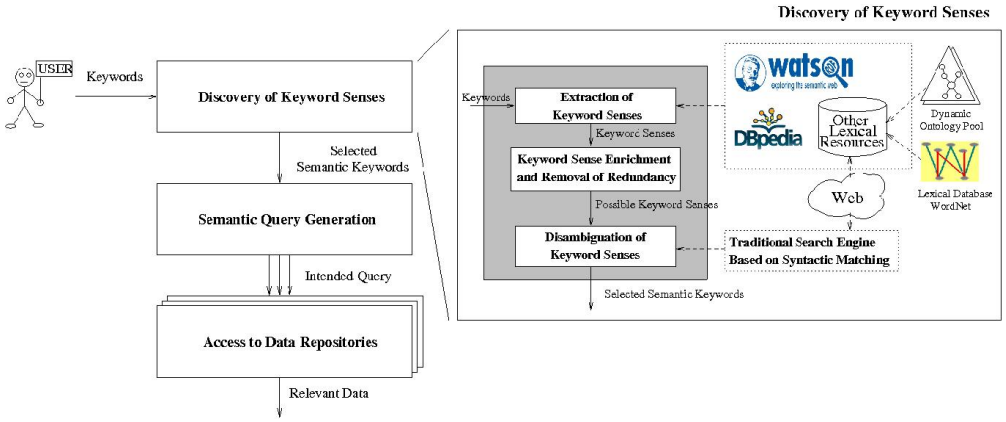

Figure 4: From plain keywords to data access: details of the discovery of keyword senses

- Semantic Query Generation**:** The output of the previous step is a set of keywords which have their meaning properly attached, which we call *semantic keywords*. Our system automatically integrates this information and, then, automatically builds a set of formal queries which, combining all the keywords, represents the possible semantics that could be intended by the user when s/he wrote the list of plain keywords. To do so, our system performs the steps shown in Figure 5:

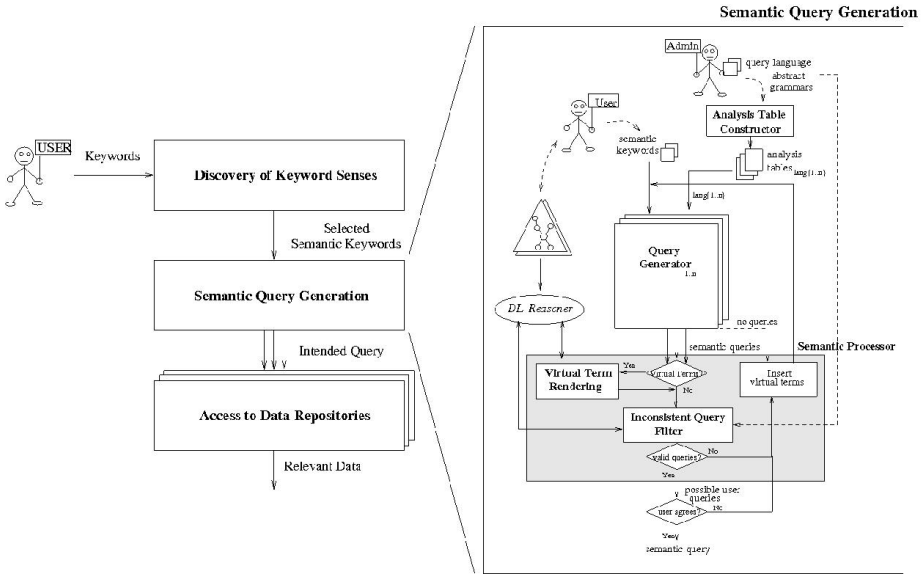

Figure 5: Multi-language query generation process

- Access to Data Repositories**:** Finally, once the user has validated the generated query that best fits his/her intended meaning, the system forwards it to the appropriate underlying structured data repositories (databases, Linked Data endpoints, etc.) that will retrieve data according to the semantics of such a query. This is not a trivial task: our system must be able to deal with different query processing capabilities, access methods, data models, and formats of the retrieved data.

3.2. Ontologies in Mobile Information Systems: SHERLOCK

The astonishing penetration of mobile computing in our daily lives, thanks to devices such as smartphones and tablets, leads us to a scenario where mobile users have access to huge amounts of information *anytime* and *anywhere*. However, users are starting to be overwhelmed with the amount of data they receive from different sources, as it is sometimes difficult for people to distinguish which information is valuable. For example, imagine a researcher attending a conference who arrives on an evening flight and needs to reach his/her city hotel. At first, s/he would be interested in transport information and s/he might need to know the different options (e.g., buses, metros, taxis, or car rental options), traffic conditions, and perhaps even where available parking spaces

are located. This information could be obtained by visiting a tourist office, searching a local transportation website, or even downloading a mobile app. After checking in, s/he could be interested in finding other nearby conference attendees to talk to them or even to go sightseeing (again, s/he should browse the Web to find information about interesting places to visit). Thus, it is the user her/himself who is in charge of knowing/finding the interesting and updated information sources and gathering and correlating all this information; even worse, s/he will have to know/find all these *updated* information about each city s/he visits.

Semantic Web techniques become particularly useful in these scenarios. These techniques can help to find the most appropriate information from a range of different sources by inferring useful information providers. Hence, information extracted from heterogeneous sources can be presented in an integrated way by using common representation models such as ontologies.

SHERLOCK (System for Heterogeneous mobilE Requests by Leveraging Ontological and Contextual Knowledge) [18], is our proposal to provide mobile users with interesting Location-Based Services (LBSs). SHERLOCK[4] relieves users from the need to obtain up-to-date information about the services they need. Using ontologies to model the knowledge related to these services, SHERLOCK devices exchange information among themselves, for example, about LBSs in the area. Also, with the help of a semantic reasoner, our system is able to determine which information could be interesting for a user regarding her/his context, and to obtain this information from objects nearby by leveraging the collaboration among devices. For that purpose, SHERLOCK deploys a network or mobile agents [16] which move from one device to another autonomously to be near the needed information source and collect data (see Figure 6).

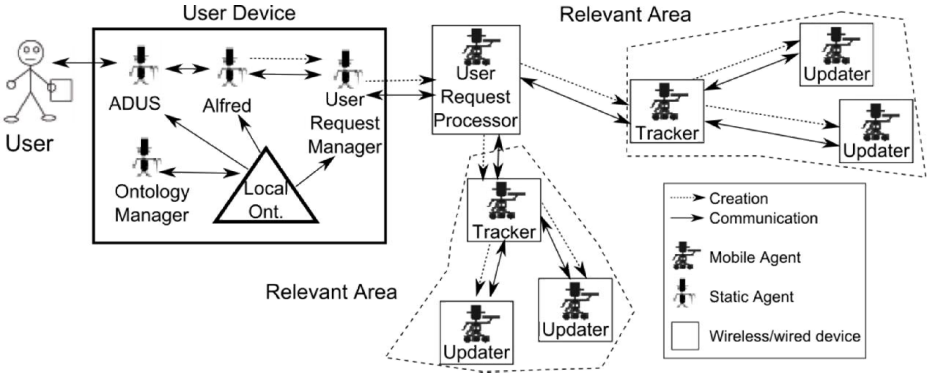

Figure 6: Agent network deployed to process a request in SHERLOCK

SHERLOCK is based on knowledge sharing among devices. Each participating device starts with a basic OWL local ontology containing the basic terms to define LBSs: concepts such as "Service", "Provider", "Parameter". SHERLOCK devices *learn* from the interaction with others: when two devices meet, they share part of their local ontologies. A SHERLOCK device that receives new knowledge integrates it with its local ontology, and thus it can use this new knowledge to provide the user with more interesting information. For example, the device of a user that lives in Zaragoza (Spain) knows transportation concepts such as "Taxi" or "Bus"; if the user travels to Thrissur (India) and his/her device starts communicating with others, it can learn that "Tuk-Tuk" (a vehicle defined in the ontology of local people as a private transport that carries people to a certain destination) is similar to a taxi, and thus, it can be interesting for a user that needs transport.

Agents in SHERLOCK are in charge of managing the local ontology of the device and share knowledge among other agents. When new knowledge is received, specialized agents apply automatic ontology matching techniques [9] to integrate the new knowledge into its local ontology.

We have developed a prototype of the SHERLOCK as an Android app. The prototype uses the OWL API, an ontology API to manage OWL 2 ontologies in Java applications, the JFact reasoner, and the SPRINGS mobile agent platform. In [5] we evaluated the performance of semantic reasoners on smart devices and our results show that current smartphones can handle reasoning on small/medium ontologies.

4. Conclusions

We have presented several examples of how the use of ontologies and semantic techniques helps to develop intelligent information systems, since the first approaches 30 years ago until today. The Semantic Web and its

[4] http://sid.cps.unizar.es/SHERLOCK

associated technologies are here to stay, and are no longer restricted to Web environments. In the Distributed Information Systems (SID) research group, we are following this line of thought and research, attempting always to go a step further into exploiting semantics to improve the capabilities of our systems. We have seen how the use of semantic techniques has allowed us to perform semantic keyword-based searches on heterogeneous information systems, or to develop a platform to provide smarter Location-Based services, among other examples.

Of course, there are still open issues and a long road to research in this field. For example, there is still no general purpose search engine that given a query returns the exact answer the user is looking for. Although some important knowledge bases have been generated in the last years that might contain this information, we are still far from a scenario where computers understand the meanings behind any type of data and data repository.

Acknowledgments. I would like to thank all the members of my research group for their work; without them the semantic-based systems described here would not exist. This work has been supported by the CICYT projects TIN2010-21387-C02-02, DGA-FSE, and TIN2013-46238-C4-4-R.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., & Patel-Scheneider, P. (2003). The Description Logic handbook: Theory, Implementation and Applications. Cambridge University Press.
2. Berners-Lee, T. (2007), Linked Data Design Issues, World Wide Web Consortium, Informal Notes.
3. Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. Scientific American, 284(5), 34–43.
4. Bobed, C. (2013). Semantic Keyword-based Search on Heterogeneous Information Systems. PhD in Computer Science, University of Zaragoza, Spain.
5. Bobed, C., Bobillo, F., Yus, R., Esteban, G., & Mena, E. (2014). Android Went Semantic: Time for Evaluation. In Proceedings of the 3rd International Workshop on OWL Reasoner Evaluation (ORE'14) (pp. 23–29). CEURWS.
6. Bobed,, C. Yus, R., Bobillo, F., Ilarri,S., Bernad, J., Mena, E., Trillo-Lado, R., and Garrido, A.(2015), "*Emerging Semantic-Based Applications*", Semantic Web: Implications for Technologies and Business Practices, Michael Workman (ed.), Springer Publishing, , Switzerland, To appear.
7. Bobillo, F., Delgado, M., Gómez-Romero, J., & Straccia, U. (2012). Joining Gödel and Zadeh Fuzzy Logics in Fuzzy Description Logics. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 20(4), 475–508.
8. Bobillo, F., & Straccia, U. (2011). Fuzzy Ontology Representation Using OWL 2. International Journal of Approximate Reasoning, 52(7), 1073–1094.
9. Euzenat, J., & Shvaiko, P. (2007). Ontology Matching (Vol. 18). Springer.
10. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., & Wang, Z. (2014). HermiT: An OWL 2 Reasoner. Journal of Automated Reasoning, 40(2–3), 89–116.
11. Gracia, J., & Mena, E. (2008, September). Web-based Measure of Semantic Relatedness. In Proceedings of the 9th International Conference on Web Information Systems Engineering (WISE'08) (Vol. 5175, pp. 136– 150). Springer.
12. Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5(2), 199–220.
13. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P. F., & Rudolph, S. (2012). OWL 2 Web Ontology Language Primer (Second Edition). (http://www.w3.org/TR/owl-primer/)
14. Mena, E., & Illarramendi, A. (2001). Ontology-Based Query Processing for Global Information Systems. Kluwer Academic Publishers.
15. Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., & Katz, Y. (2007). Pellet: A Practical OWL-DL Reasoner. Web Semantics: Science, Services and Agents on the World Wide Web, 5(2), 51–53.
16. Spyrou, C., Samaras, G., Pitoura, E., & Evripidou, P. (2004). Mobile Agents for Wireless Computing: The Convergence of Wireless Computational Models with Mobile-agent Technologies. Mobile Networks and Applications, 9(5), 517–528.
17. Trillo, R., Po, L., Ilarri, S., Bergamaschi, S., & Mena, E. (2011). Using Semantic Techniques to Access Web Data. Information Systems. Special Issue: Semantic Integration of Data, Multimedia, and Services, 36(2), 117–133.
18. Yus, R., Mena, E., Ilarri, S., & Illarramendi, A. (2014). SHERLOCK: Semantic Management of Location-Based Services in Wireless Environments. Pervasive and Mobile Computing.To appear.