



Polibits

ISSN: 1870-9044

polibits@nlp.cic.ipn.mx

Instituto Politécnico Nacional

México

Alonso-Rorís, Víctor M.; Santos Gago, Juan M.; Pérez Rodríguez, Roberto; Rivas Costa, Carlos; Gómez Carballa, Miguel A.; Anido Rifón, Luis
Information Extraction in Semantic, Highly-Structured, and Semi-Structured Web Sources
Polibits, vol. 49, 2014, pp. 69-75
Instituto Politécnico Nacional
Distrito Federal, México

Available in: <http://www.redalyc.org/articulo.oa?id=402640463009>

- How to cite
- Complete issue
- More information about this article
- Journal's homepage in redalyc.org

redalyc.org

Scientific Information System

Network of Scientific Journals from Latin America, the Caribbean, Spain and Portugal

Non-profit academic project, developed under the open access initiative

Information Extraction in Semantic, Highly-Structured, and Semi-Structured Web Sources

Víctor M. Alonso-Rorís, Juan M. Santos Gago, Roberto Pérez Rodríguez, Carlos Rivas Costa, Miguel A. Gómez Carballa, and Luis Anido Rifón

Abstract—The evolution of the Web from the original proposal made in 1989 can be considered one of the most revolutionary technological changes in centuries. During the past 25 years the Web has evolved from a static version to a fully dynamic and interoperable intelligent ecosystem. The amount of data produced during these few decades is enormous. New applications, developed by individual developers or small companies, can take advantage of both services and data already present on the Web. Data, produced by humans and machines, may be available in different formats and through different access interfaces. This paper analyses three different types of data available on the Web and presents mechanisms for accessing and extracting this information. The authors show several applications that leverage extracted information in two areas of research: recommendations of educational resources beyond content and interactive digital TV applications.

Index Terms—Information extraction, web data processing, semantic enrichment, data mining, web scraping.

I. INTRODUCTION

PRESENTLY, the Web hosts billions of pieces of information of all kinds and origins. A relevant part of this information can be freely accessed and reused. Under these assumptions, it would be possible to collect these freely shared pieces of information to enrich any database, that is, to complete the information stored locally using the huge amount of data available on the Web. This will enable software agents to increase their knowledge from a wide range of sources providing different points of view and levels of detail. This way, potentially complete information may be obtained.

Section II explains different strategies for information extraction depending on the type of source. Sources available may be classified according to the language used to expose this information (e.g., HTML, XML, SQL, RDF, JSON). Each of these types of sources has been designed to satisfy a different need (e.g., to be accessible to humans, to facilitate the communication between applications, to facilitate data storage). Enriching a system's database means taking the most of the available information sources online to complete

information stored locally. Insofar automated enrichment is concerned, the most interesting repositories are those specifically designed to be easily readable and interpretable by machines, that is, those offering their information according to a Knowledge Representation language like the ones proposed by the Linked Open Data (LOD) initiative [1]. The reason for that is because the enrichment process' complexity is dramatically reduced, and therefore its automation is greatly simplified. On the other side, other types of sources would need a previous translation or transformation process to represent them according to a normalized language, such as RDF. This latter process will require the active participation of experts.

Section III describes the application of information extraction techniques in the context of the iTEC project, which is the flagship FP7 project in the education area, financed by the European Commission with 12 million euros. iTEC tries to contribute to the conception of the classroom of the future, in which technology is complemented with the most innovative pedagogical approaches, which entail a major level of dynamism in the educational practice.

Section IV briefly discusses the use of information extraction techniques in two projects that have to do with the access to services through digital TV interfaces. The first one, Berce TV, is a portal that enables parents to access information on their children in early childhood education centers. The second one, Empleo, is a project aimed at providing an accessible job search interface through the TV.

Finally, we give conclusions and discuss future work.

II. INFORMATION EXTRACTION FROM THE WEB

In the Internet, you can find many different sources of information, being a lot of them available for free. The nature of the information that you can freely access to is very heterogeneous. From a technical point of view—and for convenience—, we make a classification of the sources of information by the format they use for encoding information¹²—which many times is associated with a particular protocol for accessing to that information. The

Manuscript received on January 7, 2014; accepted for publication on February 28, 2014.

The authors are with Department of Telematics, University of Vigo, Spain (e-mails: {victor.roris, Juan.Santos, carlosrivas, miguelgomez, lanido}@det.uvigo.es, robertoperezrodriguez@gmail.com).

¹Some of those formats are, among others, RDF, HTML, XML, and JSON.

²This idea is not new. See for instance the work in [2].

rationale for having so many different formats for encoding information is that each one is best suited for satisfying a particular need. For instance, RDF is well suited to be accessed from systems that belong to the so-called semantic Web; whereas HTML is particularly well suited to encode information in such a way that a Web browser is able to understand and display it in a computer's screen.

It becomes thus clear that, in order to populate a local database using external sources of information, the more semantic the source format is the less processing we need to perform at the local side. Therefore, purely semantic sources that share information in RDF format are by far the preferable ones, whereas sources that are meant to represent markup—for presentation—are the less desirable ones, because they need much more processing in order to squeeze them and extract the “semantic juice” out of them. Most of the times, that task is very labor-intensive, and no good automated procedures are yet available.

Below, we describe the three types of sources of information: semantic sources, highly-structured sources, and semi-structured sources.

A. Semantic Sources

The kind of sources of information from which we most easily can extract information are those pertaining to the Linked Open Data (LOD) initiative. The main purpose of the LOD project is to enrich the traditional Web by publishing open datasets—made of data items—which include links to data items in different datasets. Since the purpose of that format for representing information is that it must be easily understandable by machines, those sources are therefore very convenient for extracting information.

In order to be compliant with Linked Open Data conventions, information is represented in Resource Description Framework (RDF) format. Despite the fact that it was meant as a format for representing metadata, RDF is used—in the context of LOD—for modelling general information. The simple model for representing information that RDF implements accounts for its success in representing information beyond the limits of the Semantic Web³. RDF stores information as triples, which format every piece of information as an statement composed of subject, predicate, and object.

Information stored in RDF triples and published in public access points is commonly accessed by using the SPARQL query language—in fact, those public access points are frequently called SPARQL endpoints [3]. With SPARQL we can compose queries that are able to retrieve and manipulate data stored in those sources that comply with LOD conventions.

In addition to those sources, there exist a number of services that allow for launching searches on information stored in the

LOD network, much like Google does with the traditional Web. Those tools, such as Sindice [4] and SWSE [5] are very useful and enable us to discover a lot of interesting sources of information.

B. Highly-structured Sources

Outside of the LOD world we can find sources of information that, even though they do not use RDF as their information representation format, enable us to access information in a easily automatable way. We are referring to Web sources that expose information that is encoded with formats such as JSON and XML. Those are formats that encode pure—pristine—information, not mixing it with presentation markup. This makes the extraction process from these sources easier than from those in, for instance, HTML.

The main difference of these sources from those belonging to the LOD initiative is that the intended meaning of the information exposed at the remote sites is not automatically translatable to the meaning at the local side⁴. In addition to that, RDF triples are always RDF triples, whereas two different websites may use different strategies to encode their data, even though both use XML as the format for data representation.

In summary, highly-structured sources—using our terminology—enable us to easily retrieve information, but a dedicated processing is necessary for each and every one source in order to extract that information.

C. Semi-structured Sources

We call semi-structured sources to those that encode information in such a way that it is mixed with presentation markup. The most common of its kind are sources that expose information that is represented in HTML—that is to say, traditional Web pages. In order to retrieve and extract information from a source of this type, we need to browse the DOM tree of the page and identify concrete DOM elements that represent particular pieces of information. That is, we need to kind of reverse-engineering the web page in order to figure out the mappings between presentation-related tags and their meanings.

The good news is that the DOM tree is usually nothing else than a blur reflection of the pure information stored in a relational database⁵, after having been intermingled with presentation markup. Thus, once we get rid of all the presentation tags, we can get the most precious pieces of information⁶.

In summary, information retrieval and extraction from semi-structured sources is more costly than from highly-structured ones, because in addition to a dedicated processing for each and every source we also need to reverse engineer and get rid of a lot of uninteresting pieces of data.

⁴Conversely, LOD conventions allow for sharing information together with its meaning.

⁵According to [6], most Web sites use a SQL database as their persistence solution.

⁶That is the allegory of data mining, to get “gold” from heaps of “dust”.

III. APPLICATION IN THE ITEC PROJECT

iTEC promotes an educational practice in which students interact in small projects which include participation in events, speeches with experts, and all that seasoned with the use of technology. The *leiv motiv* of this new paradigm is “Resources beyond Content”. Thus, in iTEC educational resources go beyond educational content, frequently consumed in the form of textbooks, and they include technological resources, as well as cultural events and people who are experts on some knowledge area.

The initial assumption was that teachers were going to have a hard time trying to figure out which events could have the greatest relevance among such an enormous offer. This hypothesis was the main motivation for the design, development, and posterior rolling out of the SDE (Scenario Development Environment) [7], [8], which is a software system that works as a recommender, in such a way that a certain teacher, during the phase of preparing an educational experience, may rely on the SDE for selecting the most interesting events to be attended by learners during the performance of the educational experience. The SDE’s recommendation algorithm has several factors into account, such as the appropriateness of the event and the proximity of the event to the school [9].

After three years of project and more than 2000 pilots in schools across Europe, that starting hypothesis has shown to be wrong. That is to say, the number of events registered at the P&E directory is not of the scale that it was supposed to be, and thus it does not makes necessary to use a recommender. The main reason for the low number of registered events is that the registering process is very time consuming, in addition to being very error-prone. In order to tackle this drawback, it was implemented and integrated in the SDE an enrichment module, which automatically extracts and processes huge amounts of data coming from relevant Web sites that list applications, events, and experts in different areas of knowledge. Figure 2 shows events that were extracted from the Web in the SDE user interface.

A. Sources of information

All across the Internet there are a great number of websites that offer information on applications, events, and experts. On applications we extract information from three websites that we find particularly interesting:

- Softonic⁷ is a huge repository of information on standalone applications. From every entry on that repository we can extract information such as its description, the operating system required, tags, and what is very important: the rating from users.
- Softpedia⁸ is, similarly to softonic, a big database of information about applications.

⁷<http://www.softonic.com>

⁸<http://www.softpedia.com>

- AlternativeTo⁹ is an incredible resource for extracting meaning about the features of an application. For each application, AlternativeTo provides a list of “substitutes”—applications with similar functionalities and that may suit a similar need. This is extraordinarily useful from the point of view of populating a database aimed at serving as the knowledge base of a recommender, because we can extract very relevant information that may lead to more accurate recommendations. As an example, AlternativeTo enables us to extract information such as: Skype is similar to Google Hangouts, and Firefox is similar to Chrome.

Regarding experts, two websites are the most relevant ones:

- Google Scholar¹⁰ is an enormous database of researchers. From each registry we can extract very useful information, such as the name and position of researchers. To extract the information on their location—which in the context of iTEC is a very important piece of information, because experts of a near location should gain relevance in an hypothetical recommendation—we use an mechanism that takes as an input the position of the researcher (which is a text string that usually contains their affiliation) and their email address. The email address, in most cases, enables us to get the IP address of the expert’s institution—provided that the institutions hosts the mail server, which is often true. The affiliation can be processed with NLP software¹¹ to get rid of the position and retrieve the institution, and then geocode the institution.
- LinkedIn¹² is a huge social network targeted at professionals. Unlike Google Scholar, which sorts entries by relevance¹³, in LinkedIn we cannot measure the relevance of a particular professional. In order to overcome that difficulty—it is neither reasonable nor efficient to replicate all the LinkedIn public entries—we rely on Google’s relevance calculations. Let’s see that with an example, if we want to find experts on, let’s say, biology we look for “biology site:www.linkedin.com” in Google. In this way, we get a bunch of search results ordered by the relevance that Google assigns to those entries.

In order to get events we need to follow a brute-force approach¹⁴. All across Europe there are a great number of

⁹<http://alternativeto.net>

¹⁰<http://www.scholar.google.com>

¹¹To that end we use the Geocoder library, which is available for the Ruby programming language.

¹²www.linkedin.com

¹³Google Scholar measures relevance as the number of references that researchers gathered to all their publications.

¹⁴At the time of writing, we extracted information from the following websites: www.spainisculture.com, www.discoveringfinland.com, www.unesco.org, www.finnbay.com, www.openeducationeuropa.eu, www.visitportugal.com, www.ulisboa.pt, www.uio.no (the University of Oslo), visit-hungary.com, visitbudapest.travel, visitbrussels.be, www.belgica-turismo.es, www.ualg.pt (University of Algalve), noticias.up.pt (University of Porto), www.globaleventslist.elsevier.com (worldwide conference registry).

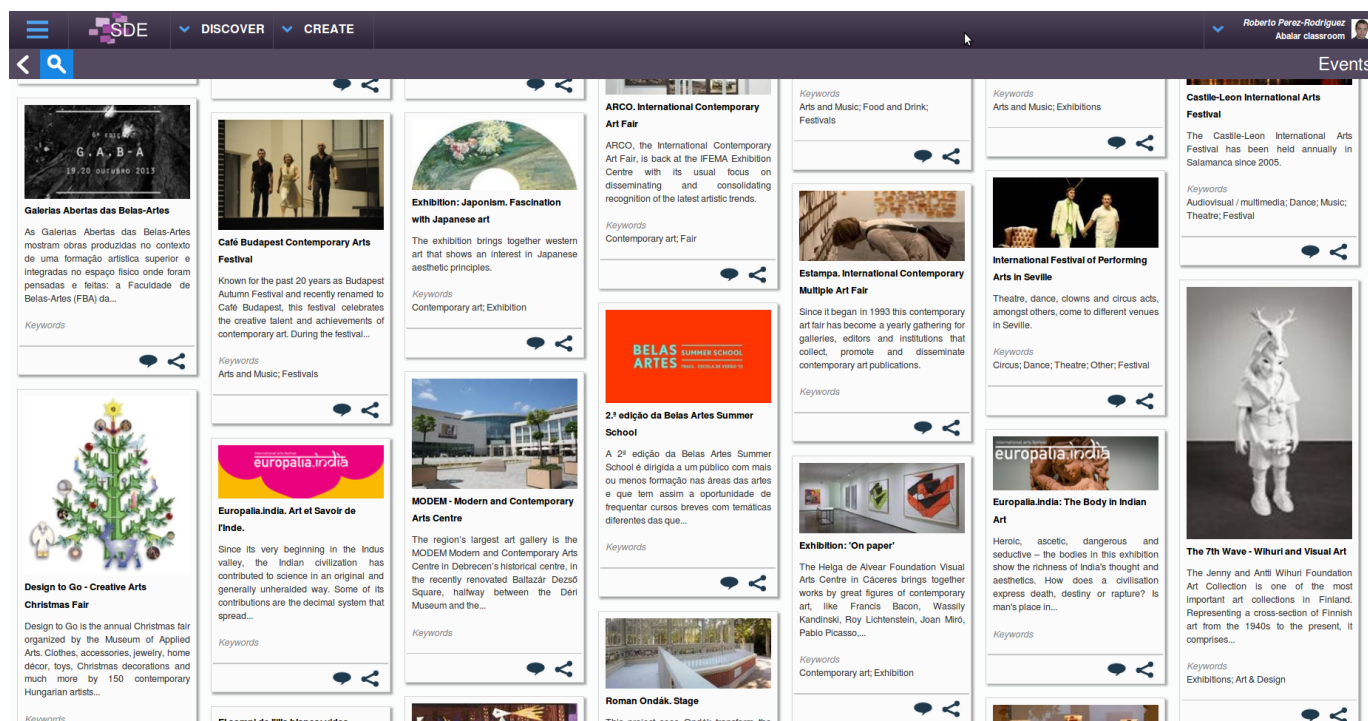


Fig. 1. Search results for events on “art” in the SDE user interface

websites that offer information on cultural events. Some of them, such as the Open Education Europa¹⁵ initiative, are at a European scale. Conversely, others such as the Spain is Culture¹⁶ portal are at a lesser scale.

B. Considerations on the Information Extraction Technique

There are several factors that you need to take into account in order to make an efficient scraping. The first one is that the scraping process of a web site may take quite a long time, so it should be performed as a background process. Besides, you have to take care of not launching too many requests to a given web server in a short time interval, because that could be seen as a Denial of Service attack, and the web server could block your IP.

Taking all the above into consideration, the scraping process starts with studying the HTML layout of the involved web pages. In order to do that efficiently we use the Firefox Firebug

extension. The list of events in the Spain is Culture portal serves as an example of our procedure. We use Firebug for inspecting the HTML structure of the document. The next step is to test CSS selectors using the JavaScript console that is embedded in Firebug, issuing some JavaScript sentences that use jQuery.

Once we identify the concrete DOM elements that contain relevant information¹⁷ we are ready to write some Ruby code that automates the parsing of applications, events, or experts. To that end, we make use of the Nokogiri¹⁸ library. Nokogiri is an open source HTML, XML, SAX, and Reader parser.

If we are dealing with events, once we got the venue of an event we try to get its coordinates. To that end, we make use of the Geocoder¹⁹ gem, which issues requests to the Google Maps API. Thus, we geolocate all the events. The following snippet of code shows how we parse the text that represents the address and, from that, we get the latitude and longitude of the event using Geocoder:

```
1 address = event.css('a')[0]['href'].split('/')[3]
2 coordinates = Geocoder.coordinates(address)
3 latitude = coordinates[0]
4 longitude = coordinates[1]
```

¹⁷We identify which element in the DOM tree contains the information on the venue of a certain event in the Spain is Culture portal. The same matching has to be made for each field in the detailed view—the name, the description, the date range and so on. To ease that process, we can use the JavaScript console to test CSS selectors in jQuery until we find the correct one.

¹⁸<http://nokogiri.org/>

¹⁹www.rubygeocoder.com

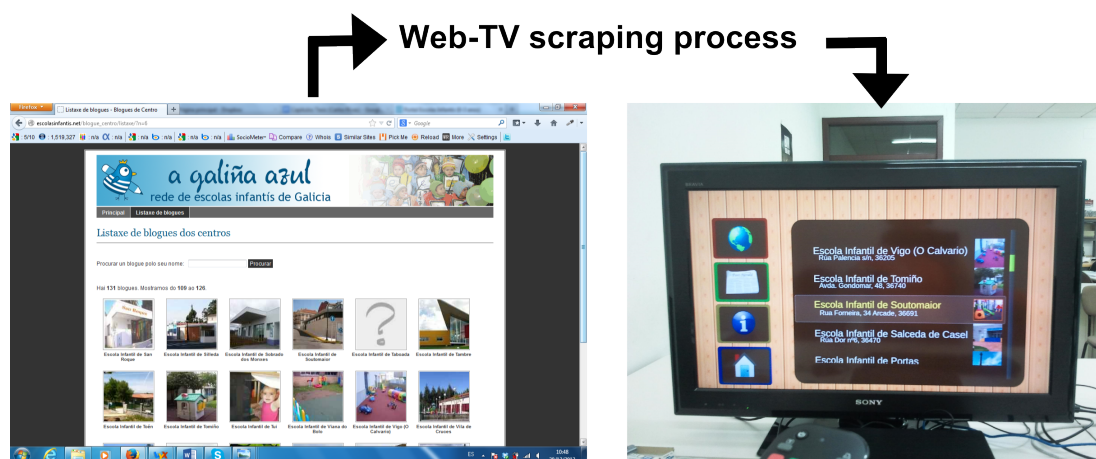


Fig. 2. The contents displayed on the TV screen are taken from the Web portal.

One important thing to consider when extracting events is to check if the scraped event is a new event, an update of an existing event, or a duplicate. To that end, we build a hash with the fields that make up an event. If the event is already present in the database but the calculated hash is different, then the update the fields of the event. Conversely, if the event is already present and the hash is identical, then we discard the scraped event because it is a duplicate. The following snippet of code shows how to create a hash with some of the fields that make up an event:

```
1 hash_event = Digest::MD5.hexdigest(name + description
    + start_date + end_date + latitude.to_s + longitude.
    to_s)
```

The filtering of duplicates is of special importance when dealing with events. Many times, when an event is published for the first time some of the fields might not be set yet. For instance, the final date might not be known yet; or maybe the venue is not fixed yet.

IV. INFORMATION EXTRACTION FOR INTERACTIVE DIGITAL TV APPLICATIONS

Visualisation of content on digital TV depends on how digital information is presented—the same is true on smartphones and tablets. As it can be observed in smartphones, it is increasingly frequent to develop native applications so that the user does not have to access contents on the cloud through a Web browser. Following the same approach, in order to show third-party content on a digital TV we need either accessing an API that offers information in a structured way or accessing directly to content in HTML format, received from HTTP requests; HTML content is then processed in order to obtain and structure the information that will be later shown in the TV screen.

Regarding the way of obtaining the information that will be displayed we can distinguish between two different models for obtaining information: a passive one and an active one.

Those depend on the needs of the final system that we are developing for. In the passive model, the user tells the system which information he/she wants to visualise at the time. The application is thus, at that very moment, in charge of obtaining the information requested, processing and displaying it in the TV screen—following the specifications for that information. On the contrary, in the active model, the application is continuously retrieving information and processing it so that it may be ready when the user demands to see it.

Following is the description of two applications that were built on environments for digital TV, each one making use of a different model for obtaining information.

A. BerceTV

This native application enables parents of children in early childhood education to access the information that is published in the Galician Portal of Early Education Schools²⁰ through their digital TVs. Thus, parents are able to access both information of their children as well as diverse public information that is shown in the portal.

The application acts as a middleware between the TV and the Web portal. In this way, every action the user performs is converted—in real time—into an HTTP request that is sent to the portal. The information thus received is processed, filtered, and structured so that it may be displayed in an interface adapted to the characteristics of a TV screen. Figure 2 illustrates this process.

Since all the information is obtained in real time, this can result on requesting redundant information such as images. In order to improve the performance and efficiency, the system includes a cache layer in charge of receiving and managing multimedia information. This operating mode prevents downloading duplicated information; besides, it keeps the response time low, because the obtaining of multimedia information is performed in independent threads, which avoid

²⁰<http://escolasinfantis.net>

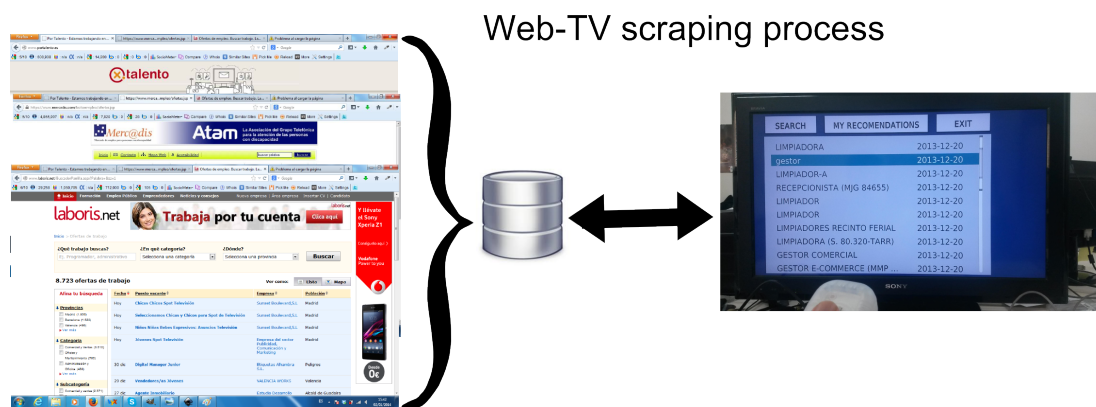


Fig. 3. Job offers are gathered from different sources and then displayed on the TV screen.

blocking the normal flow of the program in await for some information.

Besides obtaining public information that everybody can access to, one of the main features of this application is to let parents to access private information about their children. In order to access that information, every parent has a username and a password that enables them to enter the private area of the portal. The middleware uses those credentials to emulate the behaviour of a human person and access to the private information requested by the user. Thus, parents can see in their TV screen diverse information about their children, which is uploaded by educators.

B. Empleo

This application enables users to access information on job offers right from their TVs. One of the objectives of this application is to facilitate job search and, therefore, the application integrates tools for filtering search results in accordance with users' preferences. Figure 3 illustrates how information extraction is used in the context of this project.

In order to being able to apply filters on a heterogeneous set of job offers that were published in different Web portals, it is necessary to perform a homogenization task in advance, which entails that job offers must be available before users perform any searches. To that end, the active model for obtaining information is used. Thus, the system gathers information about job offers in an autonomous way. That is to say, the insertion, modification, and deletion of job offers is performed in accordance with information that is retrieved periodically from target sources.

The heterogeneity of information coming from different sources makes necessary to define a model of basic job offer, which is then enriched with the information obtained. All the information gathered is stored in a central database that is accessed by users through a Web Services API. As the system finds and updates information, the entries in the database are modified so that they may be accessed by digital TV clients.

V. CONCLUSIONS AND FUTURE WORK

This paper showed how different information extraction techniques were used in the context of several research projects.

In the context of the iTEC project, information extraction is crucial for having a database of technologies, events, and experts that is big enough to serve as the universe of things to recommend to teachers. The strategy based on information extraction avoids us having to delegate on personnel for entering information about technologies, events, and experts. In a similar way, in the context of the Empleo project, information extraction allows us to delegate on automated software tasks the work of entering information about job offers.

In the context of the Berce TV project, information extraction is used to retrieve information from the Web portal. That information is shown to the user in a way that results suitable for being displayed in a digital TV. This strategy based on information extraction allows for integrating a legacy system in a fully-functional way. In spite of not being the most elegant solution on paper, it is an efficient one in practice, because it allows to add a new user interface based on digital TV without having to tweak legacy code—which would entail a new concession contract.

Currently, we are working on a research line that has as its objective to automatically annotation and categorise information extracted from the Web.

ACKNOWLEDGEMENTS

The work presented in this paper was partially supported by the European Regional Development Fund (ERDF); the Galician Regional Government under agreement for funding the Atlantic Research Center for Information and Communication Technologies (AtlantIC); the Spanish Government and the European Regional Development Fund (ERDF) under project TACTICA; the European Commission's FP7 programme – project iTEC: innovative Technologies for an Engaging Classroom (Grant no. 257566); and the Spanish Ministry of Science and Innovation under grant “Methodologies,

Architectures and Standards for adaptive and accessible e-learning (Adapt2Learn)” (TIN2010-21735-C02-01). The content of this paper is the sole responsibility of its authors and it does not represent the opinion of the European Commission, or the Spanish Ministry of Science and Innovation, which are not responsible of any use that might be made of the information contained herein.

REFERENCES

- [1] C. Bizer, J. Lehmann, G. Kovilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, “CBpedia - A crystallization point for the web of data,” *Journal of Web Semantics*, vol. 7, no. 3, pp. 154–165, 2009.
- [2] C. Chang, M. Kaye, M. Girgis, and K. Shaalan, “A survey on web information extraction systems,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1411–1428, 2006.
- [3] E. Prud’hommeaux and A. Seaborne, “SPARQL Query Language for RDF,” World Wide Web Consortium, W3C Recommendation, Tech. Rep., 2008.
- [4] G. Tummarello, R. Delbru, and E. Oren, “Sindice.com: weaving the open linked data,” in *6th International Semantic Web Conference*, 2007.
- [5] A. Harth, A. Hogan, R. Delbru, J. Umbrich, S. O’Riain, and S. Decker, “SWSE: Answers Before Links!” in *6th International Semantic Web Conference*, 2007.
- [6] B. He, M. Patel, Z. Zhang, and K. Chang, “Accessing the deep web,” *Communications of the ACM*, vol. 50, no. 5, pp. 94–101, 2007.
- [7] A. Canas Rodriguez, V. M. Alonso Roris, J. M. Santos Gago, L. E. Anido Rifon, and M. J. Fernandez Iglesias, “The iTEC-SDE recommendation algorithms,” *International Journal of Systems and Control*, pp. 1–8, 2013.
- [8] A. Cañas Rodríguez, V. M. Alonso Roris, J. M. Santos Gago, L. E. Anido Rifón, and M. J. Fernandez Iglesias, “Providing event recommendations in educational scenarios,” in *Management Intelligent Systems*. Springer, 2013, pp. 91–98.
- [9] L. Anido, M. Caeiro, A. Cañas, M. Fernández, V. Alonso, and J. M. Santos, “ITEC - WP 10 D10.3. Support for implementing ITEC engaging scenarios V3,” The European ITEC project homepage, EUN Partnership AISBL, Rue de Trèves 61, B-1040 Brussels, Tech. Rep., 2013. [Online]. Available: http://itec.eun.org/c/document_library/get_file?uuid=1f1576cf-96b6-46bb-b34b-f66eca0f3cdf&groupId=10136