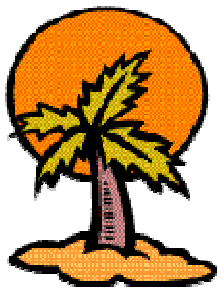
Actas del Taller de Trabajo Zoco'08 / JISBD
Integración de Aplicaciones Web

XIII Jornadas de Ingeniería del Software y Bases de Datos
Gijón, 7 al 10 de Octubre de 2008



http://www.tdg-seville.info/cfp/zoco

Organizadores
José L. Álvarez, José L. Arjona (Universidad de Huelva)
Rafael Corchuelo, David Ruiz (Universidad de Sevilla)

Ponentes
Rafael Berlanga, Ernesto Jiménez-Ruiz, Victoria Nebot, Ismael Sanz (Universitat Jaume I) · Carlos G. Figuerola, José Luis Alonso Berrocal, Angel F. Zazo Rodríguez (Universidad de Salamanca) · Francisco J. Pinto, Carme Fernández (Universidad de A Coruña) · Jorge Martinez-Gil, José F. Aldana-Montes (Universidad de Málaga) · Ana Flores Cuadrado (Telefónica Investigación y Desarrollo) y María Mercedes Martínez (Universidad de Valladolid) · Marcos López-Sanz, Jesús Domínguez, Carlos E. Cuesta, Esperanza Marcos (Universidad Rey Juan Carlos) · Rafael Z. Frantz (UNIJUI - Brasil), Rafael Corchuelo (Universidad de Sevilla) y Jesús González (Intelligent Dialogue Systems, S.L.) · Javier López, Alberto Pan, Fernando Bellas, Paula Montoto (Universidade da Coruña) · Dante I. Tapia, Juan F. de Paz, Sara Rodríguez, Javier Bajo, Juan M. Corchado (Universidad de Salamanca) · Juan M. Vara, M. Valeria de Castro (Universidad Rey Juan Carlos), Marcos Didonet del Fabro (Université de Nantes) y Esperanza Marcos (Universidad Rey Juan Carlos) · Arnoldo Zambrano, Cristina Cachero (Universidad de Alicante) y M. Ángeles Moraga (Universidad de Castilla-La Mancha) · José Luis Arjona y José Luis Álvarez, e Iñaki Fernández de Viana (Universidad de Huelva)

Agradecimiento
Financiación Proyecto IntegraWeb (TIN2007-64119, P07-TIC-02602)

Índice

Prólogo del Taller

José L. Álvarez[1], José L. Arjona[1], Rafael Corchuelo[2], y David Ruiz[2]

[1] Universidad de Huelva. Dep. de Tecnologías de la Información.
Escuela Politécnica Superior. Ctra. Huelva-La Rábida. Palos de la Frontera 21071.
{alvarez,jose.arjona}@diesia.uhu.es
[2] Universidad de Sevilla. Dep. de Lenguajes y Sistemas Informáticos.
ETSI Informática. Avda. Reina Mercedes, s/n. Sevilla 41012.
{corchu,druiz}@us.es

1. Motivación

Nuestro interés principal es estudiar los problemas relacionados con la integración de aplicaciones web que tan sólo ofrecen una interfaz de usuario. Estas aplicaciones suelen ser fuentes de datos muy valiosas, pero no resulta fácil aprovecharlas a gran escala debido a las dificultades que supone integrarlas dentro de procesos de negocio automatizados.

Desde el punto de vista de la investigación, se trata de un tema que está atrayendo a diversas comunidades: la de Base de Datos, por ejemplo, está prestando mucha atención al desarrollo de lenguajes de consulta estructurados específicos para la Web; la de Inteligencia Artificial, está más centrada en el diseño de herramientas que permitan extraer la información de las páginas web y dotarla de un significado bien definido usando ontologías; la de Minería de Datos también ha mostrado interés por este problema y centra su esfuerzo en el desarrollo de técnicas de verificación de información; etcétera. Sin duda alguna, creemos que una de las comunidades que más puede beneficiarse de todos estos avances multidisciplinares es la de los Servicios Web, ya que las aplicaciones que tan sólo ofrecen una interfaz de usuario suelen suponer una complicación en muchos casos insalvable a la hora de diseñar un proceso de negocio basado en estándares como, por ejemplo, BPMN/BPEL. Para la comunidad de Ingeniería del Software, este tipo de aplicaciones también supone un reto interesante ya que hasta el momento no existe ninguna metodología adaptada a este tipo de problemas.

Zoco'08 continúa una andadura que comenzó en 2001 con el objetivo de ofrecer un foro apropiado a los investigadores interesados en el desarrollo de aplicaciones de negocio en la Web. En esta edición ha pretendido ser un punto de encuentro multidisciplinar en el que diversas comunidades hayan podido discutir propuestas relacionadas con la integración de aplicaciones web.

2. Contribuciones

En total han sido once las contribuciones aceptadas en esta edición del taller. En su conjunto creemos que proporcionan una visión bastante amplia del trabajo que se está realizando en el campo de la integración en distintos grupos de investigación y, en algunos casos, en empresas de nuestro entorno.

1. El primer trabajo, presentado por Berlanga, Jiménez-Ruiz, Nebot y Sanz, de la Universitat Jaume I, describe un sistema para en análisis de formularios y la extracción de información con el objetivo de construir ontologías de forma automática.
2. Figuerola, Alonso y Zazo, de la Universidad de Salamanca, presentan un trabajo sobre extracción de información con el que es posible recuperar los fragmentos de texto dentro de una página web más útiles para poder responder una consulta.

3. Pinto y Fernández, de la Universidade da Coruña, han tratado el problema de la desambiguación del sentido de las palabras usando un modelo vectorial y WordNet.
4. El cuarto trabajo está relacionado con la comparación de fuentes de información heterogéneas con el objetivo de realizar alineamiento de ontologías. Este trabajo ha sido desarrollado por Martinez-Gil y Aldana-Montes, de la Universidad de Málaga.
5. El quinto trabajo ha sido desarrollado por Flores, de Telefónica Investigación y Desarrollo, y Martínez, de la Universidad de Valladolid. En él presentan su propuesta de plataforma de búsqueda semántica en fuentes de información heterogénea con una aplicación al campo concreto del comercio electrónico.
6. López-Sanz, Domínguez, Cuesta y Marcos, de la Universidad Rey Juan Carlos, son los autores del sexto trabajo, que trata sobre el modelado de estrategias de coordinación inspiradas en tácticas deportivas, como método efectivo para modelar la integración de aplicaciones haciendo uso de tecnología de servicios web.
7. Frantz, de la Universidad Unijuí (Brasil), Corchuelo, de la Universidad de Sevilla, y González, de la empresa Intelligent Dialogue Systems, S.L., presentan sus avances en el diseño de un lenguaje específico de dominio para el diseño y despliegue de soluciones de integración.
8. López, Pan, Bellas y Montoto, de la Universidade da Coruña, presentan una arquitectura de referencia para el desarrollo de mashups.
9. El trabajo de Tapia, Paz, Rodríguez, Bajo y Corchado, de la Universidad de Salamanca, enfoca el problema de la integración desde una perspectiva completamente diferente, ya que su objetivo es la búsqueda de sinergia entre la orientación a servicios y los sistemas de agentes distribuidos como forma efectiva de lograr la integración de aplicaciones.
10. El décimo trabajo vuelve a tratar el problema de la integración desde la perspectiva de la mezcla de modelos para automatizar el desarrollo dirigido por modelos de aplicaciones web. Ha sido desarrollado por Vara, Castro, y Marcos, de la Universidad Rey Juan Carlos, y Didonet del Fabro, de la Université de Nantes (Francia).
11. El penúltimo trabajo presentado corresponde a Zambrano y Cachero, de la Universidad de Alicante, y Moraga, de la Universidad de Castilla-La Mancha. En él presentan una primera experiencia de integración a gran escala en un escenario real: los sistemas de información de los municipios costarricenses.
12. El último trabajo presenta unas ideas preliminares sobre el problema de la verificación de la información devuelta por un wrapper y ha sido desarrollado por Arjona, Álvarez y Fernández de Viana, de la Universidad de Huelva.

3. Comités

3.1. Programa

A continuación aparecen los nombres y afiliaciones del comité de programa y de organización de este taller. Damos las gracias a todos ellos por su apoyo y contribución desinteresada para conseguir que esta edición sea un éxito.

- Rafael Berlanga, Universidad Jaume I
- Emilio Corchado, Universidad de Burgos
- Juan M. Corchado, Universidad de Salamanca
- Vicente Luque, Universidad Carlos III de Madrid
- José Manuel López-Cobo, Ximetrix, S.A.
- Esperanza Marcos, Universidad de Rey Juan Carlos
- Francisco Moriana, MP Sistemas, S.A.
- Juan Pavón, Universidad Complutense de Madrid

- Alberto Pan, Universidad de la Coruña
- Macario Polo, Universidad de Castilla-La Mancha
- Nieves Rodríguez, Universidad de la Coruña
- Hassan A. Sleiman, Indevia Solutions, S.L.L.
- Miguel Toro, Universidad de Sevilla

3.2. Organización

- José L. Álvarez, Universidad de Huelva
- José L. Arjona, Universidad de Huelva
- Rafael Corchuelo, Universidad de Sevilla
- Iñaki Fernández de Viana, Universidad de Huelva
- Rafael Z. Frantz, Unijuí, Brasil
- Pablo Palacios, Universidad de Huelva
- David Ruiz, Universidad de Sevilla

FAETON: Form Analysis and Extraction Tool for ONtology construction

Rafael Berlanga[1], Ernesto Jiménez-Ruiz[1], Victoria Nebot[1], y Ismael Sanz[2]

[1] Departament de Llenguatges i Sistemes Informàtics
[2] Departament d'Enginyeria i Ciència de la Computació
Universitat Jaume I, Castelló, Spain

Resumen. This paper presents a method for semi-automatically building tailored application ontologies from a set of data acquisition forms. Such ontologies are intended to facilitate the integration of very heterogeneous data generation processes and linking with external sources useful to enrich the generated data. The resulting tool is being applied to the medical domain where well-known knowledge and linguistic resources are publicly available. Preliminary results are shown in this paper, and demonstrate that the approach can perform effectively. However, further experiments should be performed to demonstrate their applicability for hard integration issues.

1 Introduction

The Semantic Web is aimed at facilitating the automatic processing of Web contents and services. Nowadays there are many methods to migrate current Web contents to this semantic space. Most of them have concerned with the semantic annotation of Web pages by using Information Extraction techniques (see [12] for a review). Other works like [1] have concerned with analysing data records published in the Web (e.g. book information) to build proper ontological instances that facilitate their integration.

In this work we focus on the migration of data acquisition forms to the semantic space. More specifically, we deal with data acquisition forms coming from disparate workflows (usually from different organisational units), which generate data to be processed by Decision Support Systems. The high heterogeneity of these forms in both contents and structures makes it crucial to enrich them somehow by means of external knowledge resources (e.g. thesauri like WordNet and UMLS). In our case, we propose to build tailored application ontologies so that forms and data can be processed in an effective way. Moreover, the annotations provided by these ontologies are also useful for linking forms and data to existing external sources like bibliographic repositories and publicly available databases also annotated with the same resources.

Our application scenario for validating the proposed approach is a biomedical application developed in the Health-e-Child (HeC) project[3]. Data acquisition processes are medical protocols that have been specified within hospital departments. Thus, forms present very different formats and contents depending on the clinician specialities (e.g. Rheumatology, Cardiology, Oncology, etc.) and what they consider relevant for characterizing patients. Generated data is usually expressed in XML and stored, after anonymisation, in a distributed repository in order to be shared by other clinicians of the project. Among the tasks performed over this repository we highlight that of decision support, which consists on applying data mining algorithms and visualisation tools over a set of selected patients to find new evidences for proper diagnosis, follow-up and treatments.

In this scenario, application ontologies are essential to perform the following tasks: (1) to homogenise vocabulary and terminology used in the data forms, (2) to discover data forms

[3] `http://www.health-e-child.org`

providing similar information, (3) to organise data fields according to the concepts provided by the ontology, and (4) to link forms and data to well-established external sources like PubMed and UMLS.

The main differences of our approach from other works in the literature mainly reside in the nature of the elements to be semantically annotated. Contrary to works based on Information Extraction techniques, we do not deal with plain text sources. Usually, forms contain many small text sections (labels) consisting of only a few words. Unlike wrapping approaches, we have no regularities to exploit as we deal with the data forms (schemas) instead of the data they can generate (instances), other than the implicit conventions of form layout [15, 3]. In other words, we face a more heterogeneous scenario where patterns are scarce.

The rest of the paper is organised as follows. Section 2 is dedicated to the extraction of structural properties of the forms. In Section 3 we present the semantic annotation of the forms as well as how related knowledge is imported from external domain ontologies. Section 4 is dedicated to the preliminary experiments. Finally we give some conclusions and present future work.

2 Extracting Structural Properties from Forms

The first techniques for the automatic acquisition of knowledge from data acquisition forms were developed in the context of the "paperless office" vision, and focused on for the processing of paper-based documents; a survey can be found in [13]. Most of this effort was focused in low-level techniques such as image segmentation and optical character recognition (*document analysis*), followed by a phase of so-called *document understanding*, which involved deriving a logical structure for the document.

In contrast with the approaches for paper-based documents we have just outlined, we will be dealing with electronic forms. Obviously, this greatly reduces the difficulty of document analysis, but it does not eliminate the need for it. Consider, for example, the case of forms produced by popular database tools such as Microsoft Access, or many HTML forms in Intranets, which provide little structural metainformation. The designers of these forms typically assume they are designed to be interpreted only by humans, and therefore they rely on the same visual conventions used by their paper-based counterparts.

2.1 Form analysis

In order to perform the analysis of an electronic form, we assume that we have access to the following basic information about it:

- The basic components in a form: *labels* (i.e. simple text) and standard *controls* such as text entries and combo boxes.
- Visual and geometric features of the basic components: font size, colour, absolute positions and rectangular areas are all useful and generally available.
- Type information associated to the controls, including the data type (e.g. a combo box may be restricted to a lists of acceptable inputs) and, if possible, the data source to which a control is linked.

It is important to note that additional metainformation may be available depending on the original format of the form. For instance, Microsoft Access forms provide richer data than HTML forms, such as the data source of each control, which are exploited by FAETON.

This metainformation is processed during the *analysis phase*, whose goal is to obtain a logical representation of the form. In our case, this representation will contain the following related models:

Logical model The logical model is the main result of this phase, and contains the inferred hierarchy of elements in the original form. According to the natural groupings found in forms, the model is structured as a multi-level structure: forms are composed of sections, which in turn may contain labels, control groups, and independent controls. This model also contains basic horizontal relationships between controls and their corresponding labels.

Layout model Even in a poorly designed form, there is a limited number of visual styles (combination of font, colour, alignment, and so on) present. It is possible to automatically find a relationship between these styles (i.e. finding out which are more salient), which convey structural information. The layout model contains these descriptions.

Link model The interpretation of many forms does not proceed linearly; rather, it is directed by indications such as "if the answer to such question is yes, then proceed to that other question". The correct extraction of these relationships is crucial, since they contain information about the workflow that underlies the form.

Figure 1 shows a fragment of data acquisition form processed by our approach. The form has two sections, entitled "Brain MRI" and "Spinal MRI". Control names are drawn in grey (e.g. MRINORM, PRECON1, etc.) The rest of strings are labels. These labels can be associated to controls (e.g. "Pre-contrast Intensity" to PRECON1), control values (e.g. Normal and Abnormal) and titles (e.g. "Brain MRI").

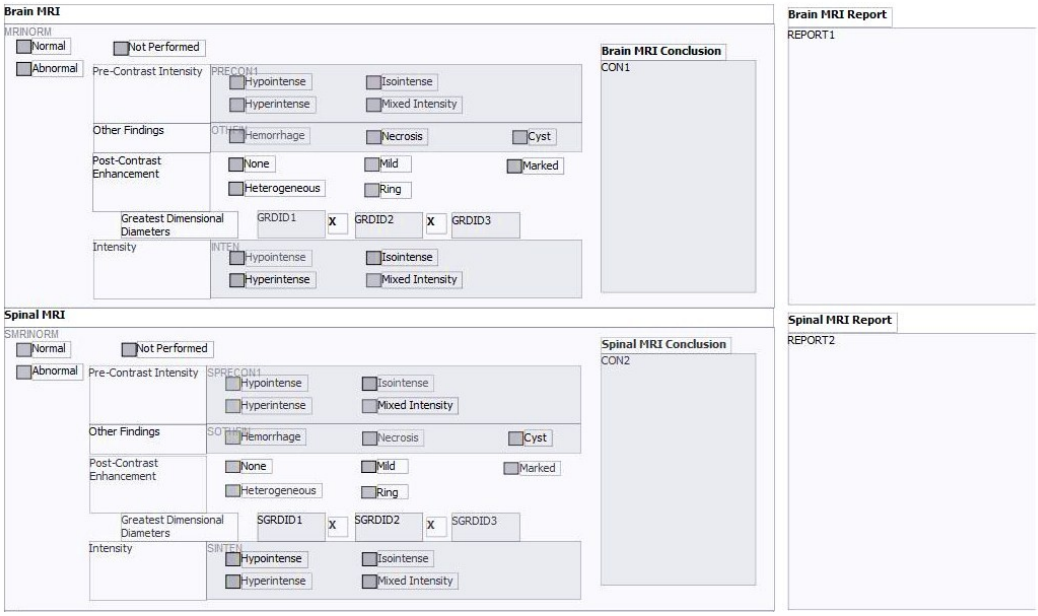


Fig. 1. Prototype showing the segmentation produced for the form *Diagnosis*.

2.2 Algorithm for extracting structural properties

In order to obtain the models described above, we follow the following steps:

1. First, we obtain an inventory of visual styles (layout model). A ranking of the visual relevance of the styles is computed using an entropy function which can be parametrized by the user.

2. Next, a segmentation is performed using a variant of a X-Y tree decomposition. This technique, originally proposed in [7], consists of successively splitting a page by horizontal and vertical cuts on white spacing; the end result being a segmentation of the page into hierarchically related boxes.
3. The information obtained by the segmentation and the layout analysis is used to build the hierarchical structural information, using techniques adapted from [14]. These techniques analyse *visual influence areas* (which allow us to find which labels and controls fall under a higher-level heading) and positional relations (adjacency and nesting) to find out relations between labels and controls, and in particular, which controls are organized as a grid, described by row and column headers. Alternative approaches can be found in [15, 3, 9, 10].
4. Labels are then analysed using simple techniques to find out workflow relations.
5. Finally, every control is analysed to extract type information. A particularly useful feature is the list of possible values.

The models obtained by this process are encoded using an XML-based internal representation, and form the basis for the semantic enrichment described in the remainder of this paper.

3 Bringing Knowledge to Forms

The main aim of the present work is to enrich data acquisition forms with both knowledge extracted from the forms and existing knowledge in external sources. For this purpose, the first step to be done consists of identifying known concepts from the different elements of the forms, mainly short descriptions associated to form controls. This task is known as the semantic annotation of the form components.

In our scenario, we can take profit from several annotation tools for biomedical texts that have emerged during the last years. Most of them use the Unified Medical Language System (UMLS)[4] since it provides a rich lexicon with a wide coverage over the biomedical domain. Additionally, it contains a potential taxonomy and ontology over the covered concepts. Several works have used successfully UMLS to perform semantic annotation [6, 2]. However, it is worth mentioning that managing this resource is not trivial, as it contains around 2 million of strings associated to 1 million of concepts.

3.1 Annotating Forms

Once forms have been analysed and segmented as described in Section 2, each form control is associated to a text segment describing it (i.e. its labels) and an optional list of values (usually a list of single words). Other interesting textual sections that deserve analysis are form titles and section titles as they provide useful contextual information. All these form elements are subject to semantic annotation.

As previously mentioned, works about semantic annotation are mainly focused on free-text sources. In our case, annotations must be done over a set of very short text sections extracted from the forms. A series of differences can be identified between annotating free-text and data forms. Firstly, there are a lot of concepts from the thesaurus that should be avoided when annotating free text. These concepts are related to highly frequent and ambiguous words such as *sex*, *name* and *date*, as well as words indicating broad categories such as *disease*, *sign*, etc. However, when annotating forms these concepts are very relevant for integration issues. Second, semantic annotation of texts use to be oriented toward statistical

[4] `http://www.nlm.nih.gov/research/umls/`

analysis of concepts to find new knowledge from the literature. In this way, approximate annotations are enough to capture interesting correlations. However, forms require very precise annotations for the controls as any error can produce a disastrous effect when processing their associated data.

Another relevant issue about semantic annotation is the way annotations are generated. In MetaMap [2] a series of techniques based on Natural Language Processing are applied to find the noun phrases in texts that fit the best to existing concept descriptions (i.e. lexicon). As a result, MetaMap returns a list of concepts ordered by their similarity w.r.t. the identified noun phrase. Instead, dictionary-based approaches like MWT (Multi-Word Tagger) [11] return the concepts whose descriptions match exactly with a segment of the text.

3.2 Expressing Semantic Annotations

As earlier mentioned, the annotation of form controls will be quite useful in order to classify and define groups of forms and controls. Therefore, it is necessary to include the hierarchical relationships involved by the found annotations in order to enable such a classification. For this purpose, we build a simple ontology module that allocates the annotations and their relationships as follows:

- Each control of the forms will be represented by a uniquely identified concept A_i. The UMLS concepts associated to this control, $CUI_1, \cdots, CUI_n$, characterize it by means the following axiom:

$$A_i \sqsubseteq \exists \mathsf{hasUMLS}.CUI_1 \sqcap \cdots \exists \mathsf{hasUMLS}.CUI_n$$

- Each UMLS concept CUI_i used in the annotations will be related to its *semantic types* [5], $ST_1, \cdots, ST_k$, with the following axioms:

$$CUI_i \sqsubseteq ST_1 \ \cdots \ CUI_i \sqsubseteq ST_k$$

- Finally, by using a fragment generator for UMLS [8] we add all the *is-a* relationships involved by the identified concepts. These axioms have the form $CUI_i \sqsubseteq CUI_j$.

3.3 External Domain Ontologies

The previous subsection has presented the link between the protocol attributes and the UMLS sources: with the Metathesaurus by means of the annotation CUIs and with the Semantic Network by means the inherent semantic types of the CUIs. However, this linking may not be enough for complex classifications, since the UMLS semantic network provides an upper level granularity. Domain ontologies will become a key point since they will represent the bridge between the UMLS lexicon and the UMLS semantic network.

The use of medical ontologies like Galen and NCI poses new challenges in this scenario: (1) selection of the proper ontology or ontologies that better characterise the form contents, (2) extraction of the appropriate ontology fragments from the selected ontologies.

In current works about ontology segmentation and modularisation (e.g. [4, 5]) it is usual to define an input set of concepts and properties to express the requirements of the user with respect to the desired module contents. This input set is also referred as input signature as it contains the symbols of the target ontology that will guide the extraction process.

In our application, input signatures are generated from the semantic annotations associated to controls. In order to get the proper symbols of the target ontology we need a

[5] UMLS Semantic Network: `http://www.nlm.nih.gov/research/umls/meta3.html`

mapping between UMLS concepts and the ontology symbols. In our preliminary experiments we have used the NCI ontology because it already provides these mappings in the OWL file [6]. Future work we will be focused on automatically linking other medical ontologies like Galen to UMLS.

3.4 Building the Application Ontology

The main goal of FAETON is to generate an application ontology for semantically describing in an homogeneous way a set of heterogeneous data acquisition forms. This application ontology comprises three layers (see Figure 2), namely:

- **Level 1:** Top level knowledge represented by means the UMLS Semantic Network.
- **Level 2:** The ontology fragments covering the relevant domain knowledge for the annotated protocol attributes.
- **Level 3:** The knowledge coming from the forms. This consists of the annotated UMLS concepts as explained in Section 3.2, and the structural relationships between controls derived from the forms.

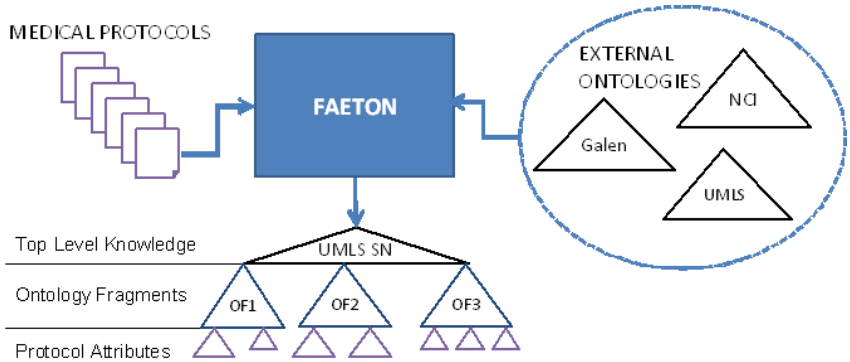


Fig. 2. General FAETON Architecture for the medical scenario

Elements of Level 1 and 2 and the UMLS annotations have been treated in the previous sections. They are included in the application ontology as imported modules. Table 1 shows the possible axioms that can be established from the structural information extracted by FAETON.

4 Prototype and Preliminary Experiments

We have produced a prototype of FAETON, which currently features a full implementation of form segmentation and a selection of semantic enrichment techniques. The prototype has been developed in Python using the GTK+ libraries for the visualisation and analysis of forms. In this section we show the preliminary results obtained with this prototype over a set of forms representing medical protocols.

[6] NCI Thesaurus properties:
`ftp://ftp1.nci.nih.gov/pub/cacore/EVS/ThesaurusSemantics/Properties.pdf`

Axiom Pattern	**Conditions**
$E_{base} \sqsubseteq \exists \mathsf{has\_}E_{form}$	E_{base} is the underlying entity on which forms are focused (e.g. Patient). The entity E_{form} is identified from the form's title.
$E_{form} \sqsubseteq \exists \mathsf{has\_}E_{section}$	$E_{section}$ is the concept associated to a section of the form represented by E_{form}.
$E_x \sqsubseteq \exists \mathsf{hasControl}.E_{control}$	E_x represents the concept of the closest structure to the control represented by $E_{control}$, which can be either a section or a form.
$E_{control} \sqsubseteq \exists \mathsf{hasM\_}E_{control}.DT_{control}$	$DT_{control}$ is the numeric data type associated to the control represented by $E_{control}$.
$E_{control} \sqsubseteq \exists \mathsf{has\_}E_{control}.\mathsf{BoolDomain}$	In this case the data type is boolean.
$E_{control} \sqsubseteq \exists \mathsf{has\_}E_{control}.\{E_{v_1} \cdots E_{v_k}\}$	The control represented by $E_{control}$ has associated a set of values $\{v_1 \cdots v_k\}$ which are represented by the entities $E_{v_1} \cdots E_{v_k}$.
$E_{row_i} \sqsubseteq E_{grid_{i,1}} \sqcap \cdots \sqcap E_{grid_{i,n}}$	Given a grid structure, each entity represented at each row is related to the entities represented by the controls placed in that row. These controls take its concepts from their corresponding header labels.

Tabla 1. Axiom patterns expressing the structural semantics of a form. E_x represents the concept associated to the form element x.

Combining structural information and semantic annotations. The accuracy of the final application ontology will depend on both the quality of the semantic annotations and the quality of the inferred structures from forms. If controls are incorrectly tagged, then the generated axioms will produce wrong classifications. Whereas if the form's structure contains errors (e.g. elements incorrectly related) the system will produce both wrong annotations and wrong axioms. Additionally, controls without annotations will be useless for integration and classification issues. In order to refine both the extracted annotations and the structures, the system can assess them to detect flaws and to give experts clues to fix them up. Two kind of flaw detections are currently applied: (1) Identify form elements whose annotations differ from the majority of its nearest neighbours in the form, and (2) associate temporarily each untagged control to a new concept placed under the NCA concept of its neighbours in the form.

Evaluation of Annotations. In the preliminary experiments we have evaluated two methods for annotating the data acquisition forms: MetaMap [2] and MWT [11]. Table 2 shows the coverage achieved by the tested semantic annotators for seven protocols within the *Brain Tumours* domain in the HeC project. We have manually evaluated the results by considering the following measures: **right annotations**/*incomplete or wrong annotations*/*Non-tagged controls*. Comparing the two annotators, it can be concluded that MWT produces slightly better results in precision, and MetaMap slightly better results in recall due to its flexibility with term variations. Therefore, both annotators could be combined to complement each other (see last column of the table).

To sum up, nearly 90% of the controls have some annotation associated, although some of them are not as precise as required. For example, we have found 7 wrong annotated elements in the section *Permanent Sequels* of the protocol *Follow up Data*. This is due to the fact that UMLS does not cover the knowledge needed to express these elements. Thus, further knowledge should be extracted from other sources in order to the able to correctly

Protocol	Controls	MetaMap	MWT	Missing in both
Baseline	10	**5** / *5* / 0	**8** / *1* / 1	0
Chemotherapy	47	**36** / *9* / 1	**32** / *8* / 6	0
Diagnosis	74	**54** / *12* / 8	**59** / *9* / 6	4
Follow up Data	29	**19** / *7* / 3	**22** / *4* / 3	3
Radiotherapy	11	**9** / *2* / 0	**10** / *0* / 1	0
Relapse	8	**8** / *0* / 0	**8** / *0* / 0	0
Second Look	27	**18** / *6* / 3	**20** / *3* / 4	2

Tabla 2. Coverage and precision in the *Brain Tumours* protocols

annotate these controls. On the other hand, we have found 4 untagged elements that can be placed correctly in the ontology thanks to the their neighbours in the form. Finally, partial annotations should be carefully analysed. Although they contain correct concepts, they do not fully express the semantics of the control in the form. As an example, in the section *Patient Status* of the *Follow-up Data* protocol, some of the controls such as *Alive with disease* and *Dead of disease* have been homogeneously annotated with *diseases* concepts, but they not capture the very sense of these controls. Since no concepts exist in UMLS that can cover these types of meanings, these annotations are considered incomplete.

5 Conclusions

This paper has presented a new method for semi-automatically building tailored application ontologies from a set of data acquisition forms. Preliminary results are satisfactory, but some issues remain open as not all the elements of the forms can be semantically annotated due to the limitations of the external resources. Future work should address more complex logical representations for the application ontologies, that is how to combine atomic annotations to provide more powerful representations. Another issue to address in future work is the use of logical queries to perform integration and linking tasks. Finally, new ontological resources need to be included in the whole process in order to cover as much as possible the form contents.

Referencias

1. José Luis Arjona, Rafael Corchuelo, David Ruiz, and Miguel Toro. From wrapping to knowledge. *IEEE Trans. Knowl. Data Eng.*, 19(2):310–323, 2007.
2. AR Aronson. Effective mapping of biomedical text to the UMLS metathesaurus: the MetaMap program. *Proc AMIA Symp*, pages 17–21, 2001.
3. Hai He, Weiyi Meng, Yiyao Lu, Clement Yu, and Zonghuan Wu. Towards deeper understanding of the search interfaces of the deep Web. *World Wide Web*, 10(2):133–155, 2007.
4. Ernesto Jimenez-Ruiz, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga. Safe and economic re-use of ontologies: A logic-based methodology and tool support. In *European Semantic Web Conference (ESWC)*, volume 5021 of *LNCS*, pages 185–199, 2008.
5. Ernesto Jiménez-Ruiz, Rafael Berlanga Llavori, Victoria Nebot, and Ismael Sanz. Ontopath: A language for retrieving ontology fragments. In *Ontologies, DataBases, and Applications of Semantics Conference (ODBASE)*, volume 4803 of *LNCS*, pages 897–914, 2007.
6. Antonio Jimeno, Ernesto Jimenez-Ruiz, Vivian Lee, Sylvain Gaudan, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics*, 9(Suppl 3):S3, 2008.
7. G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. In *International Conference on Pattern Recognition (ICPR)*, pages 347–349, 1984.

8. Victoria Nebot and Rafael Berlanga. Efficient retrieval of ontology fragments. In *JISBD*, 2008.
9. Hoa Nguyen, Thanh Nguyen, and Juliana Freire. Learning to extract form labels. In *VLDB*, 2008.
10. Sriram Raghavan and Hector García-Molina. Crawling the hidden web. In *VLDB*, pages 129–138, 2001.
11. D Rebholz-Schuhmann, M Arregui, S Gaudan, H Kirsch, and A Jimeno. Text processing through web services: Calling whatizit. *Bioinformatics*, 24(2):296–298, 2008.
12. Steffen Staab and Rudi Studer, editors. *Handbook on Ontologies*. International Handbooks on Information Systems. Springer, 2004.
13. Yuan Yan Tang, Chang De Yan, and C. Y. Suen. Document processing for automatic knowledge acquisition. *IEEE Trans. Know. Data Eng.*, 6(1):3–21, 1994.
14. Toyohide Watanabe, Qin Luo, and Noboru Sugie. Layout recognition of multi-kinds of table-form documents. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(4):432–445, 1995.
15. Zhen Zhang, Bin He, and Kevin Chen-Chuan Chang. Understanding Web query interfaces: best-effort parsing with hidden syntax. In *SIGMOD*, pages 107–118, 2004.

Recuperación de fragmentos útiles de texto a partir de páginas web. Resultados experimentales y perspectivas futuras

Carlos G. Figuerola, José Luis Alonso Berrocal, y Angel F. Zazo Rodríguez

Grupo de Investigación REINA, Universidad de Salamanca
c/ Francisco de Vitoria, 6-16, 37008 Salamanca
{figue, berrocal, afzazo}@usal.es

Resumen. Todos aceptamos que el web es el mayor repositorio de información disponible, y por ello son especialmente importantes las herramientas y las técnicas que permiten acceder a dicha información. Uno de los problemas con los que debemos enfrentarnos es la diversidad de usos posibles; esta diversidad de usos requiere enfoques especializados. En este trabajo se describe uno de los usos de la información que se encuentra en el web, los problemas que plantea la obtención de información útil para esos usos, así como los enfoques adoptados en diversos experimentos realizados tedentes a evaluar las posibilidades de diversas técnicas aplicables.

Palabras clave: recuperación de información, recuperación web, fragmentación de texto, conversión a texto plano

1 Introducción

Todos aceptamos que el web es el mayor repositorio de información disponible, y por ello son especialmente importantes las herramientas y las técnicas que permiten acceder a dicha información. Buscadores de diverso tipo, directorios manuales o automáticos, tecnologías basadas en agentes, *web mining*, son algunas de las técnicas que intentan ayudarnos en la utilización de la información disponible, de una u otra forma, en el web. Uno de los problemas con los que debemos enfrentarnos es la diversidad de usos posibles; esta diversidad de usos requiere enfoques especializados, toda vez que parece que las soluciones genéricas resultan en muchas ocasiones poco satisfactorias.

Desde esta perspectiva, una experiencia interesante es la auspiciada por el Cross Lingual European Forum (CLEF) [1], en el sentido de modelar un determinado uso de información que se encuentra en el web y plantear experimentos tendentes a medir los resultados de aplicar diferentes técnicas. Una de las ventajas de la metodología empleada por CLEF es que el entorno en el que se han de desarrollar los experimentos es homogéneo para todos esos experimentos, y las técnicas de medición y evaluación de resultados están bastante contrastadas.

En concreto, uno de los escenarios planteados es el de un supuesto usuario que requiere información sobre un tema (*topic*) determinado, sobre el cual debe escribir un trabajo o artículo más o menos académico. El usuario no requiere páginas web (o documentos PDF, etc.) sino información útil para sus fines; esta información útil toma la forma de bloques o fragmentos de texto directamente utilizables en la redacción del artículo, en el mejor de los casos susceptibles de ser copiados y pegados en el procesador de texto. En el modelo planteado, el usuario dispone de los resultados de unas cuantas búsquedas en un buscador estándar (Google), así como de unos cuantos documentos considerados como *fuentes conocidas* sobre el tema en cuestión [2]. El escenario completo consta de varios temas o *topics* (alrededor de 60, en la actualidad) en varias lenguas.

El planteamiento tiene el interés adicional de que se trata de una simulación muy cercana a la realidad, en el sentido de que es preciso resolver no sólo la implementación de modelos teóricos, sino la resolución de todos los problemas que aparecen en una situación real: páginas HTML mal formadas, etiquetas maliciosas (*web spam*) [3], multiplicidad de lenguas, errores tipográficos, conversión a texto, detección de lenguas, etc. Muchos de estos aspectos, aunque no tienen el impacto conceptual ni teórico de otros, están lejos de estar resueltos adecuadamente y no son en absoluto triviales, en el sentido de que influyen de forma notable en los resultados que se obtienen en situaciones reales.

Este trabajo trata de mostrar algunos de los enfoques aplicados en la resolución de una tarea como la descrita, de sus limitaciones y de las posibles vías de superar tales limitaciones.

2 Preparación de la colección documental

2.1 Planteamiento general

Nuestra aproximación consiste en considerar, para cada *topic*, el conjunto de documentos recuperados por Google como la colección de documentos con la que trabajar. Puesto que la tarea exige obtener bloques o fragmentos de texto, estos documentos han de fragmentarse o descomponerse en trozos, cada uno de los cuales será considerado como un documento independiente.

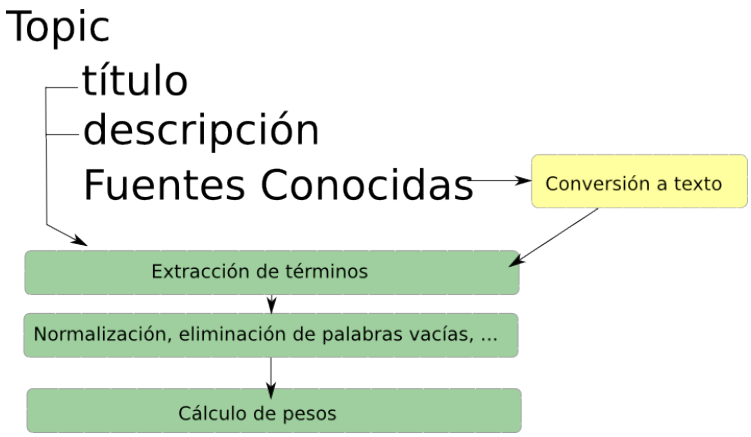


Fig. 1. De *topic* a consulta

Como consulta, para cada *topic*, podemos utilizar la descripción que tenemos para cada uno de ellos. Adicionalmente, dicha consulta puede ser enriquecida con más términos, provenientes de las *fuentes conocidas* para ese *topic*. También podemos usar los anclas disponibles que apuntan a los documentos recuperados por Google.

De esta forma, la tarea puede abordarse como un problema clásico de recuperación, y aplicar en consecuencia, técnicas convencionales.

Así, la colección estará formada por los *snippets* provenientes de los documentos recuperados por Google para cada *topic*. Para cada uno de estos *topics*, se han efectuado una o más búsquedas en Google, y se han tomado para cada una de esas búsquedas los n primeros documentos recuperados. Esto implica un número variable de documentos por *topic*.

Hemos considerado o valorado igual todas las búsquedas en Google para un mismo *topic*. Así que, para cada uno de los documentos recuperados por Google deberíamos obtener el

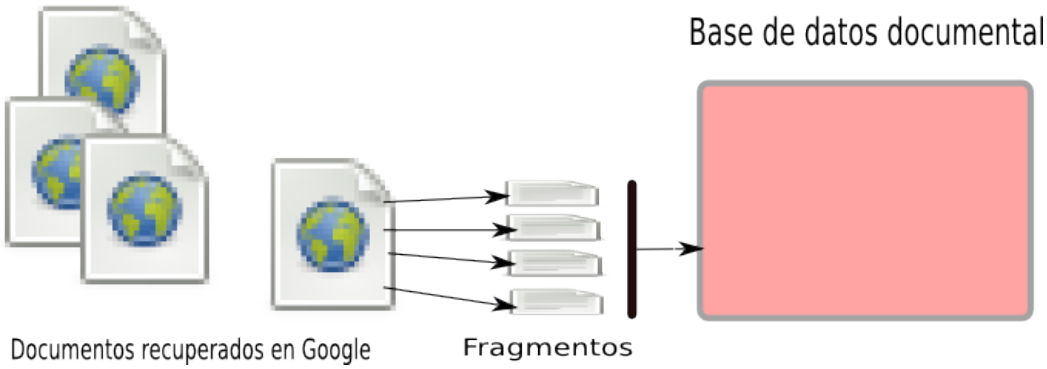


Fig. 2. El espacio de búsqueda

documento original, convertirlo a texto, descomponerlo en fragmentos, obtener los términos de cada fragmento y calcular sus pesos.

2.2 Conversión a texto

En líneas generales la conversión a texto plano puede acometerse mediante herramientas más o menos convencionales. Sin embargo, el uso de codificaciones dispares de caracteres introduce algunas dificultades. Para idiomas que usan caracteres no contenidos en el ASCII estándar, la codificacion y decodificación de tales caracteres es una fuente de problemas. Simplemente la detección del sistema de codificación es, en muchos casos, problemática [4]. Como ejemplo, hemos utilizado el Universal Encoding Library Detector (conocido generalmente como *chardet*), un módulo para Python basado en las librerías para detección de codificaciones utilizadas por Mozilla [5].

2.3 Fracionamiento de los documentos

Varias técnicas pueden ser utilizadas para fragmentar los documentos y obtener pasajes de texto más o menos cortos [6]. En líneas generales, unas están basadas en el tamaño de los fragmentos, el cual, a su vez, puede estimarse en *bytes* (o en caracteres) o en palabras. Otras están orientadas a la separación en frases o párrafos. El primer tipo de técnicas produce, obviamente, fragmentos o trozos más homogéneos en tamaño, pero a menudo carentes de sentido alguno, ya que el punto por el que se parte es ciego. Las otras técnicas tienden a producir fragmentos de muy diferente tamaño. Además, su aplicación no siempre es fácil; en muchos casos la conversión de documentos HTML a texto plano hace desaparecer la separación entre párrafos, las diferencias entre saltos de línea blandos y duros; y también elimina u oculta elementos estructurales, como las tablas.

Un enfoque simple, como la elección de un carácter ortográfico, como el punto, para fragmentar el texto, tiende a producir fragmentos demasiado cortos, y, por tanto, poco útiles para los objetivos de esta tarea. En nuestro caso, hemos adoptado un enfoque mixto. Después de varias pruebas, encontramos que un tamaño adecuado para cada fragmento era alrededor de 1500 caracteres, pero como buscábamos fragmentos que tuvieran sentido informativo, nuestro fragmentador busca el carácter ortográfico elegido (el punto) más cercano a los 1500 caracteres, y parte el texto por ahí.

2.4 Otras operaciones de análisis léxico

Adicionalmente. se efectuaron otras operaciones: conversión a minúsculas, eliminación de acentos, eliminación de palabras vacías (mediante una larga lista de palabras vacías para todos los idiomas contemplados), y aplicación de un sencillo pero eficaz *s-stemmer* [7] .

Cada fragmento obtenido tras estas operaciones fue considerado un documento independiente. De los documentos así obtenidos se extrajeron los términos, los cuales fueron pesados con un esquema de peso ATU (slope=0.2) [8], aplicando el bien conocido modelo vectorial.

3 Formación de las consultas

De alguna manera, nuestro objetivo es resolver la tarea utlizando técnicas de recuperación convencionales, o al menos ya conocidas. A partir de la colección de documentos formada con los fragmentos obtenidos, debemos seleccionar aquéllos que son más útiles para cada *topic*. La clave de nuestro enfoque está en componer consultas que puedan conseguir una selección adecuada. Para componer dichas consultas disponemos de varias fuentes de información; en primer lugar, tenemos *topics* con un título corto y una breve descripción. Además, tenemos, para cada *topic*, unos cuantos documentos denominados *fuentes conocidas*, a texto completo. También tenemos las consultas hechas a Google para obtener información sobre cada *topic*.

De esta manera, podemos utilizar los *topics* (tanto los títulos como las descripciones) como núcleo o base de cada consulta, y enriquecer o aumentar las consultas con términos provenientes de las *fuentes conocidas*. Las *fuentes conocidas* son documentos completos, algunos de ellos muy largos, que pueden contener muchos términos. Podemos preguntarnos si tal vez esto introducirá demasiado ruido en la consulta. Una posible solución es pesar los términos provenientes de las *fuentes conocidas* de una forma diferente a los que provienen del título y la descripción de cada *topic*.

Además, es también posible considerar diferentes estructuras o campos en las *fuentes conocidas* dado que en su mayor parte son páginas HTML (title, body, headings, meta tags, etc.). Experimentos anteriores en ediciones anteriores de CLEF [9] mostraron la importancia para la recuperación de algunos de esos campos, así como el escaso interés de otros. El campo más interesante, en este sentido, es el ancla de los *backlinks* [10]. Sin embargo, dado que tenemos un conjunto muy reducido de documentos, no tenemos muchos *backlinks* con los que trabajar; no obstante, parecen especialmente importantes aquéllos que, desde las *fuentes conocidas* apuntan a alguno de los documentos recuperados por Google.

Así, hemos utilizado en las consultas los términos de títulos y descripciones de los *topics*, más los términos de los anclas mencionadas antes. A esto hemos añadido los términos de las *fuentes conocidas* pero pesados de diferente forma. En ediciones anteriores de WebCLEF hemos trabajado en el uso de diferentes fuentes de información en la recuperación, y en cómo mezclar o fusionar esas fuentes [11]. En esta ocasión hemos elegido modificar los pesos de los términos operando sobre la frecuencia de éstos en cada documento. El esquema de peso elegido también para las consultas es ATU (slope=0.2), razón por la cual el peso es directamente proporcional a la frecuencia del término en el documento; así, hemos establecido un coeficiente por el cual multiplicar dicha frecuencia..

4 Resultados preliminares

Diversas pruebas llevadas a cabo varían en función de este coeficiente: una de ellas mantiene la frecuencia original, por lo que los términos provenientes de las *fuentes conocidas* son pesados igual que los de los *topics*. Otras pesan los términos de las fuentes conocidas reduciendo el peso en diversas medidas; y una prueba adicional no utiliza en absoluto los términos de las *fuentes conocidas*.

Lamentablemente, a la hora de redactar estas líneas carecemos aún de resultados oficiales, acordes con la metodología de evaluación CLEF. Sin embargo, una evaluación oficiosa, basada en una prospección manual de los *snippets* recuperados nos permite avanzar algunos de los problemas que habrá que abordar, así como áreas de trabajo futuro.

Los resultados ofociosos de estas pruebas muestran poca diferencia entre ellas. Parece que usar los términos de las fuentes conocidas es más útil que no usarlos. Pero debemos tener en cuenta que varios *topics* (casi la mitad de ellos) no producen ningún resultado útil. El problema principal parece estar en la naturaleza de los bloques o fragmentos de texto recuperados. Muchos de ellos, aún estando relacionados de alguna forma con el tema o *topic* deseado no contienen información útil para los supuestos del escenario diseñado para las pruebas o experimentos; por ejemplo, muchos de éstos simples referencias bibliográficas, o enlaces a otros documentos. Apuntan a otros documentos tal vez útiles, pero ellos mismos no contienen información directamente utilizable para los fines del supuesto trabajado.

Es importante señalar que, en este escenario concreto, tal vez el esquema de peso elegido no sea el más adecuado. En efecto, en el modo de calcular el peso o importancia de cada término en un documento dado se suele aplicar algún normalizador que permita obviar las diferencias de tamaño (en número de términos) etre los documentos; en nuestro caso ya hemos comentado que aplicamos el *Pivoted document length normalization*, que es aceptado como uno de los más eficaces. Sin embargo, en este escenario concreto puede ser interesante primar los bloques de texto más grandes, siempre dentro de ciertos límites. Ya hemos comentado que el sistema utilizado para fragmentar las páginas o documentos produce bloques de tamaños dispares, aunque con un límite máximo en torno a los 1500 caracteres. Algunos de los bloques producidos acaban siendo tan pequeños que resultan poco útiles y, en general, los bloques que, manteniendo coherencia en su contenido, son de tamaño mayor resultan ser los más interesantes. Cabe preguntarse, por este motivo, si otro esquema de normalización de pesos, o incluso la no normalización podrían mejorar los resultados.

Otro problema importante es la gran cantidad de información duplicada o casi duplicada que podemos encontrar sobre un mismo tema; la dificultad mayor estriba en que no se trata de duplicados exactos. El mismo texto con otros añadidos de otras fuentes, el mismo texto pero en páginas con cabeceras, pies, menúes, barras laterales, etc. diferentes, que producen bloques de texto que no son idénticos considerados como cadenas de caracteres, pero sí duplicados desde el punto de vista de la información contenida. Hemos utilizado el coeficiente de Dice como medida para comparar fragmentos y detectar duplicados aproximados. Comop ejemplo, si consideramos duplicados aproximados los que arrojan un coeficiente superior a 0.7 y aplicamos este umbral a los fragmentos o bloques recuperados para cada topic, encontramos que el 11.08 % son duplicados.

En este sentido, parece que la conversión de documentos HTML a texto plano mediante herramientas convencionales tiene evidentes limitaciones. En efecto, este tipo de documentos presenta en muchos casos una estructura visual [12] que la conversión convencional es incapaz de respetar. Este es el caso de los bloques en que muchas plantillas dividen las páginas web generadas por diferentes aplicaciones; algunos de estos bloques, por ejemplo, tienen sólo interés navegacional y no contienen, obviamente, información relevante para nuestros fines; otros contienen noticias de copyright, de contacto o responsabilidad de la página, o incluso publicidad, u otros elementos poco o nada útiles para nuestro caso: calendarios, votaciones, formularios de logins, etc.

La figura 3 muestra una página que es *fuente conocida* de uno de los *topics*, en la que se han marcado las áreas que contienen información relevante. El ejemplo es siginificativo, porque los artículos de la *wikipedia* son con frecuencia *fuente conocida*. La misma página, otro lado, convertida a texto plano de forma convencional, produce un fichero de 6155 caracteres, de los cuales sólo 1177 (menos del 20 %) corresponden a información relevante.

Una conversión que identificara este tipo de elementos [13] permitiría descartar partes considerables de las páginas, que producen en nuestro sistema bloques de texto no útiles. Algunas pruebas informales dan idea de las dimensiones de este problema: aplicamos un enfoque simple (incluso burdo), filtrando y eliminando fragmentos basándose en una heurística simple: bloques con demasiadas lineas en blanco, con líneas demasiado cortas, con pocas palabras en relación al tamaño del fragmento, etc.. Así, de 639 215 fragmentos obtenidos de los documentos conseguimos eliminar 165 442 (=25.88 %).

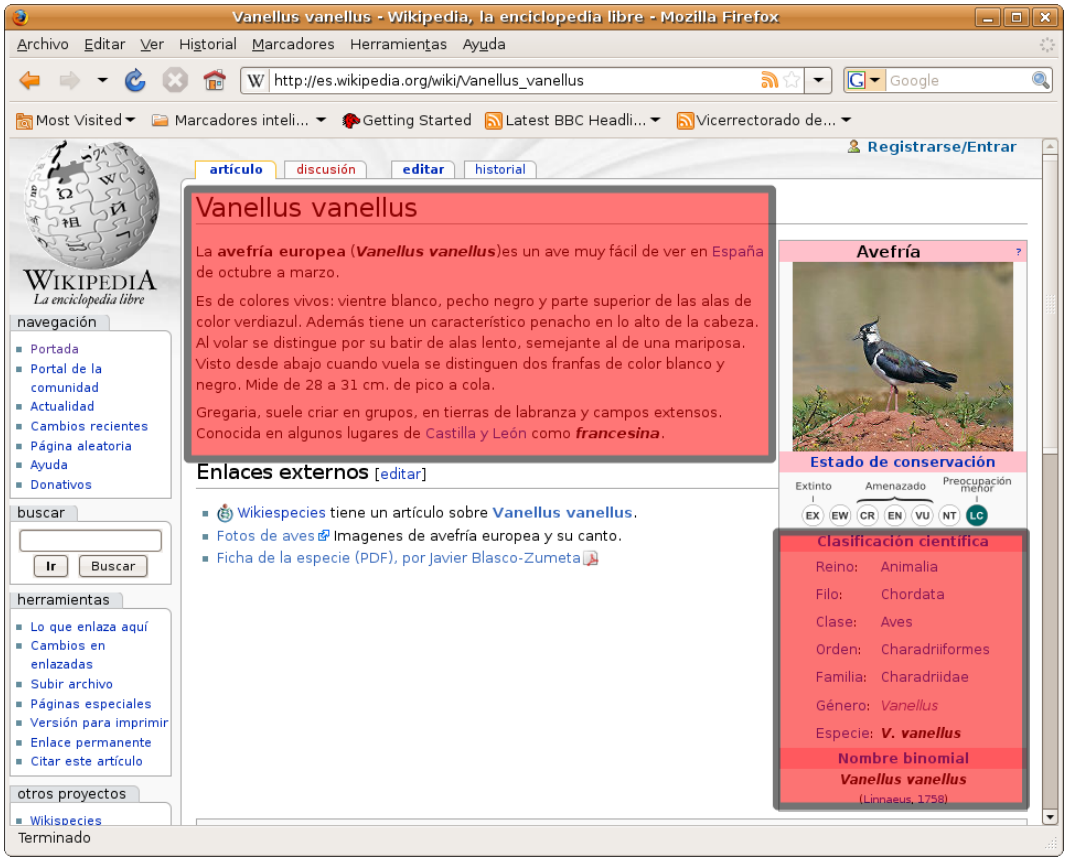


Fig. 3. Sólo las áreas marcadas son relevantes

5 Conclusiones

Hemos descrito nuestra enfoque y nuestros experimentos en la forma de abordar una situación real que se plantea cuando se necesita obtener información del web. Los primeros resultados obtenidos indican que tal vez éstos pudieran mejorarse si pudiéramos aislar en las páginas web aquéllas partes susceptibles de contener información relevante para nuestras necesidades, descartando bloques y áreas de dichas páginas no relevantes. Igualmente, la detección y eliminación de información redundante, pero no idéntica formalmente, podría mejorar de forma notoria los resultados.

Referencias

1. Braschler, M., Peters, C.: Cross-language evaluation forum: Objectives, results, achievements. Information Retrieval **7**(1-2) 7–31
2. Jijkoun, V., de Rijke, M.: Overview of webclef 2007. In Nardi, A., Peters, C., eds.: Working Notes for the CLEF 2007 Workshop
3. Becchetti, L., Castillo, C., Donato, D., Baeza-YATES, R., Leonardi, S.: Link analysis for web spam detection. ACM Trans. Web **2**(1) (2008) 1–42
4. Li, S., Momoi, K.: A composite approach to language/encoding detection. In: 19th International Conference on Unicode. (2001)
5. Pilgrim, M.: Universal encoding detector
6. Allan, J., Wade, C., Bolivar, A.: Retrieval and novelty detection at the sentence level. In: SIGIR '03. (2003) 314–321
7. Figuerola, C.G., Zazo, Á.F., Rodríguez Vázquez de Aldana, E., Alonso Berrocal, J.L.: La recuperación de información en español y la normalización de términos. Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial **8**(22) (2004) 135–145
8. Singhal, A., Buckley, C., Mitra, M.: Pivoted document length normalization. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 18–22, 1996, Zurich, Switzerland (Special Issue of the SIGIR Forum), ACM (1996) 21–29
9. Sigurbjörnsson, B., Kamps, J., Rijke, M.d.: Overview of webclef 2005. [14]
10. Figuerola, C.G., Alonso Berrocal, J.L., Zazo Rodríguez, Á.F., Rodríguez, E.: REINA at the WebCLEF task: Combining evidences and link analysis. [14]
11. Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D., Frieder, O., Goharian, N.: On fusion of effective retrieval strategies in the same information retrieval system. Journal of the American Society for Information Science and Technology (JASIST) **55**(10) (2004) 859–868
12. Yang, Y., Zhang, H.: Html page analysis based on visual cues. In: ICDAR, IEEE Computer Society (2001) 859–864
13. Kang, J., Choi, J.: A preliminary report for an information extraction system based on visual block segmentation. Technical Report TR-IS-2007-1, Hanyang University, Intelligent Systems Laboratory (2007)
14. Peters, C., ed.: Results of the CLEF 2005 Cross-Language System Evaluation Campaign. Working notes for the CLEF 2005 Workshop, 21-23 September, Vienna, Austria. In Peters, C., ed.: Results of the CLEF 2005 Cross-Language System Evaluation Campaign. Working notes for the CLEF 2005 Workshop, 21-23 September, Vienna, Austria. (2005)

Automatic query expansion and word sense disambiguation with long and short queries using WordNet under vector model

Francisco João Pinto*
Department of Computer Science, University of A Coruña,
Campus de Elviña s/n, A Coruña, 15071, Spain
fjoao@udc.es
* Corresponding author

Carme Fernández Pérez-Sanjulián
Department of Galician-Portuguese, French, and Linguistics,University of A Coruña,
Campus da Zapateira s/n, A Coruña, 15071, Spain
carme@udc.es

Abstract. This paper describes the experimentation conducted to test the effectiveness of automatic query expansion and word sense disambiguation (WSD) using short and long query of a topic TREC under vector model. We ran different experiments generating queries under vector model using linguistic information extracted from WordNet. Results show that query expansion with short queries and long queries is not able to improve retrieval performance without the selection of the correct meaning of the words but the results are better using short queries.

Keywords: Information retrieval, Query expansion, Word sense disambiguation, WordNet, Vector space model.

1 Introduction

Query expansion (QE) is the process of reformulating a seed query to improve retrieval performance in information retrieval operations. In the context of web search engines, query expansion involves evaluating a user's input (what words were typed into the search query area, and sometimes other types of data) and expanding the search query to match additional documents. Query expansion involves techniques such as: i) Finding synonyms of words, and searching for the synonyms as well; ii) Finding all the various morphological forms of words by stemming each word in the search query; iii) Fixing spelling errors and automatically searching for the corrected form or suggesting it in the results; iv) Re-weighting the terms in the original query. Query expansion is a technology studied in the field of computer science, particularly within the realm of natural language processing and information retrieval.

Search engines invoke query expansion to increase the quality of user search results. It is assumed that users do not always formulate search queries using the best terms. Best in this case may be because the database does not contain the user entered terms. By stemming a user-entered term, more documents are matched, as the alternate word forms for a user entered term are matched as well, increasing the total recall. This comes at the expense of reducing the precision. By expanding a search query to search for the synonyms of a user entered term, the recall is also increased at the expense of precision. This is due to the nature of the equation of how precision is calculated, in that a larger recall implicitly causes a decrease in precision, given that factors of recall are part of the denominator. It is also inferred that a larger recall negatively impacts overall search result quality, given that many users do not want more results to comb through, regardless of the precision.

The goal of query expansion in this regard is by increasing recall, precision can potentially increase (rather than decrease as mathematically equated), by including in the result set pages which are more relevant (of higher quality), or at least equally relevant. Pages which would not be included in the result set, which have the potential to be more relevant to the user's desired query, are included, and without query expansion would not have, regardless of relevance. At the same time, many of the current commercial search engines use word frequency (tf-idf) to assist in ranking. By ranking the occurrences of both the user entered words and synonyms and alternate morphological forms, documents with a higher density (high frequency and close proximity) tend to migrate higher up in the search results, leading to a higher quality of the search results near the top of the results, despite the larger recall.

This trade-off is one of the defining problems in query expansion, regarding whether it is worthwhile to perform given the questionable effects on precision and recall. Critics state one of the problems is that the dictionaries and thesauri, and the stemming algorithm, are driven by human bias and while this is implicitly handled by the query expansion algorithm, this explicitly affects the results in a non-automated manner (similar to how statisticians can 'lie' with statistics). Other critics point out potential for corporate influence on the dictionaries, promoting advertising of online web pages in the case of web search engines.

The expansion experiments were tested against a subset of **the TREC collection (Text Retrieval Conference (http://www.trec.nist.gov)).** Each initial query in every experiment was generated automatically from the TREC topics. Query words are then selected and expanded using the lexical relations included in WordNet. The selection of the correct meaning of each word was done automatically, because our main interest is to evaluate query expansion and an automatic disambiguation of senses. Thus the effect of expansion can be analyzed with regard to the quality of the disambiguation, which is assumed to be optimum. In addition, we select the words only from the title of the topic. This simulates a common user query with few and generic words. The pioneer work in query expansion using WordNet was made by **Voorhees (1994)**. The results of this work were not good, especially when initial queries are long. In the case of initial short queries, query effectiveness was improved but it was not better than the effectiveness achieved with complete long queries without expansion. In the work of **Mandala et al.(1999)** the relations stored in WordNet are combined with similarity measures based on syntactic dependencies and co-occurrence information. This combination improves the effectiveness of retrieval. The work of **Qiu and Frei (1993)** used an automatically constructed thesaurus and the results were good but the expansion was tested against small document collections. Other successful works used thesaurus adapted with relevance information or were tested against collections in specific domains. We will use only linguistic information and we will test expansion with a long subset of the TREC collection. In these works the retrieval model used is the Vector Space Model (VSM). In this model the queries are represented as vectors. The expansion consists in the simply addition of terms to the vector and found a problem in this direct addition of terms in VSM. Let us consider a query (a, b). If there are three expansion terms a_1, a_2, a_3 related with a and one expansion term b_1 related with b, the expanded query (a, b, a_1, a_2, a_3 ,b_1). In the next section we will examine the vector space model explaining the representation of documents and queries and the model for matching. Next we briefly recall what WordNet is. In section 4 we will present the application developed for formatting queries. Section 5 describes the experiments and their results. The conclusions are presented in the last section.

2 Vector space model

The vector space model assigns non-binary weights to terms in both documents and queries and it provides a frameworks in which partial matching is possible. Documents and queries are

represented as t-dimensional vectors as follows. Documents are represented as vectors $\vec{d}$ = ($w_{1,j}$, $w_{2,j}$, . . ., $w_{t,j}$) and queries are represented as vectors $\vec{q}$ = ($w_{1,q}$, $w_{2,q}$, . . ., $w_{t,q}$) , where each weight $w_{i,j}$ and $w_{i,q}$ is positive and non-binary. The term weights are used to compute the degree of similarity between each document stored in the document base and the query supplied by the user. Documents are stored in decreasing order of similarity and, thus, the vector space model takes into consideration documents which match the query terms only partially. This implies that the ranked answer set is a lot more precise than answer sets supplied by the boolean model. Since documents and queries are vectors in a dimensional space, measure of correlation between vectors can be applied to get a measure of similarity between documents and queries. One of the most widely used measures computes through the cosine of the angle between the two vectors. Formally,

$$sim(d_{j,}q) = \frac{\vec{d}_j \bullet \vec{q}}{|\vec{d}_j| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} W_{i_j} \times W_{i_q}}{\sqrt{\left(\sum_{i=1}^{t} W_{i_j}^2\right)} \times \sqrt{\left(\sum_{i=1}^{t} W_{i_q}^2\right)}} = \cos\alpha$$

Where $\bullet$ stands for the inner product between two vectors. The value of sim (dj, q) varies from 0 to 1 and a document might be retrieval even if it matches the query only partially. Many other matching functions have been designed to compute the degree of similarity between a query vector and a document vector. For a detailed study of them we refer to (**Baeza-Yates, 99**).

3 WordNet

WordNet (**Miller et al., 1990**). is a lexical system manually constructed by a group of people leaded by George Miller at the Cognitive Science Laboratory at Princeton University. WordNet is organized in sets of synonyms (*synsets*) with the words with the same meaning. These synsets have different relations between them. The relation of hypernymy/hyponymy (is-a relation) is the principal relation and creates a hierarchic structure. There are also relations of meronymy/holonymy (part-of relation). In addition, WordNet is divided in four taxonomies by the type of word: nouns, verbs, adjectives and adverbs. We only used the taxonomy of nouns because nouns are the most content-bearing words. Expansion terms will be selected from the correct synsets for each noun in the query.

4 Query formulation

Queries are generated starting from the TREC topics, selecting the expansion terms and fetching lexical information from WordNet. We developed an application that let us to test different options of expansion. In the definition of an experiment we can choose among different options as if it can see the continuation. We consider one topic TREC with following fields T, D and N where T is the title, D is the description and N is the narrative, that if can represent for T, D, N, only considering title T if it gets the following queries: i) Original query: given a query q_1 = (A_0, B_0, C_0), the vector of the query not expanded only with the title, where A_0, B_0, and C_0 are pertaining terms to the related vector, in which A0 and C_0 are nouns. ii)Expanded query: Also let us assume that from the expansion choosing only the nouns that appear in the literal description of the topic using the linguistic resource WordNet we get the following terms: A_1, A_2 and A_3 as we synonyms related with A_0 and C_1, C_2, C_3 and C_4 as we synonyms related with C_0, then the new vector of the

expanded consultation will be in the following way: $q_2 = (A_0, B_0, C_0, A_1, A_2, A_3, C_1, C_2, C_3, C_4)$. iii) Expanded query and election of the correct meanings: Let us assume now that in the query: q_2, if selected the terms A_1, C_1 y C_2 as correct, the query will be in the following way: $q_3 = (A_0, B_0, C_0, A_1, C_1, C_2)$ with this if intends that if in one it query an original term has diverse meanings, assumes that those that are different of the correct one do not influence in the effectiveness of the information retrieval. This process is analogous with any syntactic category and with any field of topic.

The processing of the queries after expansion and election of the correct meanings, are made using the vector space model. In this model, the documents of the collection and the queries are represented as vectors of terms inside of a vector space. A gotten time expanded query and the query with the election of the meaning correct of the words, the selected field is indexed using the routines standards of the information retrieval System Lemur , computes one ranking of documents for each query. For each query the documents a are stored in decreasing order of similarity. The value of similarity of a document d with the query q is given using the inner product between the two vectors, and later one time gotten the results of the query expanded and of the query expanded and the election of the correct meanings, on the different contemplated experiments will be proceeded to the evaluation and the comparison between them with the ones of the query without expanding.

5 Experiments and results

We used an information retrieval system called lemur **(http://www.lemurproject.org)** and a subset of documents from the reference TREC-8 text collection for evaluating the effects of query expansion in our experiments. More precisely, in all our experiments those topics numbered between 401 and 450 were taken, so that a collection composed of fifty test queries was used in each experiment. The Small Web (WT2g) was the collection of documents used from TREC-8. WT2g contains around 250,000 documents. In addition, the measurements of recall and precision were used to evaluate the results obtained in the retrieval process. The main goal of our experiments was to prove if the expansion of the original queries using all fields of topics and the selection of the correct meaning lead to some improvements in the effectiveness obtained during the information retrieval process.

5.1 Original query

In this section we focus in the effects of using the original query in an information retrieval system. In the Table 1 , we present the values of precision obtained by the original query using short and long query. In these experiments, the query expansion and word sense disambiguation (WSD) was not applied. As expected, the best results were obtained when the long query were performed. To sum up, the average precision (non-interpolated) value can be obtained. In the practice the average precision (non-interpolated) value using the title is 0, 0839, average precision (non-interpolated) value using the description is 0, 1495, average precision (non-interpolated) value using narrative is 0, 2103, and the average precision (non-interpolated) value using the full query is 0, 2431.

Short and long queries	Title	Description	Narrative	Full
Average precision(non-interpolated) value	0,0839	0, 1495	0, 2103	0,2431
Set total number of retrieved docs	1112	1404	1578	1677

Tabla 1: Results of original query

5.2 Query expansion without disambiguation

In this section we focus in the effects of using query expansion in an information retrieval system. In the Table 2, we present the values of precision obtained by the query expansion and those obtained by applying query expansion of nouns. In these experiments, word sense disambiguation (WSD) was not applied. As expected, the best results were obtained when the short query was performed, since it leads to higher values of precision. To sum up, the average precision (non-interpolated) value can be obtained. In the practice the average precision (non-interpolated) value using the title is 0, 1655, average precision (non-interpolated) value using the description is 0, 0964, average precision (non-interpolated) value using narrative is 0, 0628, and the average precision (non-interpolated) value using the full query is 0, 0445.

Short and long queries	Title	Description	Narrative	Full
Average precision(non-interpolated) value	0,1655	0, 0964	0, 0628	0,0445
Set total number of retrieved docs	1696	1507	1455	1688

Tabla 2: Results of query expansion

5.3 Query expansion with desambiguation

Our dictionary-based disambiguation method makes use of WordNet. Basically, it is based on **(Pinto, 2007a; 2007b)** and contains the following steps: *i)* Extraction from WordNet of the sets of synonyms for all the words being disambiguated, *ii)* determining the coincidence between the context of the words being disambiguated and the sets of synonyms, and *iii)* selection of the correct meaning for the words in a text, where the context is given in an automatic form. In the Table 3, we present the results obtained when query expansion is combined with WSD. By comparing these results against those presented in the Table 2, it can be seen that the use of word sense disambiguation leads to better effectiveness values for all the syntactic categories over which query expansion was performed. More precisely, by including the disambiguation of nouns into the query expansion process: i)The average precision (non-interpolated) obtained using the title is around 0,2030, ii) The average precision (non-interpolated) obtained using the description is around 0,1672, iii) The average precision (non-interpolated) obtained using the narrative is

around 0,1577 and The average precision (non-interpolated) obtained using the full query is around 0,1345.

Short and long queries	Title	Description	Narrative	Full
Average precision(non-interpolated) value	0, 2030	0, 1672	0, 1577	0,1345
Set total number of retrieved docs	1495	1371	1344	1298

Tabla 2: Results of query expansion and WSD

Conclusions

Applying to the retrieval to the original queries the results they are better using the long queries because they have more terms. In this case is important to improve the results of the short queries adding similar terms (query expansion), but some times the similar terms that are added can not be wished, for that reason it is important to select the meaning correct (WSD). Therefore query expansion is important in short queries because when adding more terms similar to long queries only bring about noise. The word sense disambiguation is a non trivial task within an information retrieval system. Many systems try to select the most appropriate sense for a polysemous word using statistics and/or automatic learning. Despite such systems, our proposal uses dictionary-based disambiguation. It uses disambiguation during the query expansion process, yielding interesting improvements in the effectiveness obtained. In our experiments (using vector space model), the new approach yielded improved effectiveness when disambiguation was applied using WordNet. In practice, the best results where obtained when disambiguation was applied using short query. The results obtained regarding both the expanded queries and the word sense disambiguation permit us to extract some interesting conclusions: By using different expansion techniques and WSD the results had slightly improved with respect to the short queries. Experiments show that by using word sense disambiguation more advantages from the query expansion with WordNet can be exploited. In practice, our results in a long text collection support the results obtained in previous research works that reported good effectiveness values when query expansion was used in conjunction with word sense disambiguation over small text collections.

Reference

Baeza-Yates, R.A. and Ribeiro-Neto, B. (1999) 'Modern Information Retrieval'. Addison Wesley.

Mandala, R., Tokunaga, T., and Tanaka, H. (1999). 'Combining multiple evidence from different types of thesaurus', *Proceedings of the 22th ACM-SIGIR Conference*, pp.191-197.

Miller, A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J (1990). 'Introduction to WordNet: An on- line lexical database'. *International Journal of Lexicography*. Vol.3, No.4, pp.235–312.

Pinto, F. J. (2007a). 'Uso del Recurso Linguístico WordNet en la Expansión de Consultas con un Modelo de Usuario de Recuperación de Información'. *Proceedings of the 2nd CEDI*. In Spanish.

Pinto, F. J. (2007b). 'Evaluación del Sistema de Recuperación de Información Lemur con Distintos Tipos de Indexación Automática'. *Proceedings of the 12th Conferencia de la AEPIA (Zoco'07/CAEPIA)*. In Spanish.

Qiu, Y. and Frei, H. (1993). 'Concept-based query expansion', *Proceedings of the 16th ACM-SIGIR Conference*, pp. 160–169.

Voorhees, E. M. (1994). 'Query Expansion using Lexical-Semantic Relations', *Proceedings of the 17th ACM-SIGIR Conference*, pp. 61–69.

Comparación de fuentes de conocimiento web para su empleo en el matching de ontologías

Jorge Martinez-Gil y José F. Aldana-Montes

Departamento de Lenguajes y Ciencias de la Computación, Universidad de Málaga
Boulevard Louis Pasteur s/n 29071 Málaga, España
{jorgemar, jfam}@lcc.uma.es
http://khaos.uma.es

Resumen. Actualmente existen muchas técnicas y herramientas que tratan de solucionar el problema del alineamiento de ontologías. No obstante, la compleja naturaleza del problema, en la que intervienen factores dependientes del contexto, de los datos y de los propios usuarios que desean integrar conocimiento, hace que las soluciones propuestas no sean completamente satisfactorias. En este sentido, ha aparecido recientemente la distancia de similitud de Google. Una métrica cuyo objetivo es hacer estimaciones acerca de la similitud de dos términos en base a su aparición en los resultados de búsqueda del conocido buscador. Nuestro trabajo consiste en un experimento sistemático que extiende dicha métrica para que pueda emplearse en otros motores de búsqueda.

Palabras clave: matching de ontologías, integración de datos, conocimiento web, web semántica.

1. Introducción

El problema del alineamiento de ontologías [14] es, quizás, uno de los más interesantes y trascendentes a los que la comunidad científica perteneciente al campo de Web Semántica ha de hacer frente. Su resolución es clave para permitir a las organizaciones modelar la información con la que trabajan sin tener que ajustarse a las normas de un estándar específico. De hecho, hay al menos dos buenas razones por las que creemos que la mayoría de las organizaciones no se han decidido en el pasado a trabajar con modelos estándar para gestionar la información que manejan. En primer lugar, es muy difícil o requiere un tremendo esfuerzo alcanzar un consenso acerca de un modelo de datos común. En segundo lugar, estos estándares no suelen ajustarse de manera perfecta a las necesidades específicas de todos y cada uno de los participantes.

Aunque el alineamiento de ontologías[1] es, quizás, la forma más valiosa de solventar los problemas de heterogeneidad semántica entre sistemas, incluso existen multitud de técnicas para alinear ontologías de manera muy precisa que se aplican en el matching de Servicios Web Semánticos [6] y de Recuperación basada en la similitud [31]. Sin embargo, la experiencia nos dice que la naturaleza no determinista del problema hace que para estas técnicas sea muy complicado responder de manera satisfactoria en todos los dominios de la forma en que todos los usuarios esperan que lo hagan. La Figura 1 nos muestra un alineamiento que es válido en España, pero no en todo los países, por ejemplo.

Nuestra opinión la comparten otros investigadores que también han experimentado este problema. Por ejemplo; *encontrar buenas funciones de similitud es una tarea dependiente de*

[1] Los términos alineamiento de ontologías y matching de ontologías suelen confundirse con frecuencia. En este artículo, llamaremos matching a la tarea de descubrimiento de correspondencias semánticas entre ontologías y alineamiento al resultado de la tarea de matching

los datos, del contexto y en ocasiones de los usuarios, y necesita, por tanto, reconsiderarse cada vez que se obtienen nuevos datos o nuevas tareas o *tratan con lenguaje natural a menudo introduce una tasa de error significativa* [21].

Como consecuencia, se han desarrollado nuevos mecanismos que oscilan desde medidas de similitud personalizadas [22,41] hasta técnicas de composición de algoritmos que dan lugar a matchers híbridos [9,16], pasando por sistemas de meta-matching [11,24]. Aún así, los resultados no son totalmente satisfactorios [15]. Nuestra hipótesis propone el empleo de conocimiento web como apoyo para el alineamiento. El conocimiento web ya se ha aplicado con éxito en la resolución de otro tipo de problemas de naturaleza computacional, como por ejemplo la resolución de crucigramas de manera automática [13].

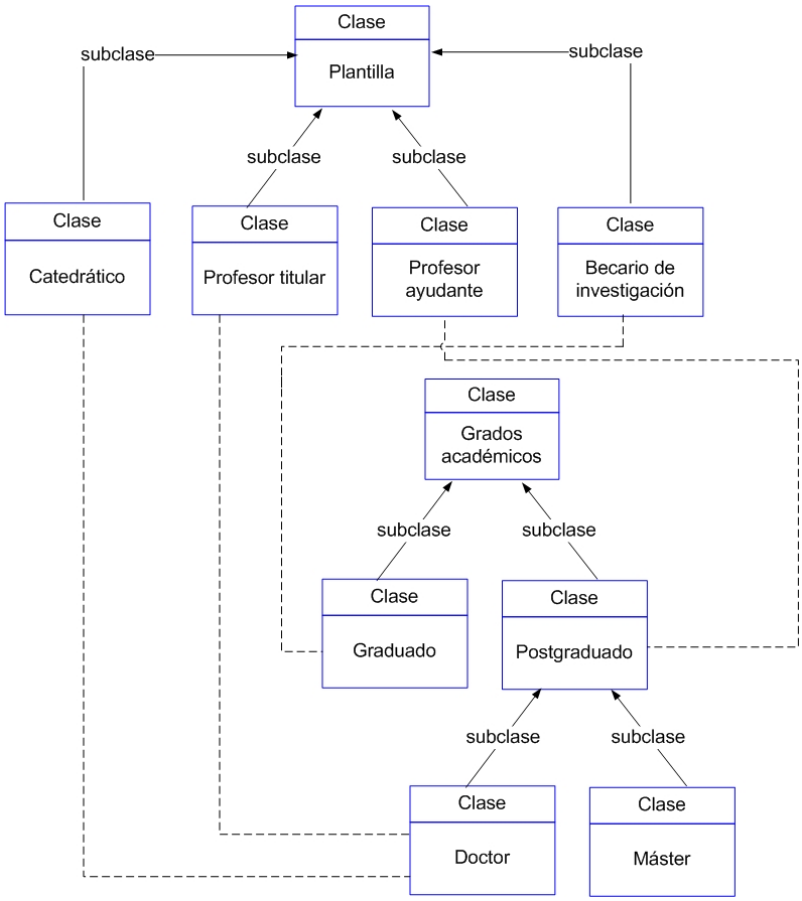


Fig. 1. Ejemplo de un alineamiento dependiente del usuario

De hecho, estamos especialmente interesados en tres características de la World Wide Web (WWW):

1. Es la base de datos más grande del mundo. Y posiblemente constituye la fuente más valiosa de conocimiento de propósito general.
2. La información (el conocimiento) que almacena está en lenguaje natural y, por tanto, puede ayudar a resolver problemas relacionados con el Procesamiento del Lenguaje Natural (PLN).

3. Ofrece mecanismos simples capaces de separar la información relevante de la información no relevante.

En este sentido, creemos que la contribución más notable de este trabajo es la identificación de las mejores fuentes de conocimiento web para solucionar el problemas del alineamiento de ontologías de manera precisa. De hecho, en [7], los autores afirman: *Presentamos una nueva teoría de similitud entre palabras y frases que se basa en la distancia de información y en la complejidad de Kolmogorov. Para demostrarlo, vamos a usar la World Wide Web (WWW) como base de datos y Google como motor de búsqueda. El método también es aplicable a otros motores de búsqueda.* Ahí es precisamente donde entra en juego nuestra propuesta.

Bajo ninguna circunstancia este trabajo puede considerarse como una demostración de que un motor de búsqueda es mejor que los demás o que la información que proporciona es más precisa. Tan sólo, que las características a la hora de procesar las consultas y la variedad de los contenidos indexados lo hacen más apropiado para dar soporte al alineamiento de ontologías pertenecientes a los dominios estudiados.

El resto de este artículo se estructura de la siguiente forma: La siguiente sección define el problema del descubrimiento de correspondencias semánticas entre ontologías que pertenecen al mismo dominio pero que se han desarrollado de manera independiente. La tercera sección muestra las definiciones preliminares que son necesarias para seguir nuestra propuesta. La sección Diseño del experimento comenta todo lo relativo a la preparación de los experimentos que determinarán cuál es la fuente de conocimiento web más apropiada. La sección de Evaluación empírica muestra los resultados numéricos obtenidos. La sección de discusión discute los resultados del apartado anterior. Y por último, Conclusiones y trabajo futuro trata las conclusiones que pueden extraerse de nuestra propuesta y comenta las futuras líneas de trabajo.

2. Definición del problema

El proceso de alinear ontologías puede expresarse como una función donde dado un par de ontologías, un alineamiento de entrada opcional, un conjunto de parámetros de configuración y un conjunto de recursos de apoyo, se devuelve un alineamiento. El resultado devuelto por dicha función o alineamiento, es un conjunto de mappings. Un mapping es una tupla compuesta por un identificador único del mapping, las entidades que pertenecen a cada una de las respectivas ontologías, la relación de correspondencia semántica (igualdad, generalización, especialización, etc.) entre las entidades y por un número real comprendido entre el intervalo 0 y 1 que representa la probabilidad matemática de que la relación anteriormente descrita sea cierta. Las entidades que se relacionan pueden ser conceptos, propiedades de objeto, propiedades de tipo, reglas, instancias e incluso axiomas pertenecientes a las ontologías que se desean alinear.

Por otra parte, y de acuerdo a la literatura, podemos agrupar los subproblemas relacionados con el matching de ontologías en ocho grandes categorías:

1. Cómo obtener alineamientos de gran calidad de manera automática.
2. Cómo obtener alineamientos de ontologías en el menor tiempo posible.
3. Cómo identificar las diferencias entre las estrategias de matching y determinar cuál se ajusta mejor a la resolución de un problema concreto.
4. Cómo alinear ontologías muy grandes.
5. Cómo interactuar con el usuario durante el proceso de matching.
6. Cómo configurar los parámetros de las herramientas de matching de manera automática y precisa.

7. Cómo explicar a los usuarios el resultado generado.
8. Cómo visualizar los resultados de un alineamiento.

Nuestro trabajo se centra en el primer punto. Existen varias propuestas destacadas en este campo [4,8,10,20,33]. La mayoría de ellas son combinaciones de los que podríamos denominar técnicas básicas de matching. Entre estas técnicas podemos destacar las siguientes:

2.1. Técnicas estadísticas a nivel de elemento

Son técnicas que sólo se fijan en las etiquetas que identifican un elemento:

- **Normalización de cadenas**. Esta técnica consiste en métodos que eliminan palabras o símbolos innecesarios. Además, intentan detectar plurales, prefijos y sufijos no relevantes, y en general, técnicas avanzadas para el procesamiento del lenguaje natural [36].
- **Medidas de similitud textual**. Las medidas de similitud textual son métodos que permiten identificar elementos similares en cadenas de texto. Pueden usarse, por ejemplo, para identificar el grado de similitud que tienen las etiquetas que identifican a dos conceptos en un par de ontologías. Entre estas técnicas destacan las distancias de edición, donde se tiene en cuenta el número mínimo de operaciones requeridas para transformar la cadena de caracteres destino en la cadena fuente. Se entiende por operación, bien una inserción, eliminación o la substitución de un carácter [23].
- **Métodos lingüísticos**. Que consiste en el empleo de recursos de carácter lingüístico como diccionarios, lemarios, o tesauros para detectar posibles coincidencias. Uno de los métodos lingüísticos más populares consiste en el uso de WordNet para identificar algunos tipos de relaciones entre las entidades [39].

2.2. Técnicas estadísticas a nivel de estructura

Son técnicas que tienen en cuenta la posición de los elementos dentro de la ontología:

- **Análisis de herencia**. Esto tipo de métodos tienen en cuenta la herencia entre conceptos para identificar relaciones. El método más popular es el análisis 'es-un' que intenta identificar relaciones de subsumpción entre elementos.
- **Análisis de datos**. Este tipo de técnicas se basan en la regla: Si dos conceptos tienen las mismas instancias, entonces probablemente serán el mismo elemento. Algunas veces es posible identificar el significado de una entidad de un nivel superior con tan sólo mirar las entidades de sus niveles inferiores. Por ejemplo, si las instancias de un concepto contienen la cadena *años*, probablemente se trata de un atributo para la edad.
- **Correspondencia de grafos**. Consiste en la identificación de estructura de grafos similares. Estos métodos suelen usar algoritmos clásicos de comparación de grafos. En la mayoría de las ocasiones, estos algoritmos procesan los caminos, los hijos, las fuentes, los sumideros, etc. del grafo.
- **Análisis estadístico**. Consiste en la extracción de palabras clave y descripciones textuales para la detección del significado de una entidad en relación con las otras. Los autores trabajaron en el pasado en esta línea [27].
- **Análisis taxonómico**. Intenta identificar conceptos similares en base a sus vecindades. La idea principal que subyace en este tipo de métodos es que dos entidades que pertenecen a ontologías tienen un cierto grado de probabilidad de representar el mismo concepto siempre que compartan vecinos idénticos.

2.3. Técnicas a nivel semántico

Son técnicas formales, más precisas pero también más ineficientes que las estadísticas [5]. Entre ellas destaca especialmente:

- **Análisis semántico**. Según [14], los algoritmos de análisis semántico manejan las entradas en base a su interpretación semántica. Suponiendo que dos entidades son la misma, si comparten su misma interpretación semántica. Son métodos deductivos. Las aproximaciones más destacables se basan en técnicas de lógica de descripciones.

La mayoría de estas estrategias han probado su efectividad a la hora de contrastarlas con benchmarks sintéticos como el ofrecido por la Ontology Alignment Evaluation Initiative (OAEI) [29]. No obstante, cuando trabajan con ontologías reales, sus resultados son peores [32]. Nosotros proponemos utilizar un tipo de recurso lingüístico, aún poco estudiado en el dominio del matching de ontologías, llamado conocimiento web (web knowledge en inglés). El conocimiento web no es más que el resultado de minar información de la Web con la ayuda de motores de búsqueda, para de esta forma dar soporte al proceso del alineamiento de ontologías.

Varios autores han usado conocimiento web en sus respectivos trabajos, o han usado una generalización: conocimiento background [17,37,38]. El conocimiento background involucra a todos los tipos de fuentes de conocimiento para extraer información: diccionarios, tesauros, colecciones de documentos, motores de búsquedas, etc. Por lo que el conocimiento web suele considerarse un subtipo más específico.

La aproximación clásica a este problema se ha abordado en la literatura con el empleo de la herramienta WordNet. En este sentido son destacables las propuestas presentadas en [30].

Por otra parte, estos trabajos presentan metodologías y prototipos para la validación de estas metodologías, pero no ofrecen estadísticas reales de comparación entre ellos. Nuestro trabajo puede verse como una extensión de [7,18], donde se ofrecen varias fórmulas y mecanismos para beneficiarse del conocimiento extraíble del Google. Nuestro trabajo sigue esa línea, pero en lugar de centrarse en aspectos teóricos, trata de fijarse en una parte más práctica, puesto que ofrece un análisis estadístico que no se limita a un solo motor de búsqueda.

3. Preliminares técnicos

En esta sección vamos a detallar algunas definiciones que son necesarias para comprender nuestra propuesta:

Definición 1 (Medida de similitud). *Un medida de similitud sm es una función* $sm : \mu_1 \times \mu_2 \mapsto R$ *que asocia la similitud de dos mappings de entrada* μ_1 *y* μ_2 *a un valor* $sc \in R$ *en el rango [0, 1].*

Un valor de 0 indica una desigualdad total y un 1 una similitud total entre los dos mappings μ_1 y μ_2.

Definición 2 (Matching de ontologías). *Un matching de ontologías m es una función* $m : O_1 \times O_2 \stackrel{sm}{\rightarrow} A$ *que asocia dos ontologías de entrada* O_1 *y* O_2 *a un alineamiento A usando un medida de similitud.*

Definición 3 (Alineamiento). *Un alineamiento a es una tupla* (t, MD)*. t es un conjunto de tuplas de la forma* (id, e, e', n, R)*. Donde id es un identificador único, e y* e' *son entidades*

que pertenecen a dos ontologías distintas, R es la relación de correspondencia entre estas entidades y n es un número real entre 0 y 1 que representa la probabilidad matemática de que R sea cierta. Las entidades que pueden están relacionadas pueden ser conceptos, roles, reglas e incluso axiomas de la ontología. Por otra parte, MD son metadatos generados durante la etapa de matching (tiempo consumido, fecha de la ejecución, etc.).

Definición 4 (Hit). *Hit es cada uno de los elementos encontrados por un motor de búsqueda que coincide con las condiciones de búsqueda especificadas. Más formalmente, podemos definir un hit como la función hit* : $\vartheta \mapsto N$ *que asocia un número natural a una palabra (o conjunto de palabras) estimando así su popularidad en términos absolutos en el subconjunto de Internet indexado por el buscador.*

Un valor de 0 indica que la palabra (o conjunto de palabras) no es popular. Y cuanto mayor sea dicho valor, mayor será su popularidad según el buscador.

4. Diseño del experimento

Para diseñar nuestro experimento, vamos a usar el siguiente método para medir la similitud semántica de dos términos. Sean $c1$ y $c2$ conceptos que pertenecen a dos ontologías diferentes, sea $f(c1, c2)$ una función bidimensional de $f : S \times S \mapsto \aleph$, donde S es el conjunto de todas las cadenas de caracteres y sea : un operador de concatenación de términos:

$$f(c1, c2) = \frac{hit(c1 : blankspace : c2) + hit(c2 : blankspace : c1)}{\frac{hit(c1) \times hit(c2)}{\chi}}$$

Podemos definir la función similitud tal como:

$$similitud(c1, c2) = \begin{cases} f(c1, c2) \; si \; 0 \leq x \leq 1 \\ 1 \qquad\quad si \quad x > 1 \end{cases}$$

Donde χ es un número entero que permite al resultado situarse en un determinado rango. Nosotros usaremos el valor 10000, obtenido de un estudio preeliminar, de modo que el rango de los valores devueltos por la función esté comprendido aproximadamente entre cero y uno.

La medida de similitud nos da una idea del número de veces que dos conceptos aparecen por separado en comparación con el número de veces que aparecen a la vez en el mismo recurso (página, documento, etc...) de Internet.

Hemos escogido, los siguientes buscadores de entre los más populares reflejados en el ranking de Alexa[1]: Google [19], Yahoo [40], Lycos [26], Altavista [2], MSN [28] y Ask [3].

Algunas de las compañías propietarias de estos buscadores no permiten que se ejecuten muchas consultas, esto está considerado como un servicio de minado por el que hay que pagar, por lo que hemos diseñado un pequeño experimento inicial que requiera un número de consultas que no supere los máximos establecidos por persona y día: Para ello vamos a alinear dos ontologías de tamaño pequeño-medio (sobre unos 50 conceptos aproximadamente) que modelan algunos aspectos de la realidad de Rusia [34] y [35]. Hemos escogido estas dos ontologías porque varias herramientas de matching las han usado en el pasado para presentar sus resultados.

Aunque podríamos lanzar una tarea de alineamiento sobre todas las entidades (conceptos, propiedades de objeto, propiedades de tipo de datos e instancias) de las ontologías, vamos a hacerlo sólo sobre los conceptos, con objeto de no rebasar el límite de consultas al que hacemos mención. La comparación se hace siguiendo un esquema de todos con todos, es decir, por cada concepto de la ontología origen se hace una comparación con todos los

conceptos de la ontología destino, se considera el par con una probabilidad de ser cierto más grande y si supera un determinado umbral, se ofrece en el alineamiento final. De esta forma, el número total de consultas a efectuar

$$n^{o}\ total\ consultas\ = m \times\ n$$

Donde m y n son el número de conceptos pertenecientes a la ontologías origen y destino respectivamente.

5. Evaluación empírica

A continuación vamos a mostrar los resultados fruto del experimento efectuado. La Tabla 1 muestra los resultados con los pares de correspondencias semánticas más relevantes que se han obtenido:

Russia1	Russia2	Google	Yahoo	Lycos	Altavista	MSN	Ask
food	food	1.00	0.00	0.01	1.00	0.01	0.02
drink	drink	1.00	0.01	0.30	1.00	0.06	0.04
traveller	normal_traveller	0.00	0.00	0.00	0.00	0.00	0.00
health_risk	disease_type	0.00	0.00	0.00	0.00	0.00	0.00
time_unit	time_unit	0.00	0.00	0.00	1.00	0.00	0.00
document	document	1.00	0.00	0.01	1.00	0.01	0.02
approval	certificate	0.84	0.20	0.00	1.00	0.00	0.00
payment	means_of_payment	0.00	0.45	0.00	1.00	0.00	0.00
monetary_unit	currency	0.00	0.42	0.00	1.00	0.00	0.00
unit	unit	1.00	0.00	0.01	1.00	0.03	0.03
adventure	sport	1.00	0.01	0.03	1.00	0.04	0.40
building	public_building	0.40	0.11	0.00	1.00	0.00	0.00
flight	air_travel	0.80	0.15	0.03	1.00	0.02	0.00
river_transfer	cruise	0.00	0.12	0.00	1.00	0.00	0.00
railway	train_travel	0.00	0.98	0.00	1.00	0.00	0.00
political_area	political_region	0.00	0.40	0.00	1.00	0.00	0.00

Tabla 1. Resultados obtenidos de los diferentes motores de búsqueda

Como puede observarse, los resultados obtenidos para Altavista son casi totalmente opuestos a los ofrecidos por MSN. Obviamente, los resultados de Altavista son más correctos.

Por su parte, la Tabla 2 muestra una comparación de las correspondencias semánticas obtenidas, donde Min. es el mínimo de todos los valores devueltos por los buscadores estudiados y Max. es el máximo. Incluimos estas dos columnas para que se pueda apreciar claramente el rango entre el que oscilan los resultados devueltos por los buscadores. FOAM [12] y RiMOM[25] son herramientas de matching que han ofrecido muy buenos resultados en el concurso Ontology Alignment Contest [32]. FOAM es un framework de alineamiento basado en heurísticas que ha usado las ontologías de Rusia para publicar sus resultados. Por su parte RiMOM [25] usa una composición de algoritmos básicos de matching y fue la herramienta que arrojó mejores resultados en el citado concurso.

Además, es importante remarcar que este experimento fue realizado en Mayo de 2008. Porque la información indexada por los buscadores no es estática.

Russia1	Russia2	Min.	Max.	FOAM	RiMOM
food	food	0.00	1.00	1.00	0.50
drink	drink	0.01	1.00	1.00	0.71
traveller	normal_traveller	0.00	0.00	0.00	0.00
health_risk	disease_type	0.00	0.00	0.00	0.17
time_unit	time_unit	0.00	1.00	1.00	1.00
document	document	0.00	1.00	1.00	0.99
approval	certificate	0.00	1.00	0.00	0.21
payment	means_of_payment	0.00	1.00	0.00	0.37
monetary_unit	currency	0.00	1.00	0.00	0.00
unit	unit	0.00	1.00	1.00	1.00
adventure	sport	0.01	1.00	0.00	0.01
building	public_building	0.00	1.00	0.80	0.60
flight	air_travel	0.00	1.00	0.00	0.62
river_transfer	cruise	0.00	1.00	0.00	0.21
railway	train_travel	0.00	1.00	0.00	0.54
political_area	political_region	0.00	1.00	0.00	0.40

Tabla 2. Comparación de los mappings obtenidos

6. Discusión

Los resultados que hemos obtenidos no han sido suficientemente grandes ni pertenecen a dominios lo suficientemente heterogéneos como para ser definitivos, sin embargo, pueden darnos una idea acerca del comportamiento de los diferentes motores de búsqueda a la hora de alinear ontologías. De hecho, sí podemos destacar dos características que llaman poderosamente la atención sobre el conjunto de datos obtenidos:

1. Existe una gran disparidad entre los resultados obtenidos por los motores de búsqueda que se han tenido cuenta. Sería especialmente interesante saber por qué.
2. Aunque la medida de similitud fue diseñada para funcionar con Google, parece funcionar mejor con Altavista.

Con respecto a la primera, hemos de fijarnos en cómo tratan los buscadores las palabras idénticas, las palabras con guiones, los antónimos y los sinónimos. Podemos observar unas oscilaciones muy amplias, en muchos casos totalmente opuestas. Esto demuestra , de manera preliminar, nuestra hipótesis inicial de que hay fuentes de conocimiento web que son más apropiadas que otras, al menos, para el dominio de las ontologías sobre las que se ha realizado el estudio.

Con respecto a la segunda característica, por qué Altavista ofrece unos mejores resultados que Google, creemos que se debe a que a pesar de que Google responde satisfactoriamente a la hora de comparar sinónimos, fracasa a la hora de comparar palabras separadas por un guión (_). Altavista indexa actualmente muchos menos contenidos que Google, pero el tratamiento de las consultas y/o la indexación de contenido relevante para el dominio de Rusia, hacen que pueda ofrecer mejores resultados.

7. Conclusiones y trabajo futuro

Este trabajo presenta un experimento para alinear ontologías usando extracción de conocimiento web mediante el empleo de la Distancia de Similitud de Google [7] en varios buscadores elegidos entre el grupo de los más usados por los usuarios de Internet.

De nuestro trabajo podemos extraer que:

1. Los buscadores web pueden considerarse como fuentes válidas de conocimiento que proporcionen soporte a la tarea del alineamiento de ontologías. Sobre todo, en los casos, donde se renuncia a, o no es posible, hacer uso de información de naturaleza estructural o semántica.
2. Existe una gran disparidad en los resultados arrojados por los buscadores estudiados, debido fundamentalmente al trato que cada uno de ellos proporciona a los términos iguales, con guiones, compuestos, etc.
3. Aunque intuitivamente y a priori Google parece la fuente de conocimiento con una cantidad de contenidos indexados más amplia y de mayor calidad, el trato que proporciona a las consultas de términos con guiones, hace que no sea el buscador más apropiado para alinear los conceptos de las ontologías estudiadas.

De manera intuitiva, podemos añadir una conclusion más; y es que el empleo de conocimiento extraído de los motores de búsqueda es muy útil para la comparación semántica de instancias. Es decir, para el caso, en el que sólo disponemos de dos etiquetas que comparar, sin ninguna otra información adicional sobre su estructura o su vecindad.

Como trabajo futuro, proponemos una comparativa entre el conocimiento propiciado por WordNet y el proporcionado por las fuentes web. Además, es necesario extender el experimento para trabajar con conjuntos de datos más grandes y, si es posible, reconocidos por algún estándar. Aunque ello depende en gran medida, de que los buscadores permitan realizar un mayor número de consultas. Por otra parte, el desarrollo de una herramienta que permita automatizar todo el proceso, desde la selección de las ontologías hasta la determinación de las mejores fuentes, está ya en marcha. El objetivo es que se puedan evaluar de manera automática las fuentes web conforme a benchmarks ampliamente aceptados.

Agradecimientos

Este trabajo ha sido financiado con fondos del Ministerio de Educación y Ciencia a través del proyecto ICARO (TIN2005-09098-C05-01) y de la Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía con cargo al proyecto P07-TIC-02978.

Referencias

1. Alexa Web Information Company. http://www.alexa.com. Última visita: 23-mayo-2008.
2. Altavista. http://www.altavista.com. Última visita: 23-mayo-2008.
3. Ask.com. http://www.ask.com. Última visita: 23-mayo-2008.
4. Aumueller, D., Do, H., Massmann, S., Rahm, E.: Schema and ontology matching with COMA++. SIGMOD Conference 2005: 906-908.
5. Baader, F., Küsters, R., Borgida, A., McGuinness, D.: Matching in Description Logics. J. Log. Comput. 9(3): 411-447 (1999).
6. Cabral L. et al.: Approaches to Semantic Web Services: an Overview and Comparisons. ESWS 2004: 225-239.
7. Cilibrasi, R., Vitányi, P.: The Google Similarity Distance. IEEE Trans. Knowl. Data Eng. 19(3): 370-383 (2007).
8. Dhamankar, R., Lee, Y., Doan, A., Halevy, A., Domingos, P.: iMAP: Discovering Complex Mappings between Database Schemas. SIGMOD Conference 2004: 383-394.
9. Domshlak, C., Gal, A., Roitman, H.: Rank Aggregation for Automatic Schema Matching. IEEE Trans. Knowl. Data Eng. 19(4): 538-553 (2007).
10. Drumm, C., Schmitt, M., Do, H., Rahm, E.: Quickmig: automatic schema matching for data migration projects. CIKM 2007: 107-116.
11. Ehrig, M., Staab, S., Sure, Y.: Bootstrapping Ontology Alignment Methods with APFEL. International Semantic Web Conference 2005: 186-200.

12. Ehrig, M., Sure, Y.: FOAM - Framework for Ontology Alignment and Mapping - Results of the Ontology Alignment Evaluation Initiative. Integrating Ontologies 2005.
13. Ernandes, M., Angelini, G., Gori, M.: WebCrow: A Web-Based System for Crossword Solving. AAAI 2005: 1412-1417.
14. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer-Verlag, 2007.
15. Falconer, S., Noy, N.: A. Ontology Mapping - A User Survey. The Second International Workshop on Ontology Matching, ISWC/ASWC 2007: 49-60.
16. Gal, A., Anaby-Tavor, A., Trombetta, A., Montesi, D.: A framework for modeling and evaluating automatic semantic reconciliation. VLDB J. 14(1): 50-67 (2005).
17. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Discovering Missing Background Knowledge in Ontology Matching. ECAI 2006: 382-386.
18. Gligorov, R., ten Kate, W., Aleksovski, Z., van Harmelen, F.: Using Google distance to weight approximate ontology matches. WWW 2007: 767-776.
19. Google. http://www.google.com. Última visita: 23-mayo-2008.
20. Hai Do, H., Rahm, E.: COMA - A System for Flexible Combination of Schema Matching Approaches. VLDB 2002: 610-621.
21. Kiefer, C., Bernstein, A., Stocker, M.: The Fundamentals of iSPARQL: A Virtual Triple Approach for Similarity-Based Semantic Web Tasks. ISWC/ASWC 2007: 295-309.
22. Lambrix, P., Tan, H.: A Tool for Evaluating Ontology Alignment Strategies. J. Data Semantics 8: 182-202 (2007).
23. Levenshtein, V: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics-Doklady, 10, pages 707-710, 1966.
24. Lee, Y., Sayyadian, M., Doan, A., Rosenthal, A.: eTuner: tuning schema matching software using synthetic scenarios. VLDB J. 16(1): 97-122 (2007).
25. Li Y., Li J., Zhang D., Tang, J.: Result of Ontology Alignment with RiMOM at OAEI'06. International Workshop on Ontology Matching collocated with the 5th International Semantic Web Conference.
26. Lycos. http://www.lycos.com. Última visita: 23-mayo-2008.
27. Martínez-Gil, J., Navas-Delgado, I., Polo-Marquez, A., Aldana-Montes, J.F.; Comparison of textual renderings of ontologies for improving their alignment. En Fatos Khafa and Leonard Barolli (Eds.). The Second International Conference on Complex, Intelligent and Software Intensive Systems. CISIS 2008, 4-7 March 2008, Polytechnic of Catalonia, Spain. Proceedings. págs. 871-876. IEEE Computer Society: Los Alamitos, California, 2008. ISBN: 0-7695-3109-1.
28. MSN. http://www.msn.com. Última visita: 23-mayo-2008.
29. Ontology Alignment Evaluation Initiative (OAEI). http://oaei.ontologymatching.org/2007. Última visita: 19-mayo-2008.
30. Pedersen, T., Patwardhan, S., Michelizzi, J.: WordNet: : Similarity - Measuring the Relatedness of Concepts. AAAI 2004: 1024-1025.
31. Prasad, A., Yu, C., Venkatasubrahmanian, R.: Similarity Based Retrieval of Videos. ICDE 1997: 181-190.
32. Results of the Ontology Alignment Evaluation Initiative 2007. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-304/paper9.pdf
33. Roitman, H., Gal, A.: OntoBuilder: Fully Automatic Extraction and Consolidation of Ontologies from Web Sources Using Sequence Semantics. EDBT Workshops 2006: 573-576.
34. Russian Ontology. http://www.aifb.uni-karlsruhe.de/WBS/meh/foam/ontologies/russia1.owl. Última visita: 03-mayo-2008.
35. Russian Ontology. http://www.aifb.uni-karlsruhe.de/WBS/meh/foam/ontologies/russia2.owl. Última visita: 03-mayo-2008.
36. Sociedad Española para el Procesamiento del Lenguaje Natural. http://www.sepln.org/.
37. van Harmelen, F.: Two Obvious Intuitions: Ontology-Mapping Needs Background Knowledge and Approximation. IAT 2006: 11.
38. Vazquez, R., Swoboda, N.: Combining the Semantic Web with the Web as Background Knowledge for Ontology Mapping. OTM Conferences (1) 2007: 814-831.
39. WordNet: http://wordnet.princeton.edu/ Última visita: 14-mayo-2008.
40. Yahoo!. http://www.yahoo.com. Última visita: 23-mayo-2008.
41. Ziegler, P., Kiefer, C., Sturm, C., Dittrich, K., Bernstein, A.: Detecting Similarities in Ontologies with the SOQA-SimPack Toolkit. EDBT 2006: 59-76.

Plataforma de Búsqueda Semántica en fuentes heterogéneas y su aplicación al comercio electrónico

Ana Flores Cuadrado[1] y M. Mercedes Martínez[2]

[1] Telefónica Investigación y Desarrollo
Parque Tecnológico de Boecillo, Parcela 120, 47151. Boecillo. Valladolid, España
{anafc }@tid.es
[2] Universidad de Valladolid, Edificio TIT.
Campus "Miguel Delibes" s/n, 47011. Valladolid, España
mercedes@infor.uva.es

Resumen. Se presenta una plataforma de búsqueda semántica que permite unificar y relacionar la información heterogénea de varias Webs, eliminando barreras idiomáticas y facilitando el uso de sinónimos o de diferentes divisas. La plataforma combina la utilización de ontologías y tesauros para mejorar los procesos de búsqueda semántica y extracción, homogeneizando el vocabulario de la información extraída y traduciendo las consultas al dominio de la ontología. El diseño de la plataforma separa en niveles distintos el funcionamiento y la definición del dominio, permitiendo su uso en diversos ámbitos, por ejemplo en el comercio electrónico.

Palabras clave: Buscador Semántico, Plataforma, Comercio electrónico.

1. Introducción

Las empresas utilizan masivamente Internet para realizar sus negocios. Sin embargo la gran cantidad de información disponible dificulta el comercio electrónico, impidiendo a los usuarios encontrar fácilmente los productos deseados. Los usuarios normalmente usan buscadores de tipo general como Google [9], Altavista, Yahoo, u otros orientados al *Business to Consumer (B2C)* como eBay. Estos buscadores hacen búsquedas por palabras claves [36] en vez de búsquedas por conceptos o semánticas (es decir, al buscar *billete de avión* se buscan documentos con las palabras *billete* o *avión*, y no se busca por el concepto *billete de avión*) [4]. Además estos buscadores no devuelven los mismos resultados si buscamos por *avión* que si buscamos por *aeroplano*, aunque son palabras equivalentes [1].

En el *B2C* se debe poder comparar la información obtenida de distintas páginas, y no sólo proporcionar los resultados de la búsqueda. De esto se encargan los *shopbots*, agentes autónomos de compra, que comparan los precios de los productos de tiendas virtuales, buscando el de mejor precio [18,15]. Pero los *shopbots* actuales no pueden procesar la información de fuentes que no se adaptan a un determinado formato o que hacen uso de vocabulario propio. Además desconocen cómo deben relacionar la información extraída de fuentes heterogéneas, expresada en idiomas diferentes, con precios en diversas divisas o que usan sinónimos.

Nuestro objetivo es demostrar cómo el uso combinado de ontologías [22] y tesauros hace más eficiente los procesos de extracción automática y búsqueda de información de fuentes heterogéneas, es decir, sobre webs del mismo dominio que utilizan distintas convenciones para representar los mismos conceptos, sinónimos, vocabulario propio o distintos idiomas. Con este fin hemos construido una plataforma que extrae datos de páginas marcadas semánticamente de distintos portales y unifica su formato, almacenando la información en un repositorio ontológico sobre el que los usuarios pueden efectuar búsquedas semánticas.

En la Sección 2 se describen los principales componentes de la plataforma y la utilización que se hace de las ontologías y de los tesauros. En la Sección 3 se presenta la aplicación de la plataforma al comercio electrónico. La Sección 4 presenta el trabajo relacionado. En la Sección 5 se concluye con las ventajas y limitaciones de la plataforma y el trabajo futuro.

2. Descripción de la plataforma

En esta sección se describen la arquitectura de la plataforma (Figura 1) y sus componentes.

Fig. 1. Arquitectura de la Plataforma

El **subsistema de extracción** recoge la información de los portales Web y la almacena en un repositorio ontológico, apoyándose en el uso de *ontologías* y *tesauros* para lograr la homogeneización de la información extraída a un vocabulario común. El **buscador semántico** permite realizar búsquedas guiadas por la estructura de la ontología en un repositorio semántico poblado previamente, y usa tesauros para expandir la consulta [30] mediante *substitución* de sinónimos, lo cual mejora la exhaustividad (*recall*) [31].

2.1. Subsistema de extracción

Este sistema está compuesto por 3 módulos: el **módulo de adquisición**, el **módulo de población** y el **módulo de inferencias** (Figura 1).

El **módulo de adquisición** está formado por un *Robot* y un *Web Crawler*[3]. El *Robot* visita las páginas Web de una lista, y las analiza buscando nuevas páginas que añadir a la lista a partir de los datos de los hiperenlaces. El *Robot* también invoca al *Web Crawler* para que procese la página cuando detecta unas marcas semánticas representadas mediante *Microformats*[4] (aunque también puede procesar marcas en *Resource Description Framework Attribute (RDFa o RDF/A)* [2]). La información se marca con *Microformats* incluyéndola entre etiquetas *HTML* representando su significado semántico con los valores de los atributos.

[3] Web Crawler: http://es.wikipedia.org/wiki/Web_crawler
[4] Microformats: http://Microformats.org/

El *Crawler* implementa un procesador del protocolo *Gleaning Resource Descriptions from Dialects of Languages (GRDDL)* [37]. El *GRDDL* es un estándar del *World Wide Web Consortium (W3C)* que intenta formalizar el concepto de *scrapper semántico* y automatizar la extracción de datos semánticos de formatos *XML* (como del *eXtensible Hypertext Markup Language (XHTML)*). El *Crawler*, mediante el procesador *GRDDL*, detecta en las páginas la etiqueta *HTML* con el atributo *rel = transformation* que le indica el algoritmo de extracción de los los datos, identificado por su *Uniform Resource Identifier (URI)*, y que normalmente es un fichero *Extensible Stylesheet Language Transformations (XSLT)*. El *XSLT* contiene las reglas para extraer la información, dotarla de semántica y almacenarla en un fichero *Resource Description Framework (RDF)* [32] compatible con la ontología, como se muestra en la Figura 2, donde el procesador identifica como algoritmo de transformación *http://.../dc-extract.xsl* (que sirve para saber cómo extraer los datos en formato *Dublin Core* [14]). Tanto el algoritmo, como el tipo de *Microformat* son configurables, siendo el funcionamiento del modulo independiente del dominio de la información procesada.

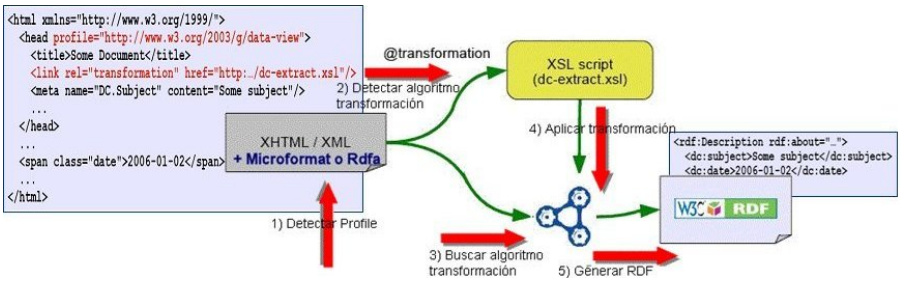


Fig. 2. Uso del GRDDL: Caso básico

El **módulo de población** homogeneíza el contenido de los datos *RDF*, lo clasifica y puebla la ontología (*Web Ontology Language (OWL)* [7,4]). La homogeneización se aplica sobre los valores de los atributos de los conceptos obtenidos del documento *RDF*. Unifica el formato de la información extraída a un vocabulario único mediante la aplicación de conversiones (cambios de divisa) o transformaciones (corrección de errores, sinónimos) y traducciones (idioma único) usando tesauros (*Simple Knowledge Organización System (SKOS)* [27]). La clasificación de la información se hace en función de una ontología dependiente del dominio para el que se utilice la plataforma. Durante la clasificación, se crean instancias con atributos y relaciones a partir de las tuplas del *RDF*, y que cumplan las condiciones suficientes para que una instancia pertenezca a la clase raíz. Si en el *RDF* no hay información de un atributo, se crean relaciones de negación para representar que se desconoce si el objeto posee un atributo o no. Estas relaciones permiten que al restringir el valor del atributo en las consultas se obtengan las instancias donde el atributo toma ese valor más aquellas en las que no el atributo no existe. También se crean relaciones específicas del dominio, es decir, si en la ontología existe una relación entre dos clases, se busca en el *RDF* la tupla correspondiente a esa relación. Si no existe y la relación no es necesaria, se modela con una relación de negación. Si es necesaria y no existe, la instancia se clasifica como del tipo tipo *owl.Thing*. Además, para modificar la ontología con nuevos datos y hacerla consistente con los datos anteriores se aplicará una política de actualización (en caso de conflicto decidirá si la nueva información será considerada como nuevas instancias o se actualizarán las existentes).

El **módulo de inferencias** usa un razonador (con interfaz *Description Logic Interface (DIG)*[5]) para inferir conocimiento adicional sobre la ontología poblada, procesando las

[5] DIG: http://dl.kr.org/dig/

instancias de la ontología para determinar si satisfacen todas las restricciones impuestas sobre alguna de sus clases. En el caso de que una instancia cumpla con todas las condiciones necesarias y suficientes definidas para una clase el razonador inferirá que la instancia debe pertenecer a esa clase, y le añadirá todos los atributos y relaciones que ésta hereda por su pertenencia a la clase. Si una instancia puede ser clasificada como de varias clases, el razonador tratará de clasificarla como una instancia de la clase más especializada. El resultado se almacenará en un repositorio semántico *(OWL)* accesible vía *HTTP*.

2.2. Subsistema de búsqueda semántica

El **buscador semántico** (Figura 1) permite hacer búsquedas sobre un repositorio ontológico vía *HTTP* combinando búsquedas semánticas (guiadas por la ontología), con operaciones de conversión y uso de tesauros para transformar la consulta al vocabulario usado en el repositorio. Su funcionamiento es independiente del dominio de la ontología. Esto es así gracias a la accesibilidad del repositorio vía *HTTP* y a que *SPARQL Query Language for RDF (SPARQL)* [10] permite consultar en repositorios mediante su *Universal Resource Locator (URL)*.

El usuario indicará en la **interfaz Web** la *URL* del repositorio, y se le mostrará una pantalla que le ayudará a efectuar la consulta. Esta pantalla se construye con un *XSLT* extrayendo los valores que determinarán las opciones que el usuario podrá seleccionar en la pantalla de algunas etiquetas del *OWL*, como *owl:Class, owl:Restriction, rdfs:subClassOf, owl:ObjectProperty, owl:DataProperty,etc.*. El usuario podrá limitar los valores de los atributos de las instancias o añadir restricciones más complejas navegando a través de las relaciones entre dos, tres o más conceptos de la ontología. Como puede apreciarse, las opciones disponibles en la interfaz de consulta del usuario se determina en función de la ontología utilizada. La consulta se envía en un mensaje *XML* al **motor** y contiene toda la información necesaria para traducirla a *SPARQL*, incluyendo el repositorio, la clase buscada, los valores y tipo de atributos o las relaciones restringidas, etc. Un ejemplo del *XML* para la consulta correspondiente al caso práctico de la sección 3 es el siguiente:

```
<consulta xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="schema.xsd" clase="GamaAlta" lang="es">
  <ontologia url="http:..../ModeloInferido.owl"" />
  <condicion><compuesta operador="and"> .....
    <relacion clase="GamaAlta" nombre="vendido_en_tienda">
        <relacion clase="Tienda" nombre="tiene_pv">
          <relacion clase="PuntoDeVenta" nombre="ubicado_en">
             <propiedad clase="Direccion" nombre="cp" valor="37008"... />
    </relacion></relacion></relacion>.....
  </condicion> ...
    <result clase="GamaAlta" campo="nombreMovil" />
    <result clase="Marca" campo="nombreMarca" />
   .... <orden clase="GamaAlta" campo="nombreMovil" sentido="desc" />
 </consulta>
```

Antes de ejecutar la consulta, el **motor** reemplazará los valores de los atributos que coincidan con términos alternativos de los tesauros por sus conceptos equivalentes. Sobre estos valores también se aplicarán conversiones dependiendo de la naturaleza de los mismos. Además podrá añadir otras condiciones a la consulta para obtener como resultado instancias de las subclases e instancias de clases de las que se desconoce la existencia de ciertos atributos. El resultado de la consulta es devuelto a la **Interfaz Web** en un *XML* estándar [38].

3. Caso práctico

El ejemplo de búsqueda semántica elegido consiste en la búsqueda de móviles de determinados precios y marcas. Los datos están extraídos de Webs de distintas tiendas de móviles. Cada tienda *on-line* tiene asociados varios puntos de venta, con una dirección postal, y usa un idioma y vocabulario propios. Los precios pueden estar expresados en monedas distintas. Se permite buscar móviles comparando sus precios, o buscar móviles que se vende en una dirección, sin conocer el nombre de la tienda donde se venden.

3.1. Definición de la ontología y los vocabularios

Muchos autores han tratado de definir una ontología a partir de los estándares del comercio electrónico [18,21], pero la gran cantidad de vocabularios especializados [41] dificultan la integración de los estándares. Recientemente han aparecido nuevas ontologías basadas en estos estándares (*Electronic Business XML (ebXML)*[6], *Universal Business Language (UBL) Ontolgy*[7], *Universal Standard Products and Services Classification (UNSPSC)* [8], *International Standard for the Classification and Description of Products and Services (eCl@ss)*[9]). Algunas, como el *UNSPSC* o el *eCl@ssOwl* no tienen características intrínsecas al *e-Business* (como el precio o el IVA) y son jerarquías de productos y servicios. Otras, como *UBLOntolgy* están en fase de definición. Como creemos que en un futuro *UBLOntolgy* será un estándar, creamos una ontología aplicando la metodología *METHONTOLOGY* [19] para probar el funcionamiento de la plataforma. Por ejemplo en la ontología (Figura 3) se indica que *Móvil* es subclase de la clase *Producto*; todo *Producto* se vende en una *Tienda* con un *Precio* único; cada *Móvil* tiene una *Marca* y un *Modelo*; y que cada *Tienda on-line* tiene asociado un *Punto de Venta* con una *Dirección*.

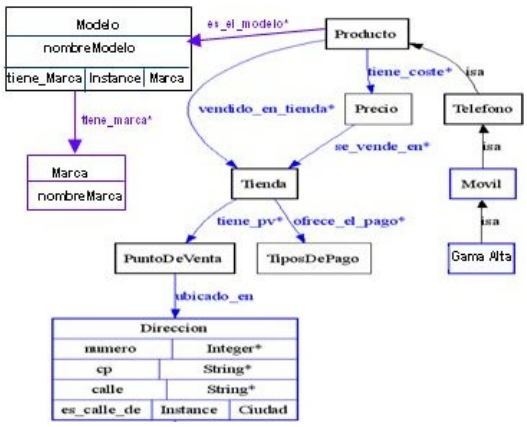


Fig. 3. Ontología del caso práctico

Por compatibilidad con los estándares del *W3C*, hemos limitado el uso de tesauros a su representación con *SKOS* [27]; sin embargo el número de tesauros representados con *SKOS*

[6] ebXML Registry Profile for Web Ontology Language OWL: http://www.oasis-open.org/committees/download.php/17042/regrep-owl-profile-1.0-draft %204.pdf
[7] UBLOntolgy: http://ontolog.cim3.net/cgi-bin/wiki.pl?UblOntology
[8] UNSPSC: http://www.eccma.org/unspsc/browse/ ; http://www.cs.vu.nl/ mcaklein/unspsc/
[9] eCl@ssOwl: http://www.heppnetz.de/eclassowl/

para el dominio del caso es limitado. Los tesauros se utilizan al poblar la ontología para almacenar en el repositorio información en un único idioma, con los precios en una divisa común, y con conceptos correspondientes a los términos principales del tesauro. También se utilizan para expandir la consulta.

3.2. Ejemplo de extracción

En el ejemplo extraeremos los datos del móvil de marca *Sony Erics*, modelo *V630i Blanco* con *Bluetooth*, *Mp3* y *3G* que se vende en la tienda *Vodafone* a *59 euros*.

Durante el proceso de extracción, el **modulo de adquisición** procesa únicamente las páginas donde aparezca el *Microformat producto odd* y que sigan el protocolo *GRDDL*. Para marcar los datos hemos utilizado un *Microformat* a medida (existe un *Microformat* estándar, el *hReview*, para descripciones de productos del *e-Business*, pero carece del campo precio; otro distinto, *hListing*, describe el uso de un objeto e incluye el precio, pero no es estándar). Los atributos del móvil se etiquetan en *HTML* con valores concretos del atributo *class*. Por ejemplo, el precio del móvil sería el contenido de las etiquetas *div class ="precio"* y *div class ="moneda"*. El fichero *RDF* generado contiene los datos de los móviles, y también los datos de la tienda donde se venden, y además para cada tienda online, la dirección de los puntos de venta. Con los datos *RDF*, el **modulo de población** crea las instancias de los móviles y de las tiendas. Antes de almacenarlos, busca entre los valores de los atributos todos aquellos valores que se correspondan con los términos alternativos del tesauro *(SKOS)*, por ejemplo la marca *Sony Erics*, y los reemplaza por los términos preferidos, por ejemplo *Sony Ericsson*. El **modulo de población** también aplica otro tipo de conversiones como la conversión de divisas, por ejemplo para convertir el precio del móvil de *59 euros* a la divisa utilizada en el repositorio (*78,47 dólares*). Al poblar la ontología también se modelan las relaciones del dominio (Figura 3) que indican que un móvil se vende en una tienda que tiene distintos puntos de venta. En la ontología hemos definido reglas para aplicar inferencia y clasificar como móviles de *Gama Alta* aquellos que tengan *3G* o que tengan *Radio*, *Camara*, *Bluetooth* y *MP3*. El resultado de la inferencia se almacenará en un fichero *OWL* accesible mediante *HTTP*.

3.3. Ejemplo de búsqueda

En las búsquedas los usuarios eligen la clase de la ontología sobre la que buscar, los atributos de la clase que se mostrarán, y los atributos que se utilizarán para ordenar los resultados (Figura 4). También pueden restringir las condiciones de búsqueda, limitando tanto los valores de los atributos de la clase buscada, como los valores de los atributos instancias con los que se relaciona la clase buscada.

En el ejemplo de la figura 4 se buscan móviles de *Gama Alta* de la marca *Sony Ericcson* y a un precio menor que *100$* que se vendan en tiendas con código postal *37008*. Para ello se introducen distintas restricciones. Como se ve en la Figura 3, la propiedad *vendido_en_tienda* relaciona las instancias de las clases *Producto* y *Tienda*. Mediante la propiedad *tiene_pv* se relacionan las instancias de las clases *Tienda* y *PuntodeVenta* y por último la propiedad *ubicado_en* relaciona las instancias de *PuntodeVenta* y *Dirección*. Navegando a través de estas relaciones y limitando el valor del atributo *cp* de la clase *Dirección* a *37008* (Figura 4) se restringe la dirección del punto de venta en la consulta. La restricción sobre la marca se establece de forma similar, a través de las relaciones entre *Producto*, *Modelo* y *Marca*. En el caso de que el usuario introduzca valores en los atributos que no coinciden con ninguno de los almacenados en el repostorio, como *Sony Erics*, el **buscador** usará los tesauros para reemplazar (en la consulta *SPARQL*) el valor introducido por su equivalente en el repositorio, por ejemplo, *Sony Ericcson*. El **buscador** también usa tesauros para traducir la divisa al valor del repositorio *dólar*. La consulta traducida en *SPARQL* es:

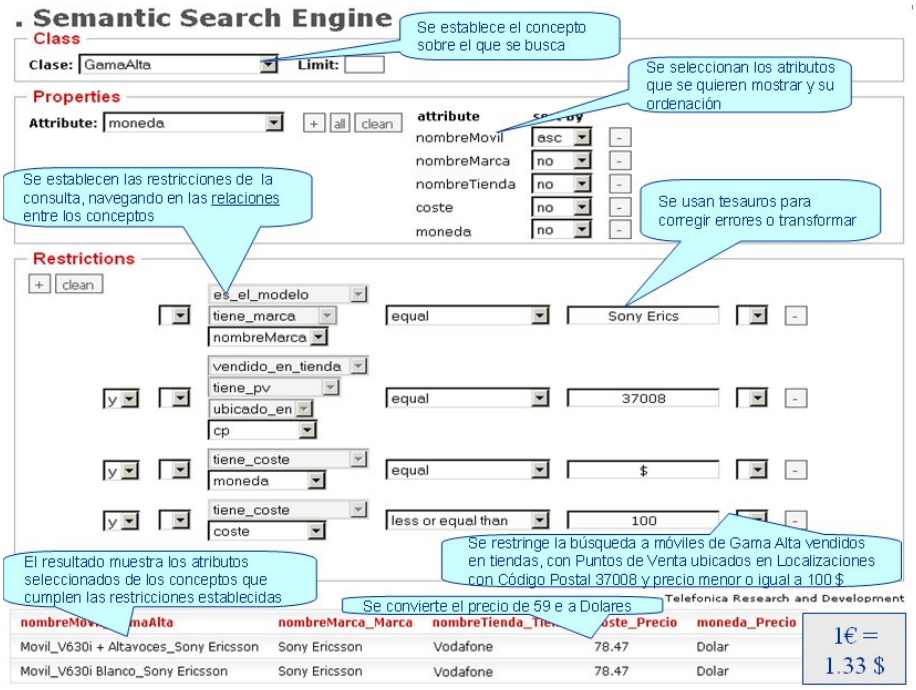


Fig. 4. Búsqueda Semántica y resultados

```
PREFIX rdf: http:....  PREFIX mov: http:...
SELECT DISTINCT ?nombreMovil_Movil ?nombreMarca_Marca
  ?nombreTienda_Tienda ?cp_Direccion ?coste_Precio ?moneda_Precio
FROM <http://localhost:8080/ontologias/ModeloInferido.owl>
WHERE{?clase rdf:type mov:Movil.
   ....?clase mov:vendido_en_tienda ?vendido_en_tienda_Movil .
   ?vendido_en_tienda_Movil mov:tiene_pv ?tiene_pv_Tienda .
   ?tiene_pv_Tienda mov:ubicado_en ?ubicado_en_PuntoDeVenta .
   ?ubicado_en_PuntoDeVenta mov:cp ?cp_Direccion .
   .....OPTIONAL {?clase mov:nombreMovil ?nombreMovil_Movil }.
   OPTIONAL {?clase mov:nombreMarca ?nombreMarca_Marca }  .
   .....FILTER ((( ?nombreMarca_Marca = "Sony Ericsson" ))&&
   (((?cp_Direccion = "37008" ))&& ...} ORDER BY asc (?nombreMovil_Movil)
```

Los resultados de la búsqueda (Figura 4) contienen el móvil *Sony Ericsson V630i Blanco*, pero con su precio original de *59 euros* en dólares. El **motor** los encapsula en un *XML* de formato similar al que se muestra a continuación y se lo envía el *XML* a la **interfaz Web**.

```
<sparql xmlns:rdf="http:... xmlns:xs="http:... xmlns:res="....">
 <head><variable name="nombreMovil_Movil"/><variable name="nombreMarca_Marca"/> .....
 <results ordered="true" distinct="true"> ...
     <binding name="nombreMovil_Movil"><literal datatype=".../XMLSchema#string">
          Movil_V630i Blanco_Sony Ericsson</literal></binding>
     <binding name="nombreMarca_Marca">
       <literal datatype=".../XMLSchema#string">Sony Ericsson</literal></binding>
  .....
</sparql>
```

4. Trabajos relacionados

Muchos autores han usado *wrappers* para la extracción en el dominio del comercio electrónico. Para la construcción de reglas se han hecho análisis de plantillas o etiquetas [5], o de árboles *DOM* y patrones [12]. Algunos han ideado un método automático de generación de reglas haciendo análisis de las etiquetas *HTML* [35] para la extracción. Otros autores usan técnicas inductivas y de reconocimiento de patrones para la construcción de *shopbots* [15]. El problema de estas técnicas radica en que la información extraída carece de significado semántico y no es posible conocer su significado una vez extraída. Para solventar este problema, otras iniciativas crean *wrappers* que utilizan ontologías o semántica para dotar de significado a la información extraída [28]. En el dominio de *B2B*, [39] combina el reconocimiento de patrones con analizadores semánticos, [17] usa una ontología para reconocer los patrones y las relaciones que se deben extraer, o [8] hace uso de ontologías para calcular la equivalencia semántica de los elementos *B2B*. Otros sistemas [16] que extraen información de páginas *HTML* apoyándose en la semántica utilizan razonadores que permiten crear nuevas relaciones (por ejemplo entre los productos y sus especializaciones). En esta línea nuestro sistema utiliza marcas semántica en lugar de *wrappers* para la extracción, pero usa semántica y razonadores para mejorar el significado y homegeneizar la información extraída.

En el ámbito de la búsqueda semántica se usan anotaciones *RDFs* [33] y ontologías para facilitar la búsqueda [26]. *SEWASIE* [11] utiliza una ontología de dominio para crear la interfaz de consulta, en la cual el usuario selecciona los conceptos con los que restringe la búsqueda seleccionando clases y sus instancias. *CIRI* [3] también proporciona una interfaz basada en una ontología donde el usuario selecciona los conceptos para restringir la búsqueda. ésta se realiza usando buscadores tradicionales, donde las palabras clave serán los conceptos y subconceptos seleccionados unidos por *OR*. En *SHOE* [24], el usuario selecciona la clase cuyas instancias desea buscar en una interfaz que muestra un árbol con las clases de la ontología. En función de la clase elegida se presenta un formulario que contiene sus relaciones y atributos. El usuario puede introducir palabras clave en los campos del formulario para buscar instancias que tengan en sus atributos y relaciones alguna de esas palabras. Cada instancia representa un documento categorizado dentro esa clase o concepto. Una aproximación similar es la del portal *SEAL* [29]. y las de *OntoIR* [20] y *QuizRDF* [13], que combinan la búsqueda basada en la estructura de la ontología con la búsqueda en texto libre. *GRQL* [6] presenta una interfaz basada en ontologías, donde se permite seleccionar una clase y visualizar todas sus propiedades. Cuando el usuario selecciona una propiedad se despliega el grafo asociado, y se crea un patrón de búsqueda del tipo *Clase propiedad Clase relacionada* donde la *Clase relacionada* se refiere tanto a la clase o como a sus subclases. Por último *Piggy Bank* [25] permite al usuario construir scrapper semánticos para extraer información de páginas Web anotando los datos extraídos mediante conceptos. En el dominio del comercio electrónico, *WISE* [23] usa semántica para la integración de los resultados de otros buscadores.

5. Conclusiones y Trabajo Futuro

Se ha presentado una plataforma de búsqueda semántica en fuentes heterogéneas que combina el uso de ontologías y tesauros, junto con su aplicación al comercio electrónico. Según la clasificación propuesta en [31], el sistema tiene una arquitectura *Stand-alone*, y usa la estructura de sinónimos de los tesauros para expandir y modificar la consulta de modo que mejore la exhaustividad. Los usuarios pueden hacer búsquedas semánticas estructuradas, de forma manual, guiadas por una ontología en *OWL*, con un navegador Web. Las búsquedas se hacen sobre un repositorio accesible vía *HTTP*, con lo cual el acoplamiento es alto y no

se tiene en cuenta el contexto del usuario para buscar. La arquitectura de la plataforma tiene 2 niveles que aíslan su modo de funcionamiento de la definición del dominio. Tanto la ontología como los algoritmos de extracción y población se definen en ficheros externos, lo que permitiría utilizar distintos ficheros según el ámbito de uso de la plataforma.

Una limitación de la plataforma es que es aplicable a Web´s con páginas que estén marcadas semánticamente siguiendo el protocolo *GRDDL*. ésta podría evitarse utilizando técnicas de extracción de información y web mining [40]. Para mejorar la búsqueda se considera la posibilidad de incluir información contextual y el uso de tesauros como *Wordnet* [34] para expandir la consulta, así como identificar los conceptos, relaciones y atributos de la ontología utilizada en las consultas a partir de palabras clave [30]. Finalmente, nos planteamos manejar otras relaciones definidas en tesauros y ontologías como *hypernyms*, *hyponyms* u *homonyms*, además de las relaciones de sinonimia, para expandir la consulta [31].

Referencias

1. M.A. Abían. El futuro de la web. xml, rdf/rdfs, ontologías y la web semántica. http://www.javahispano.org/contenidos/es/el_futuro_de_la_web/, 2003.
2. B. Adida and M. Birbeck. Rdfa primer embedding structured data in web pages, March 2008.
3. E. Airio, K Järvelin, P. Saatsi, J. Jaana Kekäläinen, and S. Suomela. CIRI - an ontology-based query interface for text retrieval. In *Proceedings of the 11th Finnish Artificial Intelligence Conference STeP 2004, September 1-3, Vantaa, Finland*, volume 2 of *Conference Series – No 20*, pages 73–82. Finnish Artificial Intelligence Society, 2004.
4. G. Antoniou and F. vanHarmelen. *A Semantic Web Primer*. MIT Press, Cambridge, 2004.
5. A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proceedings of the ACM SIGMOD international conference on Management of data*, pages 337–348, New York, NY, USA, 2003. ACM.
6. Nikolaos Athanasis, Vassilis Christophides, and Dimitris Kotzinos. Generating on the fly queries for the semantic web: The ics-forth graphical rql interface (grql). In Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen, editors, *International Semantic Web Conference*, volume 3298 of *Lecture Notes in Computer Science*, pages 486–501. Springer, 2004.
7. S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrock, D.L. McGuinness, P.F. Patel-Schneider, L.A. Steinnd, and F.W. Olin. Owl web ontology language reference, February 2004.
8. B. Bornhövd. Semantic metadata for the integration of web-based data for electronic commerce. In *Proc. of the International Workshop on Advance Issues of ECommerce and Web-based Information System, Santa Clara, USA*, 1999.
9. S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.
10. J. Broekstra and D. Beckett. Sparql query language for rdf, January 2008.
11. T. Catarci, T. Di Mascio, E. Franconi, G. Santucci, and Tessaris. S. An ontology based visual tool for query formulation support. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence*, pages 308–312. IOS Press, 2004.
12. C. Chang, C. Hsu, and S Lui. Automatic information extraction from semi-structured web pages by pattern discovery. *Decis. Support Syst.*, 35(1):129–147, 2003.
13. J. Davies and R. Weeks. Quizrdf: Search technology for the semantic web. In *HICSS '04: Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, page 40112, Washington, DC, USA, 2004. IEEE Computer Society.
14. Dublin Core Metadata Initiative (DCMI). Expressing dublin core in html/xhtml meta and link elements., November 2003. http://dublincore.org/documents/dcq-html/.
15. R.B. Doorenbos, Etzioni O., and D.S. Weld. A scalable comparison-shopping agent for the www. In W. Lewis Johnson and Barbara Hayes-Roth, editors, *Proc. of the 1th International Conference on Autonomous Agents*, pages 39–48. ACM Press, 1997.
16. D. Embley, C. Tao, and S. Liddle. Automating the extraction of data from html tables with unknown structure. *Data Knowl. Eng.*, 54(1):3–28, 2005.

17. D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, Y.Ñg, D Quass, and R. D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data Knowledge Engineering*, 31(3):227–251, 1999.
18. D. Fensel. *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, 2003.
19. M. Fernández, A. Gómez-Pérez, and N. Juristo. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, pages 33–40, Stanford, USA, March 1997.
20. E. García and M.A. Sicilia. Designing ontology-based interactive information retrieval interfaces. In *Proceedings of OTM Workshops*, pages 152–165. Springer, 2003.
21. A. Gómez-Pérez, O. Corcho-García, and M. Fernández-López. *Ontological Engineering*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.
22. T. R. Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993. Kluwer Academic Publishers.
23. H. He, W. Meng, C. Yu, and Z. Wu. Wise-integrator: a system for extracting and integrating complex web search interfaces of the deep web. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1314–1317. VLDB Endowment, 2005.
24. J. Heflin and J. Hendler. Searching the web with shoe. In *In AAAI-2000 Workshop on Artificial Intelligence for Web Search*, pages 35–40. Press, 2000.
25. D. Huynh, S. Mazzocchi, and D. Karger. Piggy bank: Experience the semantic web inside your web browser. *Web Semant.*, 5(1):16–27, 2007.
26. Eero Hyvönen, Samppa Saarela, and Kim Viljanen. Application of ontology techniques to view-based semantic search and browsing. In Christoph Bussler, John Davies, Dieter Fensel, and Rudi Studer, editors, *ESWS*, volume 3053 of *Lecture Notes in Computer Science*, pages 92–106. Springer, 2004.
27. A. Isaac and E. Summers. Skos simple knowledge organization system primer, February 2008.
28. A. H.F. Laender, B.A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Record*, 31(2):84–93, 2002.
29. A. Maedche, S. Staab, N. Stojanovic, R. Studer, and Y. Sure. Seal - a framework for developing semantic web portals. In *BNCOD 18: Proceedings of the 18th British National Conference on Databases*, pages 1–22, London, UK, 2001. Springer-Verlag.
30. E. Mäkelä. Survey of semantic search research. In *Proceedings of the Seminar on Knowledge Management on the Semantic Web*, 2005.
31. C. Mangold. A survey and classification of semantic search approaches. *Int. J. Metadata Semant. Ontologies*, 2(1):23–34, 2007.
32. F. Manola and E. Miller. Rdf primer, February 2004.
33. M. Michelson and A. K. Craig. Semantic annotation of unstructured and ungrammatical text. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 1091–1098, 2005.
34. Dan I. Moldovan and Rada Mihalcea. Using wordnet and lexical operators to improve internet searches. *IEEE Internet Computing*, 4(1):34–43, 2000.
35. A.Ñaveen and A. K. Craig. Semi-automatic wrapper generation for internet information sources. In *Second International Conference onCooperative Information Systems*, 1997.
36. T.B. Passin. *Explorer's Guide to the Semantic Web*. Manning Publications Co, 2004.
37. A. Seaborne and E. Prud´ hommeaux. Gleaning resource descriptions from dialects of languages (grddl), september 2007.
38. A. Seaborne and E. Prud´ hommeaux. Sparql query results xml format, January 2008.
39. S. Soderland. Learning information extraction rules for semi-structured and free text. *Mach. Learn.*, 34(1–3):233–272, 1999.
40. G. Stumme, A. Hotho, and B. Berendt. Semantic web mining: State of the art and future directions. *Web Semantics: Science, Services and Agents on the WWW*, 4(2):124–143, June 2006.
41. Zhao Y. Develop the ontology for internet commerce by reusing existing standards. In *Proc. of International Workshop on Semantic Web Foundations and Application Technologies*, pages 51–57, 2003.

Modelado Arquitectónico Orientado a Servicios de Estrategias de Coordinación Inspiradas en Tácticas Deportivas

Marcos López-Sanz, Carlos E. Cuesta, Esperanza Marcos, Jesús Domínguez

Departamento de Lenguajes y Sistemas II, Universidad Rey Juan Carlos
C/ Tulipán, S/N – 28933 Móstoles (Madrid)
{marcos.lopez, carlos.cuesta, esperanza.marcos, jesus.dominguez}@urjc.es

Resumen. El paradigma orientado a servicios ha sido considerado tradicionalmente como la solución a los problemas de integración entre aplicaciones de distintas organizaciones. Uno de los aspectos que más influyen a la hora de integrar diferentes aplicaciones es la forma de coordinar los diferentes servicios que participan en la solución de integración. Las coreografías de servicios permiten coordinarlos sin que ninguno de ellos dirija el proceso de coordinación, sino que todos son considerados como iguales. Este artículo expone el desarrollo de estrategias de coordinación basadas en coreografías de servicios utilizando un enfoque dirigido por modelos y centrado en la arquitectura. La validez de la propuesta de modelado se demuestra utilizando una metáfora deportiva en un entorno de cooperación real: una situación de juego puntual en el transcurso de un partido de baloncesto.

Palabras clave. SOA, Desarrollo Dirigido por Modelos, Estrategias de Coordinación, Situaciones Tácticas Deportivas, Coreografías de Servicios

1 Introducción

El paradigma de la orientación a servicios (*Service-Oriented Computing*, SOC) [26] se relaciona habitualmente con la posibilidad de integrar e interconectar aplicaciones pertenecientes a diferentes organizaciones o empresas [14]. Es por ello que uno de los aspectos que más importancia tiene durante el desarrollo de soluciones orientadas a servicios sea, junto con el dinamismo, la composición de servicios [27]. Desde un punto de vista conceptual, la composición de servicios ha de entenderse como la coordinación de las funcionalidades ofrecidas por cada uno de ellos con el fin de alcanzar una meta común.

La coordinación entre servicios puede darse, principalmente, de dos formas: como orquestación o como coreografía [28]. En el primero de los casos uno de los participantes actúa de coordinador definiendo qué funcionalidad de qué servicio es la que debe ejecutarse en cada momento. Esta opción es la que tradicionalmente se utiliza para la implementación de procesos de negocio basados en servicios que realizan tareas pertenecientes a flujos de trabajo complejos [5]. En otros casos, sin embargo, la consecución de un objetivo o de una funcionalidad concreta no se basa en la ejecución secuencial de una serie de operaciones, sino que se obtiene a partir de la interrelación entre servicios equivalentes, en dicho caso se habla de coordinación de servicios basados en coreografías. El entorno que habitualmente se utiliza como ejemplo de integración basada en coreografías es el de la gestión de mercados [12][35] aunque, como se verá más adelante, no es el único.

En los últimos años han aparecido estrategias de diseño de coreografías basadas en tecnologías y estándares de bajo nivel tales como WS-Coordination [21], WS-CDL [31] o WS-CF [9], por nombrar algunos de los múltiples lenguajes existentes. No obstante, el contexto de aplicación de la integración de aplicaciones a gran y media escala es lo suficientemente complejo como para

requerir un enfoque de desarrollo de más alto nivel que no sea tan dependiente de la tecnología de implementación.

Para poder obtener una visión más intuitiva de la integración basada en servicios es necesario seguir una estrategia conceptualmente más genérica. Uno de los enfoques de desarrollo que más importancia y atención está recibiendo en los últimos años es el enfoque dirigido por modelos y en particular la propuesta de MDA (*Model-Driven Architecture*, [19]).

Actualmente, existen múltiples metodologías que permiten diseñar y representar soluciones orientadas a servicios [2][6][15][32][34]. Sin embargo, hasta donde nosotros conocemos, pocas o ninguna de ellas aborda el problema de la integración de aplicaciones a nivel de modelado desde el punto de vista de la coordinación de servicios basada en coreografías. El objetivo principal de este artículo es demostrar que la representación de estrategias de coordinación mediante modelos es viable para ser aplicada en un entorno orientado a servicios.

No obstante, para facilitar el modelado del sistema sería recomendable disponer de una metáfora adecuada para describir las coreografías de servicios a alto nivel. Se plantea como analogía válida la descripción de sistemas tácticos en deportes de equipo. Ésta tiene la ventaja de proporcionar una serie de estrategias de coordinación ya predefinidas, lo que la convierte en un esquema particularmente interesante. Además, como objetivo secundario, se pretende demostrar que el uso de sistemas tácticos para deportes de equipo proporciona una estrategia básica de coordinación en el contexto descrito.

En nuestro caso, utilizaremos la descripción de modelos tácticos existentes para deportes de equipo [33] como metáfora para desarrollar el modelado de arquitecturas orientadas a servicios en las que se requiere una coordinación entre ellos. La analogía deportiva resulta adecuada en nuestro caso por tratarse de un entorno en el que entidades aparentemente independientes (jugadores representados por servicios) se comunican entre sí con el fin de obtener un objetivo o funcionalidad común (o individual, según sea el caso) [29]. Mediante esta metáfora intentamos resolver el problema de la integración de aplicaciones que puedan ser transformadas en servicios o que puedan ofrecer sus funcionalidades de forma similar. La reducción del problema a modelar coreografías de servicios nos permite centrarnos en la comprobación de la viabilidad de tal aproximación.

Utilizaremos, como estrategia de coordinación a modelar, una determinada situación de juego: el *bloqueo directo* [22] durante el transcurso de un partido de baloncesto. Esta estrategia servirá para demostrar la viabilidad de la estrategia de modelado para definir coreografías de servicios y el método de desarrollo utilizado.

La estructura del artículo es la que sigue: en la sección 2 se expone el contexto metodológico dirigido por modelos en el que se enmarca la estrategia de modelado de coreografías de servicios. En la sección 3 se presenta una estrategia concreta de coordinación y se analiza la modelización de arquitecturas de servicios que representan coreografías. La sección 4 resume algunos de los principales trabajos relacionados y finalmente la sección 5 presenta las principales conclusiones y líneas de investigación abiertas a partir del tema presentado en el artículo.

2 Contexto metodológico

La composición de servicios como solución al problema de la integración de aplicaciones no se restringe a la modelización de la coordinación en sí, sino que tiene que llevar asociado una metodología de desarrollo que permita crear una solución integral basada en servicios. Por ello, en esta sección se presenta, brevemente, el marco metodológico SOD-M (*Service-Oriented Development Method*, [7]). En él se engloba el estudio de las estrategias de coordinación de servicios basadas en coreografías. Además, dichas estrategias se analizarán desde el punto de vista de la arquitectura, el cual permite una mejor representación de las relaciones entre los diferentes elementos que participan en la solución software [17].

El enfoque dirigido por modelos (*Model-Driven Development*, MDD) se ha postulado como una de las principales alternativas de elección para el desarrollo de sistemas. Este enfoque se caracteriza por utilizar el concepto de *modelo* como artefacto principal del proceso de desarrollo. Una de las propuestas concretas de MDD es MDA [23], que añade a la característica anterior la idea de dividir el proceso de desarrollo en conjuntos de modelos en diferentes niveles de abstracción (CIM, PIM y PSM). Otra de las principales aportaciones de este enfoque es la definición de transformaciones automáticas entre los modelos de diferentes niveles y entre modelos del mismo nivel.

Como se ha comentado anteriormente, la idea de utilizar un enfoque dirigido por modelos para el desarrollo de soluciones orientadas a servicios no es nueva. Existen múltiples metodologías que abordan esta problemática [2][6][15][32][34]. Una de ellas es SOD-M [7] que aboga por el desarrollo de sistemas siguiendo un enfoque ACMDA (*Architecture-Centric Model-Driven Architecture*), tal y como se expone en [17].

SOD-M sigue un enfoque dirigido por modelos multidimensional, donde cada dimensión representa un aspecto del proceso de desarrollo. Por un lado, SOD-M recoge el enfoque MDA a través de la separación en niveles de abstracción describiendo los modelos a definir en cada nivel. Por otro lado, la arquitectura juega un papel director dentro de la metodología ya que, con la creación de un modelo de la arquitectura, es posible identificar qué elementos se incluyen en cada modelo de cada nivel de abstracción. Finalmente, SOD-M es una metodología orientada a servicios, ya que el concepto principal en el que se basa el proceso de desarrollo es el servicio. Por ello, se han definido desde un metamodelo de servicios de usuario hasta modelos de descripción de la arquitectura a diferentes niveles.

Consideramos que el primer contacto real con la estrategia de coordinación, durante el proceso de desarrollo del sistema, se establece en el modelado de la arquitectura a nivel PIM (*Platform Independent Model*). En este nivel se definen aquellos modelos que son independientes de la plataforma y que por lo tanto se abstraen totalmente de cualquier aspecto tecnológico. En el siguiente apartado se exponen las principales características de este modelo.

2.1 Metamodelo de Arquitecturas Orientadas a Servicios

Esta sección expone, de forma breve, el metamodelo de arquitecturas orientadas a servicios de nivel PIM (ver [16] para una descripción más detallada). Este metamodelo tiene como elemento central la definición del concepto de servicio, entendido como *un mecanismo que permite el acceso a un conjunto de una o más capacidades donde dicho acceso es ofrecido a través de un interfaz bien conocido y se lleva a cabo de acuerdo a las restricciones y políticas especificadas en la descripción del servicio* [19].

Todo servicio además pertenece a una determinada organización, modelada como proveedor de servicios (*ServiceProvider*), que representa a aquellas organizaciones que controlan cada servicio que pertenecerá a la solución software. Cada una de estas entidades puede pertenecer a uno de los dos tipos de proveedores definidos: *proveedores externos* de servicios o *proveedores internos.*

Por otra parte, todo servicio se identifica en la solución software mediante un identificador único, denominado *SERVID*, que permite identificar, de forma unívoca cada instancia de servicio existente en el sistema. Además, cualquier servicio ofrece su funcionalidad a través de *operaciones*, consideradas como funcionalidades atómicas independientes. Cada operación puede ser de dos tipos: síncrona o asíncrona, dependiendo de si el servicio que invoca la operación debe esperar o no por la respuesta a la operación (si es que existe).

Los servicios se relacionan entre si a través de de conectores arquitectónicos representados por los *Contratos de Servicio*. Cada uno de estos contratos permite la comunicación punto a punto entre servicios caracterizado por un patrón de intercambio de mensajes. Este patrón puede ser de varios tipos, destacando las opciones de: *One-Way* o comunicación en una sola dirección, *Query/Response* en cuyo caso el tipo de comunicación que se da está formado por la petición de invocación de la operación en un sentido y el envío de la respuesta por otro (de forma síncrona o

asíncrona, según sea la operación); finalmente, el protocolo de comunicación puede estar basado en un intercambio complejo de mensajes, en cuyo caso hablaremos de *Dialogue*.

Los servicios que participan en una solución software basada en este paradigma pueden clasificarse según diferentes criterios. Dependiendo del tipo de operaciones que implementan, podemos diferenciar entre *servicios de interacción*, en el caso de que implementen al menos una operación de tipo síncrona (este tipo de servicio no existe a nivel de implementación, pero se representa a nivel de modelado con el fin de identificar a aquellos servicios capaces de iniciar conversaciones con otros servicios); o *servicios tradicionales* en los que las operaciones son únicamente asíncronas. Por otro lado, de acuerdo al rol realizado por el servicio dentro de la arquitectura del sistema, podemos distinguir entre *servicios básicos*, que ofrecerán funcionalidades propias del sistema software al que pertenecen; o *servicios de soporte*, que representan a aquellos servicios cuya funcionalidad no está relacionada con los requisitos funcionales del sistema sino que realizan tareas de soporte, tales como descubrimiento de servicios o tareas de orquestación en servicios compuestos, necesarias para que el resto de servicios cumplan con su cometido. Finalmente, los servicios pueden clasificarse de acuerdo a su atomicidad, pudiendo, de esta forma, identificar *servicios simples* y *servicios compuestos*. Este aspecto es el que más nos interesa enfatizar puesto que es donde hemos de fijarnos a la hora de diseñar estrategias de coordinación basadas en servicios.

La Figura 1 muestra parte de este metamodelo. En concreto, se muestran los elementos necesarios para poder representar la composición de servicios.

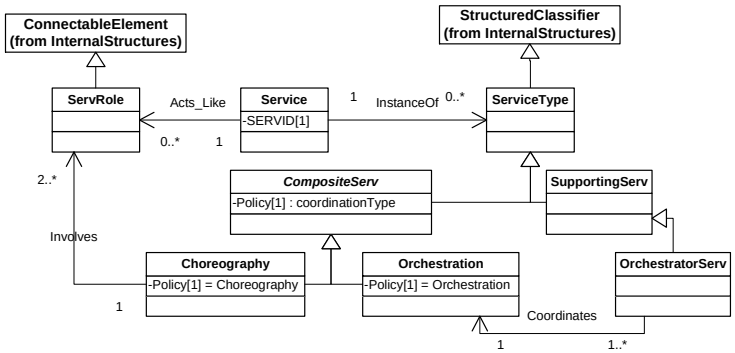


Fig. 1. Parte del metamodelo con los elementos que intervienen en la composición de servicios

Los *servicios compuestos*, o la definición de nuevos servicios a partir de la coordinación de servicios simples más propiamente, pueden ser de dos tipos: orquestaciones y coreografías. Las *orquestaciones*, desde el punto de vista de un modelo arquitectónico de servicios, necesariamente incluirán un servicio de tipo soporte que actúe según el rol de orquestador y que implemente un flujo de trabajo basado en la invocación de operaciones del resto de servicios del sistema. Las *coreografías*, por el contrario, no necesitan de un servicio de soporte especial y se estudiarán más en detalle en la sección siguiente.

3 Modelado Arquitectónico de Coreografías de Servicios

En esta sección exponemos nuestra visión de modelado de coreografías orientadas a servicios usando la metáfora del *bloqueo directo* [22] como ejemplo de estrategia básica de coordinación.

3.1 Descripción de la metáfora de coordinación

La estrategia coordinación que se utilizará como medio para explicar el modelado de coreografías será una situación táctica denominada *bloqueo directo*. Para ello, primero debemos de definir una serie de factores que nos conducirán a una mejor comprensión de esta acción ofensiva.

Partimos de una situación de juego dos-contra-dos (2c2), donde el uso de *bloqueo directo* es una de las distintas opciones de ataque que conFiguran nuestro sistema de juego ofensivo. Wissel [33] determina como en el bloqueo directo se verán implicados cuatro jugadores (dos atacantes y sus defensores), donde los dos primeros interaccionaran con el objetivo de librar a cada uno de ellos de su defensor mediante una obstrucción legal de su movimiento, así el bloqueador *irá a buscar* al defensor que marca al jugador con balón para interrumpir su desplazamiento, dejándole momentáneamente libre de su defensa.

Los roles que se derivan de esta situación táctica son cuatro: *Atacante* o playmaker (atacante con balón, AB), compañero o *bloqueador* (atacante sin balón, ASB), bloqueado o *defensor* (defensor del jugador con balón, DB) y *ayudante* o defensor del bloqueador (defensor del jugador sin balón, DSB). Cada uno de estos roles estará compuesto de una serie de acciones propias y específicas a cada uno de ellos [33]. La tabla 1 muestra algunas de ellas.

Tabla 1. Acciones asociadas a cada uno de los jugadores participantes en el bloqueo directo.

Acciones tácticas ofensivas		**Acciones tácticas defensivas**	
Atacante con balón (AB)	*Bloqueador (ASB)*	*Defensor (DB)*	*Ayudante (DSB)*
Parar y tirar	Bloquear	Negar bloqueo	Flash defensivo corto
Penetrar	Ganar posición en continuación	Pasar de 2º	Flash defensivo largo
Pasar a la continuación	Apoyar	Cambio	Cambio
Aclarado	Desmarcarse	Jugar 2c1	Jugar 2c1

Al igual que en otros ámbitos de coordinación basados en coreografías, la aplicación del bloqueo directo en el transcurso de un ataque no será el resultado de la intervención directa y predefinida de uno de los participantes en el sistema, que actúe, en un momento puntual, como coordinador principal (en este ejemplo: un entrenador dando las indicaciones de las tareas a realizar por parte de los jugadores en el terreno de juego), sino que éste viene dado gracias a interrelación que existe entre los diferentes elementos que lo componen y que buscan el mismo objetivo (atacante y compañero buscan el tanto, mientras que defensor y ayudante intentan evitarlo). Además, cada uno de los participantes jamás actuará de igual manera en las mismas jugadas, sino que sus acciones estarán determinadas por las acciones, decisiones o movimientos del resto de los participantes de la coreografía. Por ejemplo, ante la misma jugada, el jugador con balón optará por: pasar a su compañero si se ha desmarcado con claridad después del bloquear, lanzar a canasta si su defensor ha quedado perfectamente bloqueado, realizar un re-bloqueo en caso de no haber conseguido tanto mi compañero como yo no la ventaja necesaria, etc.

A este respecto Ociepka [22] alude al uso de las distintas posibilidades ofensivas en el bloqueo directo en función de las distintas variables que se presenten (tipo de defensa del equipo contrario, características de los atacantes, marcador ajustado, etc.) para plantear gran cantidad de problemas a la defensa y encontrar la solución adecuada.

3.2 Representación de los elementos arquitectónicos de una coreografía

Como se ha comentado en apartados anteriores, nuestra propuesta de desarrollo de soluciones de integración basada en servicios y centrada en la representación de estrategias de coordinación mediante modelos, se realiza dentro de la metodología SOD-M. Esta metodología utiliza UML como notación para los modelos que define, y, por lo tanto, las coreografías de servicios se representan mediante alguno de los conceptos incluidos en la versión 2.0 de UML [23].

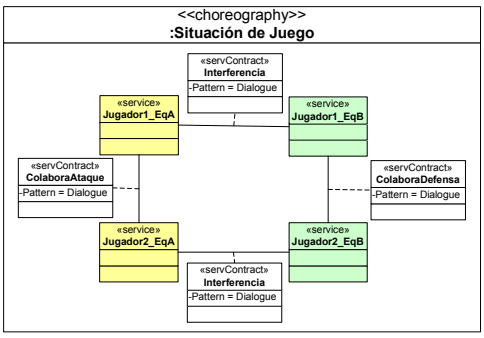


Fig. 2. Modelización del servicio compuesto.

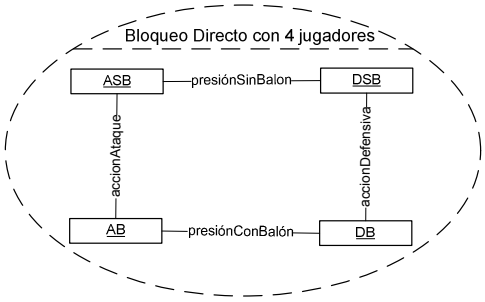


Fig. 3. Modelado de roles como una colaboración para una situación táctica de bloqueo directo.

En nuestro metamodelo, todos los tipos de servicios heredan de *StructuredClassifier* (que a su vez hereda de *Classifier*). Puesto que los servicios compuestos son un tipo de servicio, es posible representar cualquier tipo de coordinación dentro de nuestro modelo. En nuestra metáfora deportiva entendemos como *esquema de coordinación* aquél que representa una situación de juego en la que participan diferentes jugadores. Para nosotros, cada uno de los jugadores será una instancia del tipo de servicio jugador. La situación de juego será representada, por lo tanto, como un servicio compuesto que involucra a distintos *servicios simples* (jugadores) (ver Figura 2).

Cuando un servicio participa en una coreografía debe hacerlo actuando según un rol determinado. En UML 2.0 los roles se representan como elementos conectables (*ConnectableElement*) y así será como se representan dentro del metamodelo de la arquitectura de servicios de nivel PIM (ver Figura 1). Cuando un servicio actúa siguiendo un rol determinado, tendrá asociado un subconjunto de las operaciones que ofrece. Es decir, todo servicio puede realizar un conjunto determinado de operaciones, pero sólo podrá ejecutar, dentro de la coreografía en la que toma parte, algunas de ellas de acuerdo al rol que tenga asignado en ese momento. Cada *ConnectableElement* admite la definición de operaciones como parte de su descripción. En nuestro caso, las operaciones de este tipo de elemento se corresponderán con las operaciones que puede realizar un servicio cuando actúa según un rol determinado. En este punto se puede observar cómo se separa, por un lado, la definición de los servicios (instancias de un tipo de servicio representando a cada jugador) junto a los contratos establecidos entre ellos, de los roles que juegan en cada momento de la situación táctica. De esta forma, podemos redefinir las coreografías como el intercambio de mensajes que se produce como consecuencia de la interacción de los roles jugados por cada instancia de servicio. Estos mensajes provendrán de la invocación de las operaciones definidas en cada rol y de las acciones que éstos realizan en consecuencia.

En UML 2 las relaciones entre los roles de cada elemento conectable se definen a través de una colaboración (*Collaboration*). Siguiendo con la metáfora deportiva, para nosotros, cada una de estos diagramas de colaboración representará una situación táctica concreta en la que cada jugador tiene asignado un rol determinado (ver Figura 3). El conjunto de todos los diagramas de colaboración será lo que se considere como un sistema táctico completo. Cada situación táctica tendrá asociado un conjunto de variantes de evolución del juego que serán representadas con sus correspondientes diagramas de secuencia en los que se representan las invocaciones de las operaciones definidas para cada rol.

Estas dos descripciones independientes se relacionan del modo previsto en UML 2.0, mediante una correspondencia definida en un *CollaborationUse*, tal como se explica en la sección 3.4.

El número de combinaciones tácticas posibles para una misma situación de juego es bastante elevado por lo que representarlas utilizando lenguajes de bajo nivel no es trivial ni sencillo. Con la metáfora deportiva es posible comprobar cómo el enfoque dirigido por modelos es de gran ayuda a la hora de desarrollar soluciones software en las que sea necesaria la implementación de estrategias de coordinación basadas en coreografías. El siguiente apartado se centra en analizar la comunicación entre los diferentes roles involucrados en la coreografía.

3.3 Formas de comunicación dentro de una coreografía

Una vez que se han representado los elementos que intervendrán en la coreografía, es necesario definir cómo se comunicarán y relacionarán los diferentes servicios según sus roles. Existe toda una serie de posibles variantes para el esquema comunicación; en nuestro caso, éstas se pueden resumir de modo general en cuatro posibles patrones arquitectónicos, que identifican cuatro formas distintas de implementar la interacción entre los servicios. Así, puede decirse que la comunicación utiliza un *monitor*, paso de mensajes de tipo multidifusión (*broadcast*), un esquema de pizarra como estructura de memoria compartida, o bien mediante comunicación directa.

En el caso de utilizar un esquema basado en un *monitor*, cada vez que algún servicio ejecuta alguna operación, será el monitor el que detecte el cambio de rol que se produce o el que perciba el mensaje de invocación dirigido al servicio que cambia de rol. En este entorno, el monitor es un servicio especial que conoce el rol actual de cada uno de los participantes en la coreografía y cuál es la última operación que se ha ejecutado (y quién la ha ejecutado, por supuesto). Periódicamente, los participantes de la coreografía preguntan al monitor por la situación de la misma; y, en función de esta información, pueden actuar en consecuencia. Como se ha comentado, esta opción necesita de la existencia de un *servicio de soporte* que se sitúa como elemento central de la coreografía y que conoce en todo momento cuál es el estado global del sistema de coordinación. El concepto de coreografía en este caso está desvirtuado por la existencia de este elemento, que establecerá contratos con todos los servicios involucrados. Debido a esta circunstancia, la coreografía se transformaría en una orquestación.

En el caso de utilizar comunicaciones de tipo multidifusión (*broadcast*), cada vez que un servicio ejecuta una operación debe notificarlo al resto de participantes de la coreografía, a través de la invocación de una operación especial definida para todos los servicios. En este entorno se mandarán tantos mensajes como roles contenga la coreografía, de tal forma que todos estén conectados con todos. Esta solución no encaja bien con el concepto de comunicación punto a punto de los servicios pero sí con el hecho de que exista una estructura común que todos los servicios utilizan para comunicarse, es decir, nos enfrentaríamos a una integración basada en el concepto de ESB (*Enterprise Service Bus*) [14].

La tercera opción puede considerarse como una mezcla de las dos anteriores. La utilización de una *pizarra* o estructura compartida por todos los servicios en la que se reflejan, tanto las comunicaciones que se suceden entre dos servicios, como el rol de cada uno de los servicios en cada momento. En un entorno como el de la orientación a servicios, en el que cualquier elemento computacional o recurso se entiende como un servicio, la pizarra también se entendería como un servicio más. En tal caso, volveríamos a enfrentarnos al diseño de estrategias de coordinación basadas en orquestaciones.

Finalmente, existe la posibilidad de reflejar, en el modelo arquitectónico, las comunicaciones directas que se producen entre los distintos roles, en el que cada uno de ellos notifica cada una de las operaciones realizadas a aquellos servicios interesados (suscriptores) en conocer los cambios de roles que sucedan, de una manera análoga a la utilizada en el patrón *Observador*. Esta situación se modelará mediante diagramas de secuencia en los que se representan los roles involucrados y los mensajes intercambiados (invocación de operaciones). Esta opción es, por tanto, la única que describe una verdadera coreografía, y es la que se considera en este trabajo.

3.4 Modificación de los roles de los servicios dentro de una coreografía

Como se ha comentado en apartados anteriores, la situación de juego se modela mediante un servicio compuesto de tipo coreografía. La funcionalidad de este servicio dependerá de la táctica representada como interacción entre roles. La asignación de roles a servicios se modela mediante un *CollaborationUse* de UML 2.0 (no mostrado aquí por motivos de espacio). Cada uno de los servicios irá cambiando de rol a medida que la táctica evolucione, es decir, a medida que los

servicios interactúen entre sí. Esta interacción se produce, en entornos como el del baloncesto, por el cambio de dueño del balón en cada instante del partido.

Cada una de las interacciones se producirá por la ejecución del proceso definido en cada rol que conllevará, primero, la invocación de las operaciones de otro rol (por ejemplo, el atacante sin balón solicitará/invocará el *pase de balón* del servicio con el rol de atacante con balón) y, segundo, que como consecuencia de esta acción, los servicios cambien de rol.

El cambio de rol no implica la creación de nuevos contratos de servicio puesto que ya se contemplan todas las posibles combinaciones entre servicios cuando se modela el servicio compuesto. Lo único que se modificará será el conjunto de acciones que podrá realizar un servicio debido a su cambio de rol y por lo tanto el diagrama que representa el *CollaborationUse*.

4 Trabajos Relacionados

En esta sección se analizan algunas de las propuestas encontradas en la bibliografía acerca del modelado de coreografías. La gran mayoría de ellas se centran en el desarrollo de soluciones basadas en tecnologías, lenguajes y estándares concretos de coordinación de servicios. Así, podemos encontrar propuestas orientadas al desarrollo de coreografías con información semántica como la de Paolucci et al. , centrada en el lenguaje DAML-S; otras encaminadas a diseñar coreografías de procesos de negocio basadas en la notación BPMN mediante el lenguaje BPEL como ocurre en [8] con BPEL4Cor y otras basadas en los lenguajes específicos para coreografías WS-Coordination y WS-CF [9] o WS-CAF [20].

Con respecto a aquellas propuestas que utilizan un enfoque de más alto nivel, encontramos propuestas que utilizan notaciones particulares como la de Barros et al. [4], que abordan el modelado de coreografías desde diferentes vistas de la arquitectura e incluso iniciativas de representación de coreografías mediante autómatas como la de Mitra et al. [18].

De aquellas propuestas que utilizan un enfoque dirigido por modelos, podemos destacar las propuestas de Gonczy et al. [10] que proponen un metamodelo en el que se incluye el concepto de composición de servicios pero no el de coreografía u orquestación. Heckel et al. [11] y Baresi et al. [3] parten de la propuesta anterior para diseñar una alternativa basada en la propuesta de MDA pero sin que, de nuevo, se tenga en cuenta la representación de coreografías. Esta situación se vuelve a repetir en las propuestas de Autili et al. [2] y Colombo et al. [6]

Muchas metodologías de desarrollo únicamente describen la colaboración o composición entre servicios como orquestaciones. Entre ellas podemos mencionar la iniciativa de IBM [13] que propone un perfil UML para la representación de arquitecturas de servicios. Este enfoque define el concepto de colaboración entre servicios como forma de representar la implementación de coordinación entre servicios, pero mediante BPEL4WS [1].

Otras propuestas de integración de aplicaciones orientadas a servicios utilizan, como núcleo de la coordinación, diferentes plataformas de envío centradas en el paso de mensajes entre servicios. Este tipo de estrategia de integración hace que, por un lado, se dependa de la tecnología de implementación de la plataforma; y, por otro lado, que la plataforma en sí se considere como un elemento director de la coordinación. La existencia de este elemento restringe los grados de libertad que se le puede otorgar a los servicios que participan en la coordinación. En estos casos hablamos principalmente de orquestaciones más que de coreografías. Entre las propuestas que utilizan este enfoque podemos destacar la de Wada et al. [32] que modelan las coreografías siguiendo una arquitectura de filtros y tuberías.

De entre las pocas propuestas que contemplan el concepto de rol dentro del modelado de la coordinación entre servicios, podemos mencionar las iniciativas de Krüger et al. [15] donde se utiliza el concepto de rol desde el punto de vista del paradigma de componentes, o la propuesta de Zhang et al. [34] que también se centra en la definición de roles como parte de los componentes de servicio. En ambos casos no se hace ninguna referencia al concepto de coreografía.

Como se puede comprobar, hasta donde nuestro conocimiento alcanza, no existe prácticamente ningún trabajo que englobe, en una misma propuesta, el desarrollo dirigido por modelos de un sistema orientado a servicios en el que se pueda representar la coordinación entre servicios. Tampoco existen propuestas que soporten, de forma simultánea el modelado de coreografías y orquestaciones como formas diferentes de coordinación.

5 Conclusiones y Líneas de Investigación Abiertas

En este trabajo se ha presentado una propuesta de modelado, a nivel arquitectónico, de coreografías de servicios como estrategia de integración de aplicaciones.

Se ha partido de una estrategia de coordinación real que, utilizada como metáfora válida para la simulación de coordinación de servicios, demuestra que resulta inviable y poco práctico, mediante las técnicas de implementación a bajo nivel actuales, representar en toda su envergadura un sistema táctico complejo. Por ello, se ha propuesto y explicado cómo un enfoque dirigido por modelos puede resolver el problema de la representación de soluciones de integración basadas en servicios. Además, este enfoque se engloba dentro de una metodología de desarrollo más genérica que sigue los principios de la propuesta de MDA, con lo que el diseño de la solución orientada a servicios se desarrolla a través de modelos separados en distintos niveles de abstracción, con la ventaja de poder definir reglas de transformación automáticas entre cada uno de estos modelos. Este último aspecto es objeto de una investigación que actualmente está en progreso y que, por motivos de espacio, no ha sido presentada en este trabajo.

La utilización de una metáfora deportiva nos sirve además para profundizar en la forma en que debe ser definida una arquitectura orientada a servicios. El objetivo es que al tratar los servicios como entidades fundamentales de construcción de soluciones software compuestas, además de las relaciones establecidas entre ellos, resulta posible identificar los elementos que han de definirse en el resto de los modelos, dentro de una metodología orientada a servicios como SOD-M.

Cabe remarcar que en este artículo no se ha explicado en detalle (ni ejemplificado) el modelado del intercambio de mensajes que sucede como consecuencia de la coreografía y su aplicación al bloqueo directo como estrategia de juego. La descripción detallada de los diagramas de secuencia y colaboración, así como el proceso de evolución de los roles de cada servicio, se ha dejado, por motivos fundamentalmente de espacio, para otros trabajos.

6 Agradecimientos

Este trabajo se ha llevado a cabo dentro de los proyectos GOLD (TIN2005-00010) financiado por el Ministerio de Ciencia y Tecnología y el proyecto IASOMM (URJC-CM-2007-CET-1555) cofinanciado por la Universidad Rey Juan Carlos y la Comunidad Autónoma de Madrid, y el proyecto *Agreement Technologies* (CONSOLIDER CSD2007-0022, INGENIO 2010).

Bibliografía

[1] Andrews T., Curbera F., Dholakia H. et al. *Business Process Execution Language for Web Services, Version 1.1 Specification*. BEA Systems, IBM, Microsoft Corp., SAP AG, Siebel Systems, (2003).

[2] Autili M., Cortellessa V., Di Marco M. and Inverardi P. A Conceptual Model for Adaptable Context-aware Services. In Proc. of *Web Services Modeling and Testing)*, Palermo, Sicily, Italy, June 2006

[3] Baresi L., Heckel R., Thone S., and Varro D. Modeling and validation of service-oriented architectures: Application vs. style. *In Proc. ESEC/FSE 2003*, Helsinki, Finland, September 2003

[4] Barros, A., Decker, G., Dumas, M. Multi-staged and Multi-viewpoint Service Choreography Modelling. *Technical Report 4668*, Queensland University of Technology, Brisbane, Australia, 2006.

[5] BPMI Notation Working Group. *Business Process Modeling Notation (BPMN) Version 1.0*. Disponible en http://www.bpmi.org/

[6] Colombo M., Di Nitto E., Di Penta M. et al. Speaking a Common Language: A Conceptual Model for Describing Service-Oriented Systems. *In ICSOC'05*, Amsterdam, the Netherlands, December 2005.

[7] De Castro V., E. Marcos, M. López Sanz. A Model Driven Method for Service Composition Modeling: A Case Study. *Intl Journal of Web Engineering and Technology*. Vol. 2, No. 2, Pp.: 335-353, 2006.

[8] Decker G., Kopp O, Leymann F., Pfitzner K., Weske M: Modeling Service Choreographies Using BPMN and BPEL4Chor. *CAiSE 2008*: 79-93

[9] Fujitsu Arjuna. *WS-CF: WS-Coordination Framework*, http://developers.sun.com/techtopics/webservices/wscaf/wscf.pdf, Oracle IONA, Sun. 2003

[10] Gönczy L., and Varró D. Modeling of Reliable Messaging in Service Oriented Architectures, In Proc. Intern. *Workshop on Web Service Modeling and Testing (WS-MATE 2006)*

[11] Heckel R., Lohmann M., Thöne S.: Towards a UML Profile for Service-Oriented Architectures, *Workshop on MDA: Foundations and Applications (MDAFA'03)*. Enschede, June 2003.

[12] Jerome Josephraj. Web services choreography in practice. *IBM DeveloperWorks Site, mayo 2005*. Obtenido de http://www.ibm.com/developerworks/xml/library/ws-choreography/index.html

[13] Johnston S. UML profile for software services. *IBM DeveloperWorks Site*, April 2005. Obtenido de: http://www-128.ibm.com/developerworks/rational/library/05/419_soa/

[14] Krafzig, D.; Banke K., Slama D. *Enterprise SOA Service Oriented Architecture Best Practices.* Ed. Upper Saddle River: Prentice Hall PTR, 2004.

[15] Krüger I. H., Mathew R.: Systematic Development and Exploration of Service-Oriented Software Architectures. *In Proc. of WICSA'04*, pp. 177-187. Oslo, Norway. IEEE/IFIP, 2004.

[16] López-Sanz, M., Acuña, C., Cuesta, C. Marcos, E. Defining Service-Oriented Software Architecture Models for a MDA-based Development Process at the PIM level. *Proc. of WICSA'08*, IEEE/IFIP, 2008.

[17] Marcos E., Acuña C. J., Cuesta C. E. Integrating Software Architecture into a MDA Framework. *In Proc. of EWSA'06*, pp: 127-143. Nantes, France.

[18] Mitra S., Kumar R. and Basu S. Automated Choreographer Synthesis for Web Services Composition Using I/O Automata. *IEEE International Conference on Web Services (ICWS)* 2007.

[19] OASIS. *Reference Model for Service Oriented Architecture. Committee draft 1.0*. Obtenido de: http://www.oasis-open.org/committees/download.php/16587/wd-soa-rm-cd1ED.pdf

[20] OASIS. *Web Services Composite Application Framework Technical Committee*, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-caf

[21] OASIS. *Web Services Coordination (WS-Coordination) Version 1.1. April 2007*. Obtenido de http://docs.oasis-open.org/ws-tx/wstx-wscoor-1.1-spec-os/wstx-wscoor-1.1-spec-os.html

[22] Ociepka, Bob. Defending the pick and roll. *FIBA assist magazine*. Nov. – Dec. 2004, 31 - 34.

[23] OMG. Model Driven Architecture. Eds.: Miller, J., Mukerji, J. Document No. ormsc/2001-07-01. Obtenido de: http://www.omg.com/mda

[24] OMG. *UML 2.0 Superstructure Specification*, http://www.omg.org/docs/ptc/03-08-02.pdf. August 2003

[25] Paolucci M., Srinivasan N., Sycara K., and Nishimura T. Toward a Semantic Choreography of Web Services: From WSDL to DAML-S. *In Proc. of ICWS'03*. Las Vegas, Nevada, USA, 2003, pp 22-26.

[26] Papazoglou M. P. Service-Oriented Computing: Concepts, Characteristics and Directions. *In Proc. of WISE'03*. pp. 3-12. Roma, Italy, December 10-12, 2003.

[27] Papazoglou M. P. What's in a Service? *In Proc. of ECSA'07*: pp.11-28

[28] Peltz C. Web Services Orchestration and Choreography. *Computer*, vol.36 [10], pp. 46-52, Oct., 2003

[29] Remmert, H. Analysis of group-tactical offensive behavior in elite basketball on the basis of a process orientated model. *European Journal of Sport Science*, vol 3, issue 3, June 2003, pp. 1-12(11)

[30] Romay, P., Cuesta, C., López-Sanz, M., Orientación a Aspectos en UML2 sin Extensiones. *Revista Española de Innovación, Calidad e Ingeniería del Software*, 4(1), 23-49, abril 2008.

[31] W3C. *Web Services Choreography Description Language Version 1.0. W3C Working Draft 17 December 2004*. Obtenido de http://www.w3.org/TR/2004/WD-ws-cdl-10-20041217/

[32] Wada H., Suzuki J. and Oba K., Modeling Non-Functional Aspects in Service Oriented Architecture. In *Proc. of the 2006 IEEE International Conference on Service Computing*, Chicago, IL, September 2006

[33] Wissel, H. *Baloncesto, aprender y progresar*. Ed. Paidotribo. Barcelona, 1996.

[34] Zhang T., Ying S., Cao S., Jia X. A Modeling Framework for Service-Oriented Architecture. *Proceedings of the Sixth International Conference on Quality Software (QSIC 2006)*, pp. 219-226.

[35] Zimmermann, O., Doubrovski, V., Grundler, J., and Hogg, K. Service-oriented architecture and business process choreography in an order management scenario: rationale, concepts, lessons learned. In Companion To the *20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*. OOPSLA '05. ACM, New York, NY, 301-312.

Advances in a DSL for Application Integration*

Rafael Z. Frantz[1], Rafael Corchuelo[2], Jesús González[3]

[1] Universidade Regional do Noroeste do Estado do Rio Grande do Sul (Unijuí)
São Francisco, 501. Ijuí 98700-000RS (Brasil)
rzfrantz@unijui.edu.br
[2] Universidad de Sevilla, ETSI Informática
Avda. Reina Mercedes, s/n. Sevilla 41012
corchu@us.es
[3] Intelligent Dialogue Systems, S.L. (INDISYS)
Edificio CREA, Avda. José Galán Merino, s/n. Sevilla 41015
j.gonzalez@indisys.es

Abstract. Enterprise Application Integration (EAI) is currently one of the big challenges for Software Engineering. According to a recent report, for each dollar spent on developing an application, companies usually spend from 5 to 20 dollars to integrate it. In this paper, we propose a Domain Specific Language (DSL) for designing application integration solutions. It builds on our experience on two real-world integration projects.

1 Introduction

Nowadays, many companies run a large number of applications in a distributed environment to carry out their businesses. These applications are often software packages purchased from third parties, specially tailored to solve a specific problem, or legacy systems. In this environment often a business process has to be supported by two or more applications. In our experience, it is common that these applications are not prepared to interact among themselves. This usually happens when at least one application that is part of the process was not designed taking integration into account. Worse than that, it is very common that the applications have been developed using very different technologies and platforms. Such systems are commonly referred to as software eco-systems [10].

In such software eco-systems it is common that entering and carrying data from an application to another and executing functionalities is user's responsibility. Also, is very frequent the need to add new features to the existing applications, which may be prohibitive. So, in this case, there are two possibilities: to develop a new application with all the current functions and then add the new desired functions to it, or to develop another application only with the new features and integrate them all. The first choice is usually very expensive; the second requires designing an integration solution that should provide the user with a high-level view of the problem. According to a recent report by IBM, for each dollar spent with the development of an application, the cost to integrate it is from 5 to 20 times more expensive [15]. These figures make it clear that integrating business applications is quite a serious challenge.

* Partially funded by the Spanish National R&D&I Plan under grant TIN2007-64119, and the Andalusian Local Government under grant P07-TIC-02602. The work by R.Z. Frantz was funded by the Evangelischer Entwicklungsdienst e.V. (EED). We are deeply indebted to Abdul Sultán from Sytia Informática, S.L., Hassan A. Sleiman, Raúl Sánchez, and Francisco J. Dominguez from Indevia Solutions, S.L.L., for their extremely helpful collaboration to implement the DSL described in this paper.

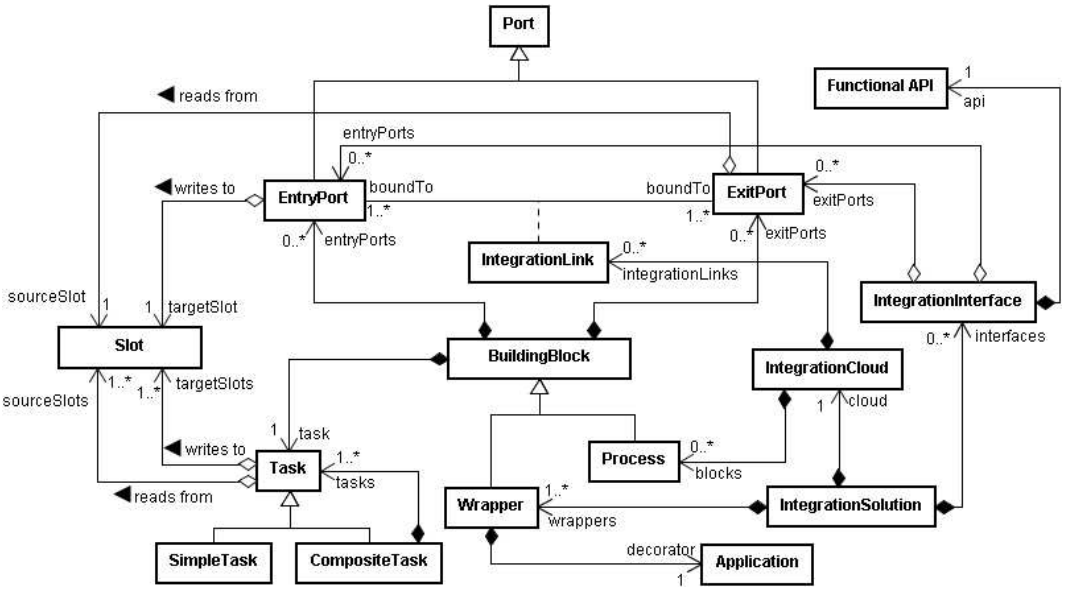


Fig. 1. Core elements of the DSL model.

When considering integration, we must consider some constraints so that a solution is viable for a company [9]. The first constraint is that integrated applications should not change, since a change might seriously affect or even break down other business processes. The other constraint is that, after integrated, the applications should be kept decoupled, as they were before. The integration solution should not change the applications and introduce dependencies that did not exist before, but just coordinate them in an exogenous way by means of building blocks. There is another constraint: integration must be performed on demand, as new business requirements emerge and require new services to be created building on the existing applications [2].

Enterprise Service Buses (ESBs) range among the most usual tools used to devise and implement integration solutions [4,5,6,12,17]. Such tools commonly rely on the well-known Pipes&Filters architectural pattern [7], according to which an integration solution is designed as a flow of messages through a series of filters that communicate by means of pipes. It is not surprising then that there are many proposals for Domain Specific Languages (DSLs) that help engineers to design both pipes and filters, within the context of ESBs. In this paper, we introduce a DSL to design filters that builds on our conclusions working on the design of two real-world projects in quite different scenarios. The remainder is organized as follows: Section 2 introduces the core elements of our DSL model; Section 3 reports two real-world integration scenarios used to validate our proposal; Section 4 compares our proposal to others; finally, Section 5 presents our conclusions.

2 The DSL Model

In Figure 1, we present the main ingredients of our DSL and their relationships. Roughly speaking, an integration solution comprises at least one flow that integrates one application with another; we refer to such flows as integration flows. They are responsible for transporting messages, but can also translate, enrich, filter or route them. Flows are built from four elements: processes, tasks, ports and slots. Processes and tasks are considered process units in the flows; contrarily, port and slots connection units.

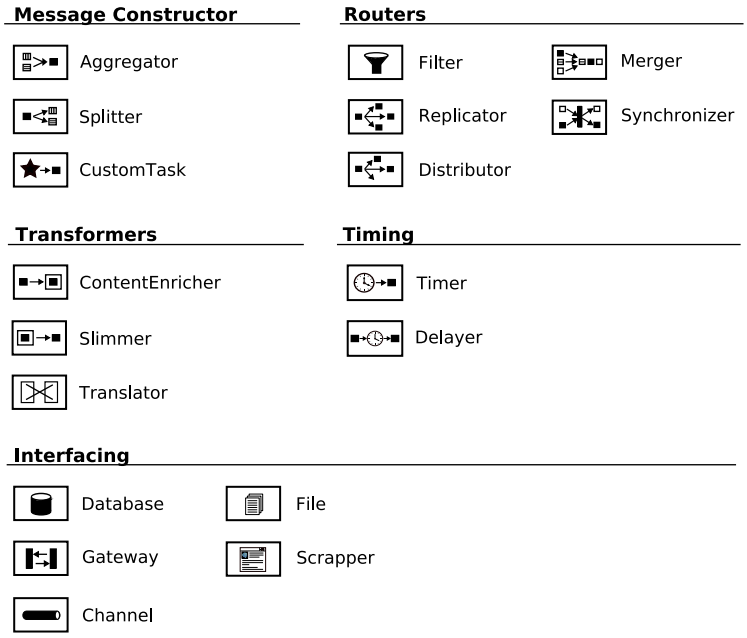


Fig. 2. Task taxonomy.

Below, we provide a short description of the main elements of our DSL.

Building Blocks: This is one of the most important elements in the model, since it represents a general construction block used to design an integration solution. A building block has, exactly, one task, that is a simple integration task (simple task) or a more complex task (composite task). Building blocks can be either wrappers or processes. Wrappers are used to connect an application to an integration solution. In our DSL model, a wrapper is a building block, with its internal task(s) used to access the application being integrated, plus a decorator. A decorator is composed of an icon of an application and a glyph to represent what the layer being integrated is. Processes allow to implement integration-specific services across a flow. They contain one or more tasks and may connect to other processes or wrappers through ports and integration links.

Tasks: A task is the element responsible for performing a simple, atomic step within a building block. Note that every process and wrapper are composed of exactly one task, which is commonly a composite task that has other inner tasks. A task reads a message from a slot, processes it, and writes the result to the next slot.

Slots: Slots are used inside building blocks to allow exchanging messages between ports and tasks, and also between tasks. Essentially slots are in-memory buffers that allow tasks, of which a building block is composed, to communicate asynchronously.

Ports: Building blocks have ports through which they can send/receive message(s) to/from another building block. An entry port writes an inbound message to an internal slot from which a task can read it. On the contrary, an exit port always reads a message from a slot and makes it available to the next port in the flow. Entry ports and exit ports are always bound with one another. It happens, e.g., when a process block is linked to another process or a wrapper. This relation between ports is represented in Figure 1 with an association called integration link.

Our DSL provides five types of tasks, namely: Message constructors, transformers, routers, timing and interfacing tasks, cf. Figure 2 and [9]. Below, we report on them:

Message Constructors: A message constructor creates new messages. There are three important tasks in this group: aggregator, splitter and custom task. An aggregator is an stateful task that can receive two or more inbound messages and combines their individual content in just one message; sometimes it may be interesting when we have different messages with individual results that may be combined to be processed as a whole latter. A splitter is the counterpart of an aggregator; this type of task receives an inbound message and produces two or more outbound messages that will be processed individually. A custom task allows users to create a message using custom code. Sometimes it may be interesting when from an inbound message we want to create a complete different message, like e.g. a message to query a database and latter enrich the original message with the result.

Transformers: Transformers modify the original content of an inbound message. There are three important tasks in this group: content enricher, slimmer and translator. Sometimes, it may be necessary to append more information to a message in order to process it latter on in the flow; a content enricher receives an inbound message and computes a new one based on the original message content or data in an external resource. On the other hand, a slimmer may reduce the message size by removing part of its content. When considering an integration solution, a very frequent task is translating messages from one format to another; it is necessary because applications that are being integrated usually work with different message formats; thus, a translator receives an inbound message, translates it to the new format and send it to the next task.

Routers: Routers are responsible for routing an inbound message to zero, one or more destinations, and here we focus on filter, replicator, distributor, merger and synchronizer. Using a filter we may avoid uninteresting messages from reaching the next task; a filter task receives an inbound message and based on a certain criteria allows this message to continue or removes it from the flow. A replicator task makes copies of an inbound message and sends them to two or more destinations; it does not change the original contents and there is no limit for message copies; however the number of messages must be equal to the number of slots to which the replicator can write; this task type should be used, e.g., to duplicate a message in order to execute a query in another system and latter aggregate the result of this query with the other copy to distribute copies of the original message for two or more applications. A distributor routes an inbound message to zero, one or more destinations. Note that, in the worst case, a distributor may behave exactly as if it was a replicator. Sometimes it may be necessary to merge messages from two or more slots into just one, like e.g. when the next task just can read from a single slot. To perform this, merger task can be used. Finally, when a task needs to receive a group of two or more messages that must follow a given pattern, and this pattern may take time to be fulfilled, a synchronizer task can be used. In this case the synchronizer permanently checks its entry slots and when a message fulfils the pattern, all messages of the group are forwarded. It is common in those cases where a message is used to create another query message to query an external resource and then both messages (the resulting message and the original message) need to be forwarded together to be processed by another task, like e.g. an aggregator task. Note that, the number of entry and exit slots for a synchronizer must be the exactly the same.

Timing: This group includes timers and delayers. A timer is a type of task that we can configure so that it produces an outbound message at fixed times. A delayer, on the contrary, is used to delay delivering a message for a fixed number of seconds; it may be used, for instance, to avoid flooding a process that runs on a slow machine.

Interfacing: To integrate an application we need to design at least one wrapper that must be responsible for reading information from the application and sending it to the integration solution and/or writing information from the integration solution to the appli-

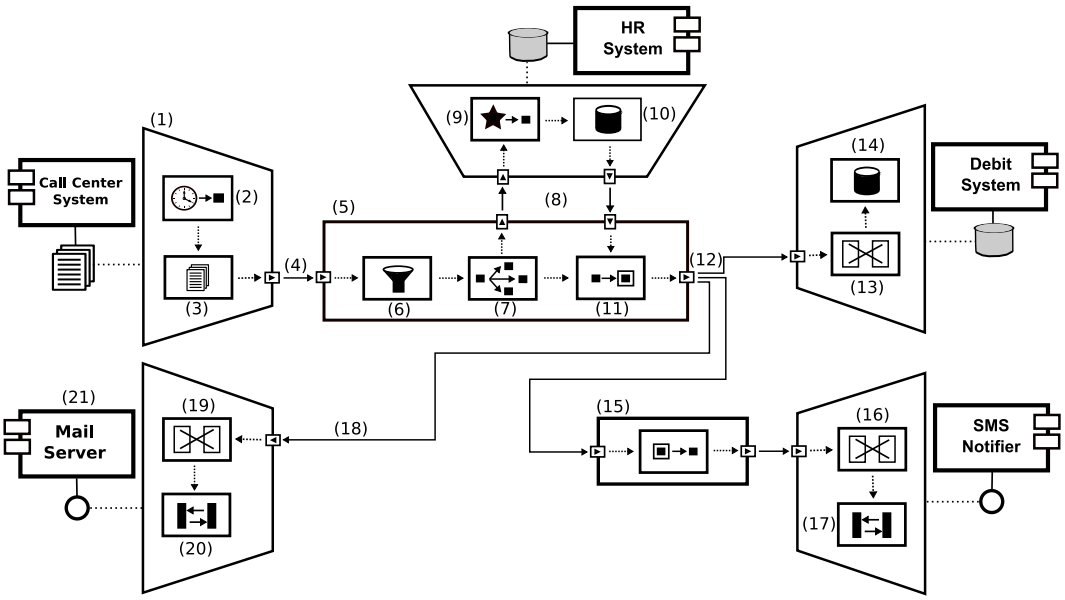


Fig. 3. Unijuí's integration solution.

cation. Actually, a wrapper interfaces a layer of the application; in most of the cases, it is a database, gateway, a messaging channel, file or even the user interface (with a scrapper [1]). Interfacing tasks relieve the programmer from the burden of performing such low level operations.

3 Integration scenarios

We have validated our proposal working on the design of two real-world integration projects. The former provides an effective solution to automate the invoicing of phone calls at Unijuí; the latter provides an intelligent interface to a number of information servers owned by a public institution. Below, we report on the details.

3.1 Call System at Unijuí

Unijuí has five applications involved in a hand-crafted process whose goal is to invoice their employees of the private phone calls they make using the University's phones. Each application runs on a different platform and was designed without integration concerns in mind. There is a Call Center System (CCS) that records every call every employee makes from one of the telephones this university provides to them. Every month, an analysis is performed to find out what calls have a cost and are not related to the work activities of the employees; such calls are debited to the employees by using a Debit System (DS). There is also a Human Resources System (HRS) that provides personal information about the employees, including their names, phone numbers, social security numbers, and so forth. There are two additional systems to send mail or SMS messages. The goal of the integration solution is to automate the invoicing of the calls that an employee makes and are not related to his or her work activities.

Figure 3 depicts our integration solution. The applications are connected to an integration solution through wrappers (1). A wrapper contains tasks to interact with the application's

interface layer (database, gateway, user interface, and so on), send and/or receive messages from the integration solution. The decorator (21) simply indicates which application and interfacing layer are being integrated/accessed.

The integration flow for the solution presented in this example begins in the wrapper of application CCS with a timer task (2). This task creates an activation message every two minutes and sends it, through a slot to the next task, which is a file interfacing task (3). This task is responsible for accessing the files where the application CCS writes the phone calls, creates a message for each call and sends them to the next slot one-by-one. The exit port of this wrapper reads each message from this slot and then sends it to the integration link (4), making it available for the entry port of the central process block (5). This process contains a composite task that starts with a filtering task (6). This task filters out messages that do not have a cost for the university, and allow just toll calls to remain in the flow. Those messages are written to the next slot, and will be read by the next task, a replicator (7). The replicator makes copies of the original message. In this case one copy is sent to the wrapper of the application Human Resource System (HRS) and the other to the next element in the current process. In this integration solution we need to append missing information about the employee to the message, like: name, department, e-mail and mobile phone. This information is in the HRS, that is why it is also integrating our solution. The message copy received by the wrapper of HRS, through the ports (8), will be processed by a custom task (9). This task produces an outbound message that represents a database query to be executed by the database data source (10). After that the content enricher (11) receives the result from the HRS's wrapper and enriches the original message with it. Now the enriched message is sent to the next slot, the one that connects with the exit port (12). This port is also connected to three integration links that allow sending a message copy to DS, SMS and MS.

The first integration flow after the exit port (12) connects the process (5) to the wrapper of the DS application. This wrapper receives the message through its entry port and makes it available for the first internal task of the wrapper, the translator task (13). The translator is responsible for translating the current message format into a new format that the DS can understand, and then immediately writes the message into the slot between the translator and the database data source (14). This task accesses the database of the DS and stores the message.

The second flow connects the same exit port (12) to the other process (15) which have a unique internal task, a slimmer task. A slimmer is responsible for removing some information from the message in order to make it smaller before sending it to the SMS. The SMS is an external application that allows sending messages to mobile phones. In its wrapper, there is a translator (16) that receives the inbound message and translates it into a special format that the SMS can understand. Once the SMS offers a public gateway the interaction can be done by a RPC access (17), that forwards the inbound message.

The last copy of the message goes to the flow (18) that now connects the process (5) with the wrapper of the MS. This wrapper integrates the application allowing the solution to send e-mails with all the details about the employee's call. As in the other wrappers it is important to translate the inbound message into a message format that the MS can understand. This is done using a translator (19) inside the wrapper, just after the entry port. The translated message now goes, through a slot, to the next task, the RPC access (20), and then to the MS.

3.2 Job Advisor for a Public Institution

Indisys is a spin-off of the University of Seville that works on the development of interactive virtual assistants (IVAs). Generally speaking, an IVA is an application that allows a user

Fig. 4. A screenshot of Indisys's interactive virtual assistant.

to talk to a computer system using natural language through a rich web interface. Such applications are starting to sprout out since they help call centers to be more effective; the applications range from answering general user information queries to on-line vending services, e.g., flight reservations, cinema tickets, and so on. Since IVAs are quite a new field from a commercial point of view, their features are quite heterogeneous. Our motivating example builds on the development of an IVA whose user interface is presented in Figure 4, which is a job advisor for a public institution.

The modules this system integrates are as follows:

NLU: This is the Natural Language Understanding module, and it is responsible for translating natural language sentences like "Where can I find a job?" into semantically rich structures that are machine understandable.

NLG: This is the Natural Language Generation, which is complementary to the NLU, that is, it transforms computer structures into natural language sentences.

TTS: This module is responsible for translating textual sentences in natural language into voice.

KM: This is the Knowledge Manager module, which is a facade to external information sources to which the system can connect by means of a plethora or communications adapters, including database drivers, web services, and so on.

NLU, NLG, and TTS are legacy systems, i.e., systems that were not designed together for this project but must be reused as they are; designing new modules that are better prepared to be integrated was considered unaffordable so they have to be reused as is. Modules, NLU and NLG are owned by Indisys, so we can have access to them freely. Contrarily, module TTS was provided by a third party, and it is proprietary. The KM module is being developed by another company.

Figure 5 depicts the complete integration solution, which is composed of process CORE, and five wrappers to the existing modules. CORE is the central process since it is responsible for coordinating the activities of the applications being integrated. It is also the cornerstone of our IVA, which just provides a user interface to this process.

Note that modules KM and TTS provide a programmatic interface; so their wrappers are the simplest ones since they just require a gateway interfacing task to integrate them (1 and 13). The wrapper to the web client is referred to as User Interface in Figure 5. It is responsible for gathering user input by means of a gateway interfacing task; however, producing an output is far more complex since this requires a replicator task (3) it is first

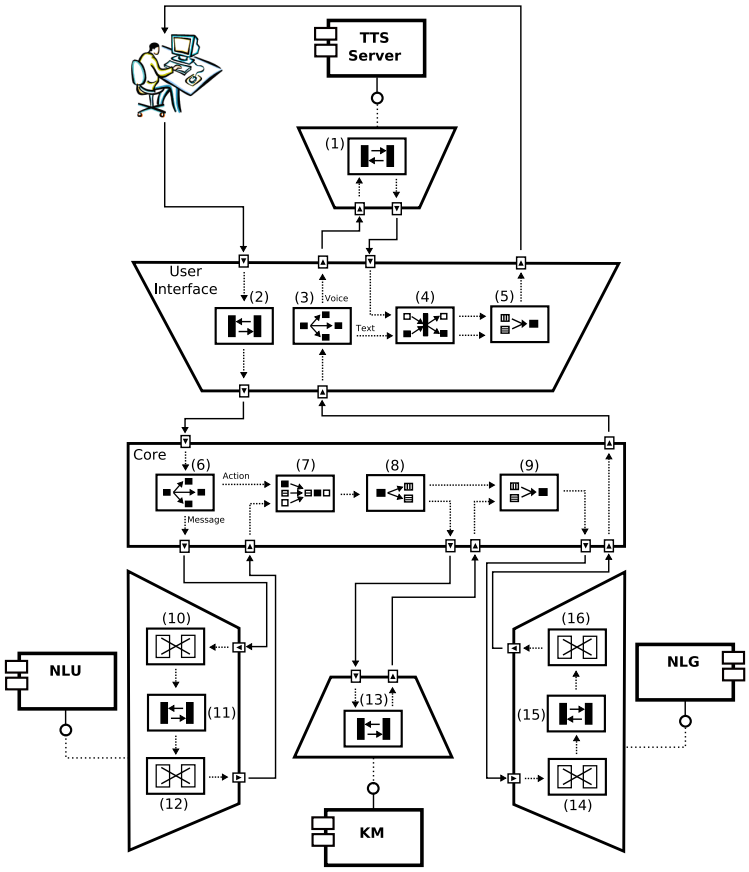


Fig. 5. Indisys's integration solution.

necessary to decide if it is necessary to use the TTS module to synthesise a voice message in response to the client's input (note that the client may switch the voice off, in which case it does not make sense to waste TTS resources on synthesising a stream of voice that is going to be discarded by the client). Next, it is necessary to use synchronizer (4) and an aggregator (5) to merge both the textual and the voice response, if any, before the results are sent to the user.

Process CORE must initially determine if the client's input is a sentence in natural language or just a click on the user interface. It must use the NLU module only in the first case. Again, this is implemented using a replicator (6) that is followed by a merger (7). Later, it is checked if it is necessary to have access to external information sources, in which case it is necessary to separate the internal structure the CORE process handles into a number of messages to request to the KM the information requested by the user. This is accomplished by using a splitter (8). Later, the information returned by the KM module is merged using an aggregator (9). In the end, the whole semantic structured is passed on to the NLG module to generate the appropriate output in natural language.

Last, but not least, the wrappers to NLU and NLG require a number of translators (10, 12, 14, and 16) to deal with a number of problems related to the fact that they are actual legacy modules, in addition to their respective gateway interfacing tasks.

4 Related Work

We compare our proposal to Camel [6], Spring Integration [12], and BizTalk 2006 [17] since they also provide DSLs for building integration filters. Tools such as Mule [5] or ServiceMix [2] are also related, but their focus is on the design of pipes, which is not ours in this paper. We have built a comparison framework out of a number of 47 objective properties that are classified into scope, modelling, and technical properties. Below we describe five properties of each group and also compare these properties with the chosen tools and our proposal.

4.1 Scope Properties

Property	Camel	Spring Int.	BizTalk	Ours
Architectural pattern	Filters	Filters	Pipes/Filters	Pipes/Filters
Context	EAI	EAI	EAI/B2BI	EAI/B2BI
Abstraction level	PSM	PSM	PSM	PIM/PSM
Transactions	ST-F	–	ST-F/LT-F	ST-F/LT-F/ST-S/LT-S
Kind of model	O/D-IoC	O/D-IoC	D-Graph. and XML	D-Graph. and XML

Table 1. Scope properties.

Scope properties (cf. Table 1) represent properties whose absence can greatly hinder or even invalidate a proposal, and to provide them it is necessary to implement extensions whose the cost for developing we believe may be prohibitive in most cases.

Architectural Pattern: As we already know, Pipes&Filters is the standard for excellence in our field of interest. Therefore, it seems reasonable to expect that the tools for building ESBs provide domain specific language for designing both pipelines as filters.

Context: Normally we discern among the following integration contexts: Enterprise Application Integration (EAI), where the emphasis is on integrating applications aiming to synchronize their data or implement new functionalities. Enterprise Information Integration (EII), whose emphasis is on providing a live-view of the data handled by the integrated applications; Extract, Transform, and Load (ETL), which intends to provide materialized views of the data on which we can apply knowledge extraction techniques [16]. In all previous cases, we implicitly assume that the integrated applications belong to the same organization. Lately the activity of integrating applications from different organizations are requiring more attention, aiming to implement inter-organizational business processes; this context is known as Business to Business Integration (B2BI). Also recently, the so-called mashups are requiring attention. They are applications that usually run on a web browser and integrate data from various sources.

Abstraction level: Working with platform independent models (PIM) allows to design stable solutions as independent as it is possible from the technology used to implement them, and its inevitable evolution. By working with PIM models, it is also important to be supported by tools that can transform them into a platform specific model in which we want to run.

Transactions: Transactions allow to design robust solutions capable of facing the failures that may occur during the execution of an integration solution. It is usual to discern

between transactions in the short term (ST) and long term (LT): The former usually is implemented using the known protocol Two-Phase Commit that basically consists of carrying out those actions that require state changes only when all parts involved have confirmed that can carry them out without any problem [11]; the latter, executes the actions when they are possible and, in case some of them fails, then executes compensation actions aiming to undo the effects of the previous actions, or in case that this is not possible anymore, try to lessen their impact. In the context of application integration it is also useful to classify transactions depending on its scope, resulting in filter transactions (F) or solution transactions (S): the first are those which guarantee that a filter achieves its goal, otherwise any change that has been done up to the moment is invalidated; the second are those that ensure this property for the whole integration solution.

Kind of model: According to the kind of model of the tool, it allows to design integration solutions operatively (O) or declaratively (D). When the design is operative, the tool provides a library that developers can use to create their integration solutions; on the contrary, when declarative developers can work at a higher level of abstraction. The first way is by using a domain specific language with graphic or XML representation, that will be automatically translated into executable code; the second is using XML configuration files, which are then interpreted by an Inversion of Control (IoC) [8] engine, e.g., Spring [14].

Our proposal allows for both the design of pipes and filters, although in this paper we have focused on filters only. We also make a clear distinction between platform independent models and platform specific models; the latter are generated automatically by means of a model transformer that builds on DSL Tools [3]. Regarding transactions, we support all types of transactions, which is partially due to the fact that we can track messages and allow for compensation actions to be defined at each building block.

4.2 Modelling Properties

Property	**Camel**	**Spring Int.**	**BizTalk**	**Ours**
Card. of tasks	$1:N$	$1:N$	$1:1$	$N:M$
Card. of locations	$N:M$-Comp	$N:M$-Comp	$1:1$	$N:M$-Comp/Repl
Correlation	No	No	Yes	Yes
Port adapters	$1:1$	$1:1$	$N:1$	$N:M$
Stateful filters	No	No	No	Yes

Table 2. Modelling properties.

The modelling properties (cf. Table 2) are not as critical as the previous ones, and that if lack of them is still possible to design a solution for effective integration at a reasonable cost, but perhaps the design is much more complex and less intuitive. Obviously, this can result in a negative effect on the subsequent maintenance.

Cardinality of tasks: The filters are made up of simpler tasks that allow to build messages, transform them, route them or interact with pipes. In our experience, it is usual that some tasks require contextual information from any external source, to deal with

messages that arrive; a clear example is when a message to be processed includes some identifier and it is necessary to query an application to know which data are related with this identifier.

Cardinality of locations: The term location refers to the physical location on which a pipe is implemented, e.g., a folder in a file system or a mail server. Generally it is interesting to discern between two types of reading: with competition (Comp), in which only one of the filters can read at a time from a particular location or replication (Repl), in which all filters can read at the same time.

Message correlation: Stated that we can not assume synchronicity among the integrated applications, is very common for messages to arrive out of sequence in the filters, so it is the filter's responsibility to correlate them in such a way that those complementary messages always must be processed together. This need is more bounden in cases where it is possible to create filters or tasks with multiple entries.

Port adapters: The possibility of having multiple adapters in a port, allows it to receive/send information from/to two or more locations. The binding of a port with multiple locations helps keeping the model simpler and more intuitive, once that, in the case only one port can be bound to a location, modelling a filter may be more complex.

Stateful filters: A filter may want to store useful information for its next executions. This allows, e.g., storing a list of those last received messages to avoid processing when receiving future messages that are semantically equivalent; in other situations can be used to store results that are costly to calculate, thus avoiding to execute repeatedly the same processing for similar messages.

Our proposal is the most general out of the tools surveyed in this paper, since it allows for multiple input and output filters and tasks, it allows to configure exit ports in both competition and replication modes, and it supports multiple adapters per port. We also provide explicit support for stateful filters.

4.3 Technical Properties

Finally, we present some technical properties (cf. Table 3), as its absence could affect the facility of programming, the performance or the management of solutions integration, also may hinder the deployment and the operation of them.

Property	**Camel**	**Spring Int.**	**BizTalk**	**Ours**
Execution model	1 : 1	1 : 1	1 : 1	$N : M$
Typed messages	No	No	Yes/No	Yes
Communication patterns	Yes*	No	No	Yes
Attachments	No	No	Yes	Yes
Abnormal messages	Yes	Yes	Yes	Yes

* No new MEPs can be defined.

Table 3. Technical properties.

Execution model: The filter's execution model may seriously affect the performance of an integration solution. The simplest model consists of assigning a thread to each message or set of messages that must be treated as a whole by a filter; of course, the threads can

be taken from a pool to keep under control the total workload of the server. We believe that this model has a shortcoming that can damage the scalability of the solutions. The problem is related to the fact that when an instance of a filter reaches a point where it can not continue running tasks, e.g, because it is necessary to wait for a message arrival, the thread is idle for a completely undetermined time. From our experience we concluded that a model capable of running on an asynchronous way multiple instances of the same filter on a pool of threads would be much more effective. Currently we are working and evaluating both alternatives in order to obtain more conclusions.

Typed messages: A typical integration solutions usually performs a lot of message transformations; any error in one of them can lead to an incoherent message, so it is highly desirable that these messages are typed.

Communication pattern: The Message Exchange Pattern (MEPs) allows to define specific communication types [13]. An integration solution can use different types of MEPs. Using MEPs facilitates the correlation between messages that reach a filter and the responses produced. For this reason, from a technical point of view, it is desirable that the tool offers support to this pattern and, besides, allows to define new types of MEPs, in addition to those that are already available.

Messages with attachment: A message can also carry, in addition to the header and body information, other objects with attached information. The attached information should not be processed in the integration solution, in other words, it only represents additional information that is transmitted with the message. We believe that the separating attached information from body, allows to reduce its processing time, once that when a task access the body to process it, will not have to deal with the attaches. Moreover, if the attaches are separated from the body, it is possible to use the ClaimCheck pattern [9] to store the attaches in a persistent repository, while the message is being processed. The attaches can be recovered latter.

Abnormal messages: When a message presents some kind of anomaly that makes it impossible to be processed by a building block, the norm is that it produces an exception and that the message in question is stored so that it can be analyzed by the system administrator. In addition, it is highly desirable that these messages can also be processed automatically so that we can try to carry out some sort of corrective action at the same time in which it was detected.

Regarding technical properties, our proposal is also quite complete since it allows for typed messages, it allows to define arbitrary message exchange patterns, and messages can have attachments to improve efficiency. The asynchronous, multi-threaded execution model is still under evaluation.

5 Conclusions

Application integration is a growing activity in companies and, according to the report published by [15], it is very expensive since it demands much more resources than the regular software development process. Knowing these, it is very important to have engineering technologies (languages, tools, frameworks, and the like) that can support this activity helping to reduce the cost and resources usually spent in. The DSL proposal presented in this paper contributes to helping engineers implement integration solutions with less effort. Our proposal is based on the concept of building block, which allows to design an integration solution visually by working at a higher level of abstraction, creating reusable, well documented and independent of technology/platform solutions. It has been validated in two real-world integration projects in quite different scenarios.

References

1. C. Chang, M. Kayed, M.R. Girgis, and K.F. Shaalan. Survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411–1428, 2006.
2. B.A. Christudas. *Service Oriented Java Business Integration*. Packt Publishing, 2008.
3. S. Cook, G. Jones, S. Kent, and A.C. Wills. *Domain-Specific Development with Visual Studio DSL Tools*. Addison-Wesley, 2007.
4. J. Davies, D. Schorow, and D. Rieber. *The Definitive Guide to SOA: Enterprise Service Bus*. Apress, 2008.
5. P. Delia and A. Borg. *Mule 2: Official Developer's Guide to ESB and Integration Platform*. Apress, 2008.
6. Apache Foundation. Apache Camel home. Available at http://activemq.apache.org/camel.
7. M. Fowler. *Patterns of Enterprise Application Architecture*. Addison–Wesley, 2002.
8. Martin Fowler. Inversion of Control Containers and the Dependency Injection Pattern, 2004.
9. G. Hohpe and B. Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley, 2003.
10. D. Messerschmitt and C. Szyperski. *Software Ecosystem: Understanding an Indispensable Technology and Industry*. MIT Press, 2003.
11. D. Skeen. A formal model of crash recovery in a distributed system. *IEEE Transactions on Software Engineering*, 9(3):219–228, 1983.
12. Inc. SpringSource. Spring integration home. Available at http://www.springframework.org/spring-integration.
13. W3C. Web services message exchange patterns. Available at http://www.w3.org/2002/ws/cg/2/07/meps.html#id2612442.
14. C. Walls and R. Breidenbach. *Spring in Action*. Manning Publications, 2004.
15. J. Weiss. Aligning relationships: Optimizing the value of strategic outsourcing. Technical report, IBM, 2005.
16. I.H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.
17. D. Woolston. *Foundations of BizTalk Server 2006*. Apress, 2007.

Towards a Reference Architecture for Enterprise Mashups

Javier López[1], Alberto Pan[1], Fernando Bellas[1], and Paula Montoto[1]

[1] Information and Communications Technology Department, University of A Coruña
Facultad de Informatica, Campus de Elviña s/n 15071 A Coruña
jmato@udc.es, apan@udc.es, fbellas@udc.es, and pmontoto@udc.es

Abstract. Mashup applications combine pieces of functionality from several existing websites to provide new integrated functionality. Existing mashup tools typically address only some aspects of the problem and use a variety of sometimes overlapping and sometimes complementary approaches that make it difficult to compare them. In this 'work in progress' paper, we present a first draft of one reference architecture aiming to ease this task. We also discuss in more detail two crucial layers of the architecture: the Data Mashup and Widget Assembly layers. For these layers, we identify their main design aspects and we briefly identify and compare the different approaches to address them.

Keywords: enterprise mashups, REST, Web 2.0, reference architecture.

1 Introduction

Mashup applications combine pieces of functionality from several existing websites to provide new integrated functionality. Many research works [5][12][16][18] and industrial tools [4][9][10][13][19][20] have appeared during the last two years to ease the creation of mashups. Nevertheless, existing tools address only some aspects of the problem and use a variety of overlapping and complementary approaches that make it difficult to compare them.

We aim to create a reference model for mashups that can be used to classify the different proposals and, at the same time, can serve as a guide for the development of complete mashup offerings. We are especially interested in mashups in the corporate world (the so-called 'enterprise mashups'), but we think the model will be useful for consumer mashups also. Our objectives are:

- Providing a reference model composed of a set of layers, useful to compare the different mashup creation approaches.
- Comprehensively identifying all the features and design aspects in each layer, discussing the different alternatives proposed to date.
- Study in detail certain crucial design aspects of the reference architecture to provide specific recommendations about them.

In this 'work in progress' paper, we present a first draft of the reference architecture. We also discuss in more detail two crucial layers of the architecture: the Data Mashup and Widget Assembly layers. For these layers, we identify their main design aspects and we briefly identify and compare the different approaches to address them.

The rest of the paper is structured as follows: section 2 overviews the reference model, including a description of each of the layers that compose it. Section 3 illustrates the architecture with a sample mashup, explaining how each layer in the model participates to create the final application. Sections 4 and 5 respectively discuss the Data Mashup and Widget Assembly layers of the architecture. Section 6 discusses related work while section 7 outlines our current and future work, and concludes the paper.

2. Overview of the Reference Architecture

Existing mashup creation tools differ in many aspects. On the one hand, while most of the tools [4][9][10][13][19][20] are server-side tools, there are also tools [5][18] where most of the components run in the client-side (typically, the browser). On the other hand, there are tools [4][13][20] that allow for the creation of 'data mashups' while others are geared to the construction of 'visual mashups' [5]. Data mashup tools allow for the construction of services that combine existing data sources and return unified data. These tools usually also provide basic display capabilities. Tools oriented to the construction of visual mashups allow building a mashup that reuses the Web interface of the sources it combines. And of course, tools also differ in the specific techniques and technologies they use to implement a particular aspect (e.g. composition of sources).

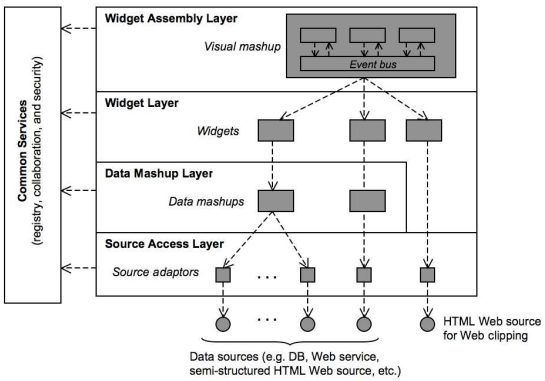


Fig. 1. Reference architecture for mashup creation tools.

Figure 1 shows our proposal for a reference architecture for mashup creation tools. As the figure shows, our proposal breaks down the functionality in 'layers' and 'common services'. Layers represent main functional blocks and have the traditional meaning in software design. Common services represent services that can be useful for several layers. In specific tools, the physical division in layers and services may not be as clear as in the reference architecture. Depending on the tool, some layers, services, or particular aspects may be not provided.

The following sub-sections present an overview of the layers and common services. The description of each layer states its role and gives some general recommendations. When possible, we recommend reusing well-knowing techniques or technologies to support particular aspects.

2.1 Source Access Layer

Most tools assume that sources are queried using the HTTP protocol and return data encoded into XML. In particular, some of them only support the particular case in which sources use either RSS or ATOM data formats. However, a more general mashup creation tool must support access to heterogeneous sources (e.g. databases, Web services, semi-structured HTML Web sources, text files, etc.). To let the higher layer access them in a generic way, the Source Access layer must provide a common interface to access individual sources.

Depending on the source or the intended use of the source in the mashup, this layer can return data or visual markup. For a structured data source (e.g. a database, a Web service, etc.), this layer returns data. For semi-structured HTML Web sources we wish to see as data sources, such as a Web page displaying a list of movies, this layer must navigate to the Web page and extract data. For HTML Web sources used for Web clipping (that is, extraction of a particular block of

markup), this layer must navigate to the Web page and return the markup of the page (particular blocks will be extracted by the Widget layer). Internally, this layer provides an adaptor for each possible type of source. To create the adaptors needed to automatically navigate and extract structured data from Web pages, Web wrapper generation techniques should be used (see for instance [3][15]). All adaptors implement the generic interface.

Since this layer must offer a generic interface to access sources independently of their nature, we propose to reuse the RESTful architectural style [6] for the interface. A RESTful Web service allows accessing a resource by using standard HTTP methods, usually, GET, POST, PUT, and DELETE. Each resource is identified by a global URI. A RESTful interface presents advantages:

- It is generic by nature. For HTML Web sources used for Web clipping, user interactions on the markup translate directly into the interface. For data sources (structured or semi-structured), each individual data item can be modeled as a resource (e.g. http://movies.acme.com/JoelCoen/OBrother) which is read, updated, inserted or removed by using GET, POST, PUT, and DELETE operations. Special URIs (e.g. http://movies.acme.com?director=JoelCoen) can be constructed for executing complex queries by using the GET method.
- It promotes reuse at the resource (data item) granularity. Since each resource is identified by a global URI, any resource can link to another one just by including its URI. For example, the information returned by a movie information service could include, for each movie, a URI (e.g. http://moviecritics.acme.com/JoelCoen/OBrother) pointing to a service providing the critics of such a movie. It also allows the output of a data source to be more lightweight. For instance, instead of returning the full data for each item, the source can return a link for each item to obtain additional information, and let the higher layer decide in an item-by item basis whether to pay the cost of obtaining the detail information or not.
- It can be described in WADL [17], which is a language for specifying the methods it supports, the parameters they require, and the schema of the returned data. This meta-information allows higher layers to know the capabilities of the underlying Web source.
- It can be accessed by standard HTTP methods, letting intermediaries (proxies) to provide additional services (e.g. security, caching, etc.).

2.2 Data Mashup Layer

This layer allows the construction of components, which we call ‘data mashups’, which access one or several data sources though the Source Access layer interface and return integrated data. For example, a data mashup accessing a movie information service and a movie critic’s information service would provide information about movies with their corresponding critics. For a data source that does not need to be combined, it is still important to define a data mashup, since it can provide additional capabilities (e.g. projections, enforcement of combination rules, caching of structured data, etc.). This layer can use WADL descriptions to discover combination capabilities.

Like the Source Access layer, the Data Mashup layer also needs to provide a generic interface to the higher layer. Most systems expose the same interface, since it allows easily reusing a data mashup as source for other data mashups. We propose the same RESTful Web interface, since it offers the same advantages outlined above. HTTP methods on the interface of a data mashup will translate into HTTP methods on individual data sources according to specific combination logic.

2.3 Widget Layer

This layer mainly allows creating graphical components, referred to as ‘widgets’, which provide a graphical interface to data mashups. For the special case of Web clipping, this layer gets the

markup though the Source Access layer, extracts the fragments corresponding to particular blocks, and possibly adapts the markup (e.g. modification of CSS classes).

Existing tools providing this layer usually include basic widgets to format RSS/Atom feeds or to display results in a map. If the underlying data mashup or data source provides meta-information about its capabilities (e.g. a WADL description as proposed), it is possible to build advanced, generic widgets. For example, for data mashups or data sources providing querying capabilities, a generic widget could generate a form with as many fields as query parameters and format the result by using the schema of the result. A generic widget that displays a pie chart from a list of results in function of one the attributes (assuming its type is enumerated) can also be built.

Widgets must offer a common interface for their final assembly in the Widget Assembly layer. Section 5 discusses the design aspects of this interface. While most existing tools provide proprietary technologies to implement widgets, we propose to reuse portlet technology [1]. Portlets are interactive, Web mini-applications which can be aggregated into portal pages. There are many tools, known as 'portal servers', that allow the construction of portlet-based portals. Implementing widgets as portlets allows reusing the assembly portal capabilities.

2.4 Widget Assembly Layer

This layer allows assembling several widgets into the visual mashup. This assembly represents the final, unified graphical interface of the mashup. Assembly capability implies 'aggregation' of widgets, and possibly, 'coordination' between them. A Web page containing two widgets, one for finding stock symbols by company name (stock symbol finder) and another for displaying stock quotes by stock symbol (stock quote displayer), is an example of page where two widgets have been aggregated by the mashup creation tool. To improve integration of widgets, the Widget Assembly layer must provide a mechanism (e.g. publish/subscribe model) that allows coordination between widgets. For example, when the user searches a stock symbol in the stock symbol finder, the stock quote displayer could show the stock value of such a symbol. To do so, the Widget Assembly layer must pass the stock symbol returned by the stock symbol finder to the stock quote displayer.

As explained in section 5, we recommend to reuse portlet-based portal servers to implement the Widget layer, since the portal server already provide basic assembly capabilities.

2.5 Common Services

Our reference architecture distinguishes three common services, namely 'registry', 'collaboration', and 'security'. The services are potentially useful for several layers. Basic registry capabilities include registering and searching components (individual sources, data mashups, widgets, and visual mashups). Additional registry capabilities could include component lifecycle management (e.g. versioning) and community feedback (e.g. tagging, rating, etc.). Most existing mashup tools include basic registry capabilities and community feedback. To implement the registry capabilities we recommend reusing standard registries, such as UDDI.

Collaboration refers to the possibility of reusing knowledge automatically obtained by the system in function of the usage their users make. This knowledge can be used to automatically or semi-automatically compose data mashups and widgets. The authors of [16] present a proposal to automatically compose a data mashup in function of the user needs, expressed in terms of tags. In [5] when the user selects a given widget, the system automatically suggests her other widgets that have been combined with such a widget by other users in the past.

Finally, security services allow enforcing policy rules for access to components and composition of components. For the first aspect, enterprise tools include usual security mechanisms (e.g. role-based). With regard to the second aspect, the authors of [12] present a

pioneer work to protect interaction of widgets in the browser, forbidding scripting code of a given widget to modify or spy another widget.

3. Example

In this section we illustrate some of the main concepts of the architecture with a sample mashup illustrated in Figure 2. Martin works as Sales Manager at Acme Inc. Acme uses an on-demand CRM hosted at Salesforce (http://www.salesforce.com) storing data about its clients. Customer data includes a 'Satisfaction Level' field measuring the customer's level of satisfaction with Acme products. Martin has an idea to discover new potential customers: querying Salesforce for satisfied clients and using LinkedIn (http://www.linkedin.com) to find their business contacts. The idea is that Martin can call them and use the satisfied customer as reference.

In addition, Martin would want to enrich the new potential clients' data with additional information to prioritize and classify them. More precisely, he would like to access information from the new potential clients' companies, classify the potential clients by industry sector and geolocalize a given potential client in a map to plan a visit.

Since the number of clients and contacts is large and variable, Martin would like to automate the task using a mashup. The application should start by querying Salesforce for the most receptive contacts, and then search LinkedIn to obtain the first level 'connections' of each satisfied customer (name, industry, company, address, …). To add information about the new potential contacts' companies (revenue, recent news, etc), Martin would like to use an on-line service as Yahoo! Finance. Google Maps could be used to geolocalize a potential client given her/his address.

Now we explain how this mashup could be mapped to the reference architecture. First, we need to access the different sources, so the Source Access layer must provide an adaptor for each one:

- Salesforce offers a non-RESTful SOAP API, so a custom adaptor should be built to map the SOAP interface to the RESTful uniform interface. It is not necessary that this adaptor provides the whole functionality offered by the Salesforce API. For instance, query functionality over the basic data entities (e.g. customer data) will be enough for most cases.
- LinkedIn does not provide a Web service to access its data. Therefore, we need to use Web wrapper generation techniques to automate queries on the source: the wrapper will automatically login with Martin's credentials, fill in the search form with a customer name extracted from Salesforce, navigate to her/his first-level contacts' detail page and extract their data in structured form. This adaptor should also provide a RESTful interface.
- Yahoo! Finance offers a HTML Web interface (http://finance.yahoo.com/search) that we wish to see as a data source too. The process is similar to the one previously used for LinkedIn. In this case, the Web automation-based adaptor returns structured information about an enterprise given its company name. Alternatively, Yahoo! Finance could be directly accessed by the Widget layer by using 'Web clipping' techniques to obtain a markup fragment showing the desired information. The choice of one alternative or another will depend on the intended use of the data: if Martin needs to further process them (e.g. to combine them with data from other sources), he should use a wrapper able to provide structured data. Otherwise, clipping may be a simpler approach. In the remaining of this section we assume the clipping option.

Now, at the Data Mashup layer, we will combine the LinkedIn and Salesforce sources to create a new data mashup we will call PotentialNewLeads. This data mashup retrieves the satisfied customers' data from Salesforce, and for each one, queries LinkedIn to obtain her/his contacts' data in structured form. Conceptually, this can be seen as a join operation.

The next step is building the graphical components that will compose the mashup. The Widget layer is in charge of this task. The following widgets could be used in the example:

- QueryWidget: This widget inspects the WADL of a data mashup to introspect its supported query capabilities and the schema of the returned data. From this information, it automatically generates an HTML form allowing to query the data mashup, showing the obtained results in an

HTML table. In the example, from the PotentialNewLeads 'data mashup', it would generate a form with query parameters such as geographical area, name, industry and satisfaction level.

– PieChartWidget: This widget generates a pie chart from an input list of records using the values of one field to classify the records. In our example, we could use it to show the retrieved potential clients classified by the values of the 'industry' data field.
– MapWidget. Given an input address, geolocalizes it in a map.
– WebClippingWidget: This widget performs a clipping operation on a Web source. In our example, we can use it to clip the desired fragments from Yahoo! Finance.

Finally, the Widget Assembly layer shows the widgets and coordinates their interaction. In the example, when QueryWidget is used to execute a query, PieChartWidget will be invoked with the returned results to update the chart (ResultsListing event in Figure2). When the user clicks on a data record in the QueryWidget response (ItemSelected event in Figure2), MapWidget receives its address as input to show her/his position in the map and WebClippingWidget receives her/his company name to access the corresponding Yahoo! Finance page.

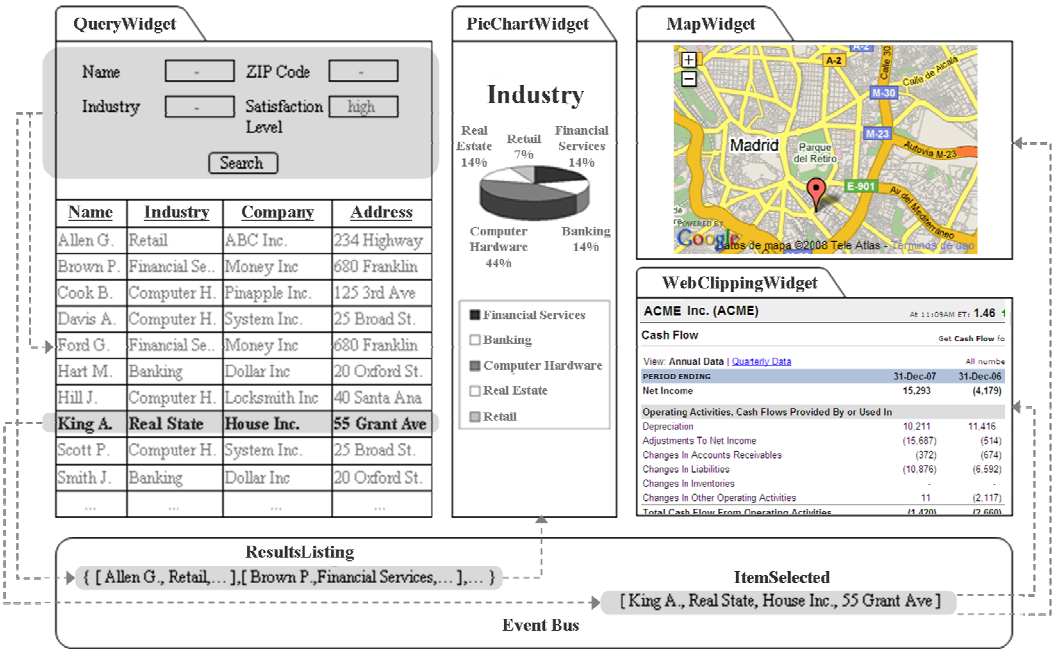


Fig. 2. Sample Mashup.

4. Data Mashup Layer

As it has already been mentioned, the Data Mashup layer combines input data from several sources to create a new component reusable at the data level. In a first analysis, we have identified the following main design aspects in this layer:

– Interface between the Source Access and the Data Mashup layers and interface provided by the Data Mashup layer to higher layers and/or external applications. Those issues were already discussed in section 2, where we advocated for the use of RESTful interfaces.
– Model used to express data combination logic.
– Method used to create the 'data mashup' definition. Most systems assume the user will directly use the model to 'manually' define the data mashup logic. Nevertheless, automatic composition approaches are also possible.

The following sub-sections describe the two latter aspects in more detail.

4.1 Model for Expressing Data Combination Logic

The model used in the majority of mashup platforms [10][13][20] to express data combination logic is based on the 'pipes and filters' pattern [8]. A 'pipe' is composed of a series of interconnected components. A link from the component 'A' to the component 'B' means that the output from 'A' is an input to 'B'. Components begin their execution when they have all their inputs available. This way, processing is propagated through the flow. Components implement operations such as accessing to sources, filtering input records and/or merging input feeds. The most mentioned advantage of this approach is simplicity. It is assumed that pipes can be created by 'power users' which do not necessarily have programming skills.

Another alternative to express data combination logic is the data federation/mediation pattern (see [7] for instance). Curiously, most tools ignore this approach although it has been widely used in the research literature to combine Web data. In this approach, the data combination logic is expressed as a 'view'. The view definition is usually written as a query over the data source schemas. To increase ease of use, a graphical interface can allow users to create the query by interconnecting components representing basic operations (selection, join, projection, …). This way, power users without programming skills can define views as easily as pipes.

An important advantage of data federation is that it is 'declarative'. The user writes a query and the system automatically computes all the possible execution plans to solve it. The user can either manually choose the plan or let the system optimizer make the choice (query optimization for data federation has been an active research field for years). In turn, the pipes and filters model uses a 'procedural' approach; a pipe can be seen as the explicit definition of a query execution plan.

Let us consider an example. Suppose we want to create a data mashup called PotentialNewLeads combining the data sources Salesforce and LinkedIn as in the example from section 3. In relational terms, this is a join operation between both sources by the 'customer name' attribute. Recall that the LinkedIn data source requires the customer name as mandatory search parameter.

If the user wants to retrieve the data for a given contact in both sources by specifying her/his name, the most efficient way is probably searching LinkedForce and Salesforce in parallel by contact name and joining the obtained records. In turn, if the user wants the data from several contacts (e.g. contacts from a certain city), then the only option is: 1) query Salesforce to obtain the contacts verifying the input conditions (e.g. city); 2) for each contact retrieved from Salesforce, search LinkedIn by contact name to obtain his/her data.

In the pipes and filter model, each of the two strategies outlined above needs to be implemented using a different pipe. In addition, if we wish to choose the most optimal strategy in function of the query parameters, we need additional logic to take the decision and invoke the desired pipe. In turn, in the data federation model, the possible execution plans for the PotentialNewLeads view will be automatically computed by the system (notice that the two strategies are simply two possible execution methods for computing a join). The query capabilities supported by each source will also be had into account. For instance, the system will be aware that the first strategy is only available when the query includes a 'customer name' parameter (see for instance the seminal work [7] on how to optimize plans with query restrictions). When the user issues a query, the system can choose the best execution plan in function of the input conditions transparently to the user.

In addition, the view model used by the data federation pattern fits perfectly with RESTful Web services. First, there is an obvious correspondence between the uniform interface provided by the HTTP methods GET, POST, PUT and DELETE and the QUERY, UPDATE, INSERT and DELETE operations used in views. Second, to assign global identifiers to each individual data resource (that is, to each tuple of a view), we can simply apply the standard notion of primary key and use it to derive unique global URIs. The concept of link can be also easily represented relaxing the foreign key notion by removing the reference integrity test (in the RESTful approach, it is not guaranteed links are not broken). As conclusion, we defend data federation as the most suitable approach for this design aspect.

4.2 Creating Data Mashups

As it was already mentioned, most systems assume the users will directly use the models discussed in the previous section to create 'data mashups'. Nevertheless, we can also consider using automatic composition techniques to create data mashups from a higher level description of the user's goals (in [16], a pioneer work is presented for combining RSS feeds). In general, automatic composition approaches implicitly assume that the ways in which people usually combines / transforms input data is a small subset of all the possible combinations/transformations. This is what makes the approach feasible. In the case of enterprise mashups, we think this assumption can prove wrong in a substantial number of cases because the nature of mashups is solving very specific needs that cannot profitably addressed by conventional methods. Nevertheless, in the long term, we think that both approaches can co-exist: automatic composition approaches are useful for non technical users trying to solve common needs; a manual approach will probably be needed, at least for a long time, to solve complex ad-hoc needs.

5. Widget Assembly Layer

As introduced in section 2, the Widget Assembly layer allows the assembly of several widgets into the visual mashup. Assembly of widgets involves two key design aspects: 'aggregation' of widgets and 'coordination' between them. The following sub-sections discuss these two aspects.

5.1 Widget aggregation

Widget aggregation refers to the capability of a tool to create a mashup by adding several widgets to a 'container' component. This container component is the mashup itself. For example, a Web page containing four 'isolated' widgets similar to those of the sample mashup (Figure 2), where all widgets force the user to introduce their input data by using a form (e.g. MapWidget would require to introduce manually the address), is an example of simple mashup. Widget aggregation requires all widgets to implement a common interface. Current mashup tools use proprietary interfaces, making it difficult to create a mashup with widgets from different tools.

In the Widget layer (sub-section 2.3) we proposed to reuse portlet technology, and in particular, to implement widgets as portlets. This allows benefiting from portal standards for the aggregation of heterogeneous portlets. These standards are the Java Portlet Specification [11] and WSRP [14]. The former standardizes the API the portal offers to a Java portlet. Portlets implemented according to this API can be deployed in any Java Portal Server supporting this standard. The second standardizes SOAP interfaces to let a producer (typically a portal) export its portlets to a remote consumer (typically another portal). This allows a consuming portal to aggregate remote portlets from a producer with no regard of the technology used by any of them. The Java Portlet Specification is in sync with WSRP, so that, a Java portlet can be exported automatically by WSRP to other consuming portals. Mashup tools implemented on top of a Java portal server can leverage these two standards to let aggregate local widgets (portlets) implemented with other Java mashup tools or remote widgets exported by other remote mashup tools.

5.2 Widget coordination

Widget coordination refers to the capability of using a mechanism to enable communication between widgets inside a given mashup. In the sample mashup, when the user executes a query with QueryWidget, PieChartWidget automatically receives the results and displays a chart. In the

same way, when the users selects a client in QueryWidget, MapWidget and WebClippingWidget automatically receive the client data, and locate the client on an map and display client's company information, respectively. To make this possible, the Widget Assembly layer must use a coordination mechanism enabling a widget to send data to other widgets.

The coordination mechanism should use a loosely coupled publish/subscribe model that allows widgets to communicate each other without creating explicit dependencies between them. For example, if another widget accepting a company name as input is added to the sample mashup (e.g. a widget displaying company information provided by Google Finance), it should also receive automatically the client data (which includes the company name) when the user selects a client in QueryWidget. Under a publish/subscribe model, when a widget wishes to send data to other widgets, it publishes an event. An event usually contains a logical name (e.g. ItemSelected) and the data (e.g. the client data). The name indicates the type of event. Each time a widget publishes an event, the mashup tool delivers it to all subscribing widgets. A widget subscribes to a type of event by specifying its name. In the sample mashup, MapWidget and WebClippingWidget are subscribed to the ItemSelected event, which is produced by QueryWidget.

Current mashup tools use proprietary publish/subscribe mechanisms [9]. Using a standard mechanism let widgets implemented with different tools to communicate each other. Implementing widgets as portlets also lets reuse the standard publish/subscribe mechanism provided by portal standards. The Java Portlet Specification enables communication between local Java portlets, and WSRP lets send and receive events to and from remote portlets.

Finally, most tools assume that the user will manually select the widgets that will make up the mashup. To achieve an integrated experience, the user must choose widgets that use the same type of events and make explicit event conversions when needed (e.g. event names, fields in the event data, etc.). An advanced tool could automatically suggest the user an initial set of widgets from a higher level description (e.g. keywords) of the mashup he/she wishes to build and apply appropriate conversions to events if needed. Furthermore, when the user adds a given widget to the mashup, the tool could automatically suggest other widgets that have been combined with such a widget by other users in the past (see [5] for a possible approach).

6. Related Work

To the best of our knowledge, the only existing reference architecture for mashups was presented by the technology advisory company Gartner Group [2]. Their architecture adopts a business point of view which limits its usefulness from the technical side. For instance, the main layers of their architecture are too coarse, making them of little help to detect overlaps or complementariness between different technical offerings. In addition, they do not discuss the different technical approaches that can be used in each layer and do not make any specific recommendations. We also think some technical key concepts are not captured by their architecture:

- Gartner Architecture emphasizes the idea of building mashups from 'reusable' components, but they make no clear distinction between components reusable at the data level ('data mashups' in our terminology) and at the presentation level ('widgets').
- Related with the previous issue, the central layer in the Gartner architecture is the 'Assembly' layer, where basic components are assembled to build the mashup. Gartner makes no distinction between the problem of assembling data components and the problem of assembling visual components. From the technical point of view, it is obvious that they are different problems and that a tool can excel at one task while providing little support for the other. There exist commonalities but they can be included as common services.
- Gartner architecture does not include a layer for dealing with the issues addressed by the Collaboration service of our architecture. Their 'Community' layer is only concerned with component searching/tagging. We believe that providing intelligent ways of reusing the work invested by other users will be a key ingredient of successful mashup offerings.

Other group of related work is formed by the different mashup creation techniques that have been recently presented in both the research [5][12][16][18] and industrial arenas [4][9][10][13][19][20]. A detailed discussion of how each of such works fits in the architecture is out of the scope of this paper and will be addressed in future works.

7. Conclusions and Future Work

In this paper, we have overviewed a new reference architecture for mashup applications. We have also discussed in more detail two crucial layers of the architecture and identified their main design aspects. We have also made some specific proposals to address some key issues. Our current and future work is focused on: (1) identifying the main design aspects involved in the rest of layers and studying the different approaches to solve them, (2) surveying real mashup applications (especially in enterprise environments) to learn about crucial difficulties and evaluate how each approach can solve them, and (3) developing a working prototype of the full architecture to further validate our proposal.

References

1. Bellas, F.: Standards for Second-Generation Portals. IEEE Internet Computing, 8 (2), pp. 54–60 (2004)
2. Bradley, A. Gartner Group Reference Architecture for Enterprise Mashups. http://www.gartner.com/DisplayDocument?id=519808
3. Chia-Hui Chang, Kayed M., Girgis, M.R., Shaalan, K.F. A Survey of Web Information Extraction Systems. IEEE Transactions on Knowledge and Data Engineering, 18 (10), pp. 1411-1428 (2006)
4. Denodo Technologies. http://www.denodo.com
5. Ennals, R.J., Brewer, E.A., Garofalakis, M.N., Shadle, M., Gandhi, P.: Intel Mash Maker: Join the Web. SIGMOD Record, 36 (4), pp. 27–33. ACM (2007)
6. Fielding, R.T., Taylor, R.N.: Principled Design of the Modern Web Architecture. ACM Transactions on Internet Technology, 2 (2), pp. 115–150 (2002)
7. Florescu, D., Halevy, A., Manolescu, I., Suciu, D.: Query Optimization in the Presence of Limited Access Patterns. Proceedings of ACM SIGMOD Conference, pp. 311-322 (1999)
8. Hohpe, G., Woolf, B. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley. ISBN: 032120068 (2003)
9. IBM Mashup Starter Kit, http://www.alphaworks.ibm.com/tech/ibmmsk
10. JackBe, http://www.jackbe.com
11. Java Community Process: Java Portlet Specification - Version 2.0, http://jcp.org/en/jsr/detail?id=286
12. Keukelaere, F.D., Bhola, S., Steiner, M., Chari, S., Yoshihama, S.: SMash: Secure Component Model for Cross-Domain Mashups on Unmodified Browsers. In: 17th International Conference on World Wide Web, pp. 535–544. ACM, Beijing, China (2008)
13. Microsoft Popfly, http://www.popfly.com
14. OASIS: Web Services for Remote Portlets Specification – Version 2.0, http://docs.oasis-open.org/wsrp/v2/wsrp-2.0-spec-os-01.html
15. Pan, A., Raposo, J., Álvarez, M., Hidalgo, J., and Viña, A. Semi Automatic Wrapper-Generation for Commercial Web Sources. Proceedings of IFIP WG8.1 Conference on Enterprise Information Systems in the Internet Context (2002)
16. Riabov, A.V., Bouillet, E., Feblowitz, M., Liu, Z., Ranganathan, A.: Wishful Search: Interactive Composition of Data Mashups. In: 17th International Conference on World Wide Web, pp. 775–784. ACM, Beijing, China (2008)
17. Web Application Description Language, https://wadl.dev.java.net
18. Wong, J., Hong, J.I.: SMash: Making Mashups with Marmite: Towards End-User Programming for the Web. In: SIGCHI Conference on Human Factors in Computing Systems, pp. 1435–1444. ACM, San Jose, California, USA (2007)
19. WSO2 Mashup Server, http://wso2.org/projects/mashup
20. Yahoo! Pipes, http://pipes.yahoo.com/pipes

Developing a Service Oriented Alternative for Distributed Multi-Agent Systems

Dante I. Tapia, Juan F. de Paz, Sara Rodríguez, Javier Bajo and Juan M. Corchado

Departamento Informática y Automática
Universidad de Salamanca
Plaza de la Merced s/n, 37008, Salamanca, Spain
{dantetapia; fcofds; srg; jbajope; corchado}@usal.es

Abstract. This paper presents a service oriented architecture approach that has enhanced the performance of a multi-agent system aimed at enhancing the assistance and health care for Alzheimer patients living in geriatric residences. The proposed architecture allows a more efficient distribution of the daily activities in the multi-agent system. The results obtained after testing the architecture in a real health care scenario demonstrate that the ALZ-MAS 2.0 based on the service oriented approach is far more robust and has better performance than the previous version of this system.

Keywords: Multi-agent Systems, Services Oriented Architectures, Case-Based Reasoning, Case-Based Planning, Health Care.

1 Introduction

The continuous growth of the Internet requires frameworks for web application integration [12]. Web applications are executed in distributed environments, and each part that composes the program can be located in a different machine. The absence of a strategy for integrating applications generates multiple points of failure that can affect the systems' performance. Some of the technologies that have acquired a relevant paper in the web during the last years are the multi-agent systems and the SOA architectures. This work describes a novel architecture for developing multi-agent systems and explains how it has been designed and applied to a real scenario. The architecture presents important improvements in the vision of the integration of web applications. One of the most important characteristics is the use of intelligent agents as the main components in employing a service oriented approach, focusing on distributing the majority of the systems' functionalities into remote and local services and applications. The architecture proposes a new and easier method of building distributed multi-agent systems, where the functionalities of the systems are not integrated into the structure of the agents, rather they are modelled as distributed services which are invoked by the agents acting as controllers and coordinators.

Agents have a set of characteristics, such as autonomy, reasoning, reactivity, social abilities, pro-activity, mobility, organization, etc. which allow them to cover several needs for highly dynamic environments. Agent and multi-agent systems have been successfully applied to several scenarios, such as education, culture, entertainment, medicine, robotics, etc. [7]. The characteristics of the agents make them appropriate for developing dynamic and distributed systems, as they possess the capability of adapting themselves to the users and environmental characteristics [9]. Most of the agents are based on the deliberative Belief, Desire, Intention (BDI) model [15], where the agents' internal structure and capabilities are based on mental aptitudes, using beliefs, desires and intentions [3]. Nevertheless, complex systems need higher adaptation, learning and autonomy levels than pure BDI model [3]. This is achieved by modelling the agents' characteristics [15] to provide them with mechanisms that allow solving complex problems and autonomous learning. Some of these mechanisms are Case-Based Reasoning (CBR) [1] and Case-

Based Planning (CBP), where problems are solved by using solutions to similar past problems [6] [7]. Solutions are stored into a case memory, which the mechanisms can consult in order to find better solutions for new problems. CBR and CBP mechanisms have been modelled as external services. Deliberative agents use these services to learn from past experiences and to adapt their behaviour according the context.

This paper briefly describes the FUSION@ architecture, a service oriented alternative for distributed multi-agent systems and ALZ-MAS 2.0, a multi-agent system aimed at enhancing the assistance and health care for Alzheimer patients living in geriatric residences. ALZ-MAS 2.0 is based on FUSION@ and implements a services oriented approach, where functionalities, including CBR and CBP mechanisms, are not integrated into the structure of the agents, rather they are modelled as distributed services and applications which are invoked by the agents.

In the next section, the problem description that motivated this work is presented. Section 3 briefly presents the FUSION@ architecture. Section 4 describes the basic components of ALZ-MAS 2.0 and shows how a CBP mechanism has been modelled for distributing resources. Finally Section 5 presents the results and conclusions obtained in this work.

2 Problem Description

Excessive centralization of services negatively affects the systems' functionalities, overcharging or limiting their capabilities. Classical functional architectures are characterized by trying to find modularity and a structure oriented to the system itself. Modern functional architectures like Service-Oriented Architecture (SOA) consider integration and performance aspects that must be taken into account when functionalities are created outside the system. These architectures are aimed at the interoperability between different systems, distribution of resources, and the lack of dependency of programming languages [5]. Services are linked by means of standard communication protocols that must be used by applications in order to share resources in the services network [2]. The compatibility and management of messages that the services generate to provide their functionalities is an important and complex element in any of these approaches.

One of the most prevalent alternatives to these architectures is the multi-agent systems technology which can help to distribute resources and reduce the central unit tasks [2]. A distributed agents-based architecture provides more flexible ways to move functions to where actions are needed, thus obtaining better responses at execution time, autonomy, services continuity, and superior levels of flexibility and scalability than centralized architectures [4]. Additionally, the programming effort is reduced because the agents cooperate in solving problems and reaching specific goals, thus giving the systems the ability to generate knowledge and experience.

Agent and multi-agent systems combine classical and modern functional architecture aspects. Multi-agent systems are structured by taking into account the modularity in the system, and by reuse, integration and performance. Nevertheless, integration is not always achieved because of the incompatibility among the agents' platforms. The integration and interoperability of agents and multi-agent systems with SOA and Web Services approaches has been recently explored [2]. Some developments are centred on communication between these models, while others are centred on the integration of distributed services, especially Web Services, into the structure of the agents [13] [14] [10] [11]. Although these developments provide an adequate background for developing distributed multi-agent systems integrating a service oriented approach, most of them are in early stages of development, so it is not possible to actually know their potential in real scenarios.

3 The FUSION@ Architecture

The development of AmI-based software requires creating increasingly complex and flexible applications, so there is a trend toward reusing resources and share compatible platforms or architectures. In some cases, applications require similar functionalities already implemented into other systems which are not always compatible. At this point, developers can face this problem through two options: reuse functionalities already implemented into other systems; or re-deploy the capabilities required, which means more time for development, although this is the easiest and safest option in most cases. While the first option is more adequate in the long run, the second one is most chosen by developers, which leads to have replicated functionalities as well as greater difficulty in migrating systems and applications. This is a poorly scalable and flexible model with reduced response to change, in which applications are designed from the outset as independent software islands.

FUSION@ has been designed to facilitate the development of distributed multi-agent systems with high levels of human-system-environment interaction, since agents have the ability to dynamically adapt their behaviour at execution time. It also provides an advanced flexibility and customization to easily add, modify or remove applications or services on demand, independently of the programming language. FUSION@ formalizes four basic blocks: Applications, which represent all the programs that can be used to exploit the system functionalities. They can be executed locally or remotely, even on mobile devices with limited processing capabilities, because computing tasks are largely delegated to the agents and services; An Agents Platform as the core of FUSION@, integrating a set of agents, each one with special characteristics and behaviour. These agents act as controllers and administrators for all applications and services, managing the adequate functioning of the system, from services, applications, communication and performance to reasoning and decision-making; Services, which are the bulk of the functionalities of the system at the processing, delivery and information acquisition levels. Services are designed to be invoked locally or remotely; and finally a Communication Protocol which allows applications and services to communicate directly with the Agents Platform. The protocol is based on SOAP specification and it is completely open and independent of any programming language [5].

These blocks are managed by means of pre-defined agents which provide the basic functionalities of FUSION@: CommApp Agent is responsible for all communications between applications and the platform; CommServ Agent is responsible for all communications between services and the platform; Directory Agent manages the list of services that can be used by the system; Supervisor Agent supervises the correct functioning of the other agents in the system; Security Agent analyzes the structure and syntax of all incoming and outgoing messages; Manager Agent decides which agent must be called by taking into account the services performance and users preferences; Interface Agents are designed to be embedded in users' applications. Interface agents communicate directly with the agents in FUSION@ so there is no need to employ the communication protocol, rather the FIPA ACL specification.

FUSION@ also facilitates the inclusion of context-aware technologies that allow systems to automatically obtain information from users and the environment in an evenly distributed way, focusing on the characteristics of ubiquity, awareness, intelligence, mobility, etc. The goal in FUSION@ is not only to distribute services and applications, but to also promote a new way of developing systems focusing on ubiquity and simplicity. An example of a service in FUSION@ can be observed in Figure 1.

SERVICE		DESCRIPTION	
readCHIP		This service identificates a chip once it has been detected for a RFID reader.	
PROFILE	**ClientRole** User Agent	**ProviderRole** Devices Agent	
	Inputs	**Outputs**	
	idDevice: string typeDevice: string deviceLocation: location	[typeCHIP OK] idCHIP: string idUser: string chipLocation: location	[typeCHIP NOT OK]

Fig. 1. ReadCHIP: An example of service in FUSION@.

Figure 1 shows the readCHIP service. This service has been implemented to facilitate indoor location based on RFID technology. When a RFID reader detects the presence of a chip, the readCHIP service is automatically invoked. The inputs considered for this service consists of the device identification, the type of device, and the location of the device. At this moment the service checks the type of CHIP and calculates the location information, that is, the identification for the chip, the user identification and the coordinates which determine the physical position. This information is then sent to the Devices Agent in order to be automatically processed.

In the next section, ALZ-MAS 2.0 is presented, where FUSION@ has helped to distribute most of its functionalities and re-design a completely functional multi-agent system aimed at improving several aspects of dependent people.

4 ALZ-MAS 2.0

ALZ-MAS 2.0 is an improved version of ALZ-MAS (ALZheimer Multi-Agent System) [6] [7], a multi-agent system aimed at enhancing the assistance and health care for Alzheimer patients living in geriatric residences. The main functionalities in the system are managed by deliberative BDI agents, including Case-Based Reasoning (CBR) and Case-Based Planning (CBP) mechanisms.

ALZ-MAS structure has five different deliberative agents based on the BDI model (BDI Agents), each one with specific roles and capabilities:

- *User Agent*. This agent manages the users' personal data and behaviour (monitoring, location, daily tasks, and anomalies). The *User Agent* beliefs and goals applied to every user depend on the plan or plans defined by the super-users.
- *SuperUser Agent*. This agent inserts new tasks into the Manager Agent to be processed by a CBR and CBP mechanisms.
- *ScheduleUser Agent*. It is a BDI agent with a CBP mechanism embedded in its structure. It schedules the users' daily activities and obtains dynamic plans depending on the tasks needed for each user. There is one *ScheduleUser Agent*s for each nurse connected to the system.
- *Admin Agent*. It runs on a Workstation and plays two roles: the security role that monitors the users' location and physical building status (temperature, lights, alarms, etc.) through continuous communication with the *Devices Agent*; and the manager role that handles the databases and the task assignment.
- *Devices Agent*. This agent controls all the hardware devices. It monitors the users' location (continuously obtaining/updating data from sensors), interacts with sensors and actuators to receive information and control physical services (wireless devices status, communication, temperature, lights, door locks, alarms, etc.).

In the initial version of ALZ-MAS, each agent integrated its own functionalities into their structure. If an agent needs to perform a task which involves another agent, it must communicate

with that agent to request it. So, if the agent is disengaged, all its functionalities will be unavailable to the rest of agents. This has been an important issue in ALZ-MAS, since agents running on PDAs are constantly disconnecting from the platform and consequently crashing, making it necessary to restart (killing and launching new instances) those agents. Another important issue is that the CBR and CBP mechanisms are integrated into the agents. These mechanisms are busy almost all the time, overloading the respective agents. Because CBR and CBP mechanisms are the core of the system, they must be available at all times. The system depends on these mechanisms to generate all decisions, so it is essential that they have all processing power available in order to increase overall performance. In addition, the use of CBR and CBP mechanisms into deliberative BDI agents makes these agents complex and unable to be executed on mobile devices. In ALZ-MAS 2.0, these mechanisms have been modelled as services to distribute resources.

The entire ALZ-MAS structure has been modified, separating most of the agents' functionalities from those to be modelled as services. However, all functionalities are the same in both approaches, since we have considered it appropriated to compare the performance of both systems in identical conditions. As an example showing the differences between both approaches, the next sub-section describes the CBP mechanism that has been extracted from the *ScheduleUser Agent* structure and modelled as a service.

As seen on Figure 2, the entire ALZ-MAS structure has been modified according to FUSION@ model, separating most of the agents' functionalities from those to be modelled as services. However, all functionalities are the same in both approaches, since we have considered it appropriated to compare the performance of both systems to prove the efficiency of FUSION@ model.

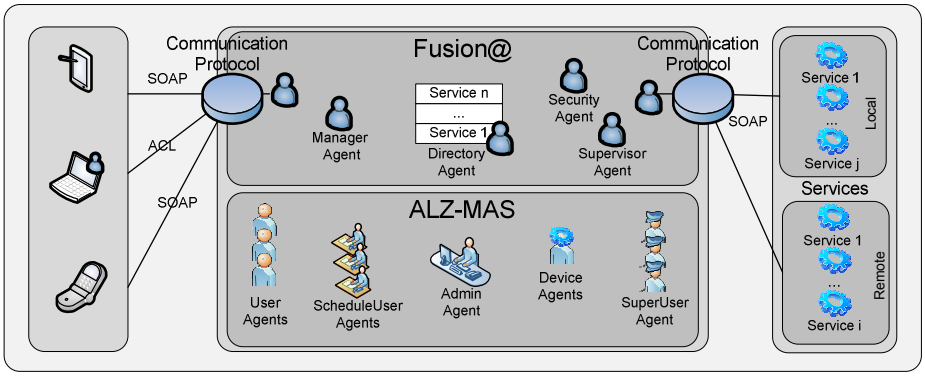


Fig. 2. ALZ-MAS 2.0 basic structure

4.1 A Case-Based Planning Mechanism for Scheduling Daily Activities

As previously mentioned, some agents in ALZ-MAS integrate CBR and CBP mechanisms (then modelled as services in ALZ-MAS 2.0), which allow them to make use of past experiences to create better plans and achieve their goals. These mechanisms provide the agents greater learning and adaptation capabilities. The main characteristics of the CBP mechanism are described in the remainder of this section.

Case-Based Reasoning (CBR) is a type of reasoning based on past experiences [1]. CBR solve new problems by adapting solutions that have been used to solve similar problems in the past, and learn from each new experience. The primary concept when working with CBR is the concept of case, which is described as a past experience composed of three elements: an initial state or problem description that is represented as a belief; a solution, which provides the sequence of actions carried out in order to solve the problem; and a final state, which is represented as a set of

goals. CBR manages cases (past experiences) to solve new problems. The way cases are managed is known as the CBR cycle, and consists of four sequential phases: retrieve, reuse, revise and retain. The retrieve phase starts when a new problem description is received. Similarity algorithms are applied so that the cases with the problem description most similar to the current one can be retrieved from the cases memory. Once the most similar cases have been retrieved, the reuse phase begins by adapting the solutions for the retrieved cases in order to obtain the best solution for the current case. The revise phase consists of an expert revision of the proposed solution. Finally, the retain phase allows the system to learn from the experiences obtained in the three previous phases, and consequently updates the cases memory.

CBP comes from CBR, but is specially designed to generate plans (sequence of actions) [6] [7]. In CBP, the proposed solution for solving a given problem is a plan. This solution is generated by taking into account the plans applied for solving similar problems in the past. The problems and their corresponding plans are stored in a plans memory. The reasoning mechanism generates plans using past experiences and planning strategies, which is how the concept of Case-Based Planning is obtained [7]. CBP consists of four sequential stages: the retrieve stage, which recovers the past experiences most similar to the current one; the reuse stage, which combines the retrieved solutions in order to obtain a new optimal solution; the revise stage, which evaluates the obtained solution; and retain stage, which learns from the new experience. Problem description (initial state) and solution (situation when final state is achieved) are represented as beliefs, the final state as a goal (or set of goals), and the sequences of actions as plans. The CBP cycle is implemented through goals and plans. When the goal corresponding to one of the stages is triggered, different plans (algorithms) can be executed concurrently to achieve the goal or objective. Each plan can trigger new sub-goals and, consequently, cause the execution of new plans. In practice, what is stored is not only a specific problem with a specific solution, but also additional information about how the plans have been derived. As with CBR, the case representation, the plans memory organization, and the algorithms used in every stage of the CBP cycle are essential in defining an efficient planner.

In the initial version of ALZ-MAS, the CBR and CBP mechanisms are deeply integrated into the agents' structure. In ALZ-MAS 2.0, these mechanisms have been modelled as services linked to agents, thus increasing the system's overall performance. To generate a new plan, a *ScheduleUser Agent* (running on a PDA) sends a request to the platform. The message is processed and the platform invokes the mechanism (or service). The mechanism receives the message and starts to generate a new plan. Then, the solution is sent to the platform which delivers the new plan to all *ScheduleUser Agents* running. The CBP service creates optimal paths and scheduling in order to facilitate the completion of all tasks defined for the nurses connected to the system [7].

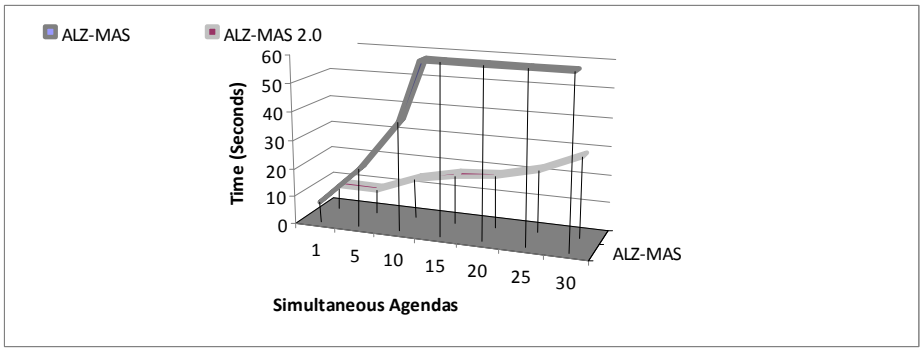
5 Results and Conclusions

The integration of web applications plays an important role in the advance of the internet. This paper has presented the FUSION@ architecture, which proposes a novel approach for integrating: applications, agents and services. FUSION@ facilitates the inclusion of context-aware technologies that allow systems to automatically obtain information from users and the environment in an evenly distributed way. The proposed architecture has been used to develop the ALZ-MAS 2.0 system, a variation of the previous ALZ-MAS system, which models the CBP-BDI and CBR-BDI mechanisms as services. The performance of ALZ-MAS 2.0 has been highly improved.

Several tests have been done to demonstrate if a SOA approach is appropriate to distribute resources and optimize the performance of multi-agent systems, in this case ALZ-MAS 2.0. The tests consisted of a set of requests delivered to the CBP mechanism which in turn had to generate paths for each set of tasks (i.e. scheduling). For every new test, the cases memory of the CBP mechanism was deleted in order to avoid a learning capability, thus requiring the mechanism to

accomplish the entire planning process. A task is a java object that contains a set of parameters (TaskId, MinTime, MaxTime, ScheduleTime, UserId, location, etc.). ScheduleTime is the time in which a specific task must be accomplished, although the priority level of other tasks needing to be accomplished at the same time is factored in. The CBP mechanism increases or decreases ScheduleTime and MaxTime according to the priority of the task: *ScheduleTime = ScheduleTime-5min*TaskPriority and MaxTime = MaxTime+5min*TaskPriority*. Once these times have been calculated, the path is generated taking the RoomCoordinates into account. There were 30 defined agendas each with 50 tasks. Tasks had different priorities and orders on each agenda. Tests were carried out on 7 different test groups, with 1, 5, 10, 15, 20, 25 and 30 simultaneous agendas to be processed by the CBP mechanism. 50 runs for each test group were performed, all of them on machines with equal characteristics. Several data have been obtained from these tests, notably the average time to accomplish the plans, the number of crashed agents, and the number of crashed services. For ALZ-MAS 2.0 five CBP services with exactly the same characteristics were replicated.

Fig. 3(Top) shows the average time needed by both systems to generate the paths for a fixed number of simultaneous agendas. The previous version of ALZ-MAS was unable to handle 15 simultaneous agendas and time increases to infinite because it was impossible to perform those requests. However, ALZ-MAS 2.0 had 5 replicated services available, so the workflow was distributed and allowed the system to complete the plans for 30 simultaneous agendas. Another important data is that although the previous version of ALZ-MAS performed slightly faster when processing a single agenda, performance was constantly reduced when new simultaneous agendas were added. This fact demonstrates that the overall performance of ALZ-MAS 2.0 is better when handling distributed and simultaneous tasks (e.g. agendas), instead of single tasks. Fig. 3(Down) shows the number of crashed agents for both versions of ALZ-MAS during tests. None of the tests where agents or services crashed were taken into account to calculate the data presented in Fig. 3, so these tests were repeated. As can be seen, the previous version of ALZ-MAS is far more unstable than ALZ-MAS 2.0. These data demonstrate that this approach provides a higher ability to recover from errors.



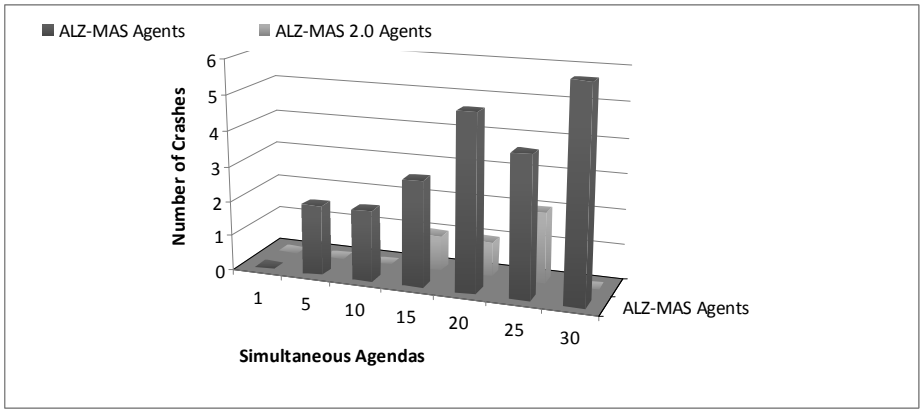


Fig. 3. Top: Time needed for both systems to generate paths for a group of simultaneous agendas; Down: Number of agents crashed at both systems

Although these tests have provided us with very useful data, it is necessary to continue experimenting with FUSION@. A SOA approach is an efficient way to distribute resources and develop more robust multi-agent systems, especially when handling complex mechanisms as the CBP presented.

Acknowledgments. This work has been supported by the IMSERSO 137/2007, the UPSA U05E1A-07L01 and the MCYT TIN2006-14630-C03-03 projects.

References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: foundational Issues, Methodological Variations, and System Approaches. AI Communications, 7, 39-59, IOS Press (1994)
2. Ardissono, L., Petrone, G., Segnan, M.: A conversational approach to the interaction with Web Services. Computational Intelligence, 20, 693-709, Blackwell Publishing (2004)
3. Bratman, M.E.: Intentions, plans and practical reason. Harvard University Press, Cambridge, MA (1987)
4. Camarinha-Matos, L.M., Afsarmanesh, H.: A Comprehensive Modeling Framework for Collaborative Networked Organizations. Journal of Intelligent Manufacturing, 18(5), 529-542, Springer Netherlands (2007)
5. Cerami, E.: Web Services Essentials Distributed Applications with XML-RPC, SOAP, UDDI & WSDL. O'Reilly & Associates, Inc. 1st Edition (2002)
6. Corchado, J.M., Bajo, J., Abraham, A.: GERAmI: Improving the delivery of health care. IEEE Intelligent Systems, Special Issue on Ambient Intelligence, 23(2), 19-25 (2008)
7. Corchado, J.M., Bajo, J., De Paz, Y., Tapia, D.I.: Intelligent Environment for Monitoring Alzheimer Patients, Agent Technology for Health Care. Decision Support Systems, 44(2), 382-396, Eslevier, Netherlands (2008)
8. de Paz, J.F., Rodríguez, S., Bajo, J., Corchado, J.M.: Dynamic Case Based Planning. In: 8th Int. Conference on Computational and Mathematical Methods in Science and Engineering, vol. 1, pp. 213-224, La Manga del Mar Menor, Spain (2008)
9. Jayaputera, G.T., Zaslavsky, A.B., Loke, S.W.: Enabling run-time composition and support for heterogeneous pervasive multi-agent systems. Journal of Systems and Software, 80(12), 2039-2062 (2007)
10. Li, Y., Shen, W., Ghenniwa, H.: Agent-Based Web Services Framework and Development Environment. Computational Intelligence, 20(4), 678-692, Blackwell Publishing (2004)
11. Liu, X.: A Multi-Agent-Based Service-Oriented Architecture for Inter-Enterprise Cooperation System. In: 2nd International Conference on Digital Telecommunications. IEEE Computer Society, Washington, DC (2007)

12. Oren, E., Haller, A., Mesnage, C., Hauswirth, M., Heitmann, B., Decker, S.: A Flexible Integration Framework for Semantic Web 2.0 Applications. IEEE Software, vol. 24, no. 5, pp. 64-71, (2007)
13. Ricci, A., Buda, C., Zaghini, N.: An agent-oriented programming model for SOA & web services. In: 5th IEEE International Conference on Industrial Informatics, pp. 1059-1064, Vienna, Austria (2007)
14. Shafiq, M.O., Ding, Y., Fensel, D.: Bridging Multi-Agent Systems and Web Services: towards interoperability between Software Agents and Semantic Web Services. In: 10th IEEE International Enterprise Distributed Object Computing Conference, pp. 85-96, IEEE Computer Society, Washington, DC (2006)
15. Wooldridge, M., Jennings, N.R.: Intelligent Agents: Theory and Practice. The Knowledge Engineering Review, 10(2), 115-152, Cambridge University Press (1995)

Using Weaving Models to automate Model-Driven Web Engineering proposals

Juan M. Vara[1], Mª Valeria De Castro[1], Marcos Didonet Del Fabro[2], Esperanza Marcos[1]

1: Kybele Research Group, Universidad Rey Juan Carlos (Madrid - Spain)
2: ATLAS Group (INRIA & LINA), University of Nantes (France)
{juanmanuel.vara, valeria.decastro, esperanza.marcos}@urjc.es,
marcos.didonet@univ-nantes.fr

Abstract. The impact of Model-Driven Software Development in Web Engineering has given raise to the advent of Model-Driven Web Engineering, a new approach for Web Information Systems development. Its basic assumption is the consideration of models as first class entities that drive the development process from analysis to final deployment. Basically, each step of the process consists of the generation of one or more output models from one or more input models. Thus, model transformations are the key to complete each step of the process. However, the special nature of the behavioral models implied at the early stages of a Model-Driven Web Engineering process complicates the specification of a model transformation that works for any input model. In such situations, it is not feasible to automate the whole development process (one of the premises of Model-Driven Development). Some design decisions has to be considered before executing each model transformation. This work shows how we solve this problem in SOD-M, a model-driven approach for the development of Service-Oriented Web applications. The technique proposed is based on the use of weaving models as annotation models and it can be easily generalized to other domains and contexts.

1 Introduction

According to the roadmap for research in Service-Oriented computing (SOC) outlined by Papazoglou et al. [18], one of the main challenges that SOC has to face is the provision of methodologies supporting the specification and design of service composition. They should provide software engineers with the tools to move from the earlier stages of business analysis to the final step of implementation. While the design and development of simpler services is a relatively simple task, the development of business process comprising several independent services is not so simple. The transformation from high-level business models, generally defined by business analysts, to an executable business process language, such as BPEL4WS, is far away from being a trivial issue [24]. MDA (Model Driven Architecture) [16] is an important tool for the alignment of high-level business processes and information technologies [11]. It provides with a conceptual structure to combine the models built by business managers and analysts with those developed by software developers.

Attending to these facts, in previous works we have defined a model-driven approach for the development of service-oriented web applications [5]. The Service-Oriented Development Method (SOD-M) is part of MIDAS, a Model Driven Architecture framework for development of Web Information Systems (WIS). Therefore, SOD-M provides with all the advantages derived from the Model Driven Engineering (MDE) approach [2, 20]. The method defines a model-driven process and proposes a set of models at the different MDA abstraction levels. It starts from a high level business model. After several model transformations a service composition model is obtained. The later simplifies the mapping to a specific Web service technology. This work focuses on some of those models. The ones defined for behavioral modeling of service-oriented applications; and the

set of mappings rules between them. Without automating the mappings between models, the effort needed to manually transform the models become prohibitive and organizations using MDA will not get a full return on MDA's promise of faster, less costly software development [9].

All the mappings between the models comprised in MIDAS framework follow a common approach [4, 21]: firstly, we specify the transformation rules with natural language to later formalize them using graph transformation rules [19]. Once formalized, those rules are implemented using the ATLAS Transformation Language (ATL) [12]. Thus, after having formalized the mappings between the PIM models of SOD-M in [5], we face the implementation of those rules. This task raises some problems due to nature of the models considered in this work.

On the first hand, PIM to PSM transformations are more prone to be automated than PIM to PIM transformations. While the former implies decreasing the abstraction level and consequently handling more specific artifacts easier to be modeled; the later requires decisions from the designer due to the higher abstraction level of the implied elements. As a matter of fact, the advent of generic model transformation approaches was mainly related with addressing this problem [23].

On the other hand, business process models, like the ones we handle here, present considerable differences compared to structural models that raises a number of issues concerning model transformation [15, 17]. One has to be familiar with the hidden concepts in the metamodels. Resulting ambiguities on the metamodel layer have to be solved either by reasoning algorithms or user input.

Consequently, defining a one-size-fits-all model transformation in such contexts is too ambitious. There is a need to define non-uniform mappings [10]. More information is needed in order to execute the model transformation. It is our belief that in a real MDE context, this additional input should take the shape of a model.

By means of a case study, this work shows how, a weaving model, can be used as a container for the extra data. We are able to parameterize a model transformation by defining a weaving model that serves to annotate the source model. Then, both the source and the weaving model are taken as input to generate the target model. The results lend strong support to the idea that current model-driven engineering tools, like model transformations and weaving models are powerful enough to fulfill the requirements of Web Engineering. Moreover, the approach presented here can be easily generalized to other domains and contexts, where simple model transformations are not enough and there is a need for some kind of design decisions making any time a transformation is executed.

This paper is structured as follows: both the proposal and SOD-M, the framework in which it is formulated, are introduced in Section 2. A case study showing how to apply the idea is explained in Section 3. Finally, Section 4 outlines the main findings of the paper and raises a number of questions for future research.

2 Using model annotations in SOD-M

SOD-M, the Service-Oriented Development Method, is a service-oriented approach for Web Information Systems (WIS) development. It defines a model-driven process to obtain a service composition design from high-level business models.

In this work we focus on the models proposed by SOD-M for the behaviour modelling of the system at a high abstraction level (Analysis models in traditional Software Engineering, Platform Independent Models in MDE jargon). Since it is a service-oriented approach, such models serve to identify the *business services* offered by the system as well as the functionalities and processes needed to carry them out. The modelling process of SOD-M at PIM level is summarized in Fig. 1.

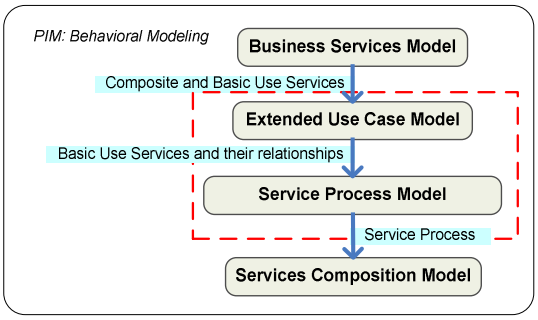


Fig. 1. Behavioural Modelling process in SOD-M

Here we address the mapping between two of these models (see [5] for more information on the others):

- The *Extended Use Case* model is represented with a Use Case model at a lower granularity than the previous one. Its main objective is to model the functionalities (services) required by the system to carry out each one of the business services identified in the Business Services model.
- The *Service Process* model is a kind of Activity Diagram used to represent the *service processes*. Thus, it shows the set of logically related activities that need to be performed to carry out a business service.

This work is based on an earlier study about the mappings between SOD-M PIM models. The aim of this work was initially to implement a set of mappings previously sketched [5] using the ATL language. However, once we started to code the ATL program, we realized that some information needed to generate the target model was not included in the source model. For each execution of the transformation some *extra* data was needed. In some sense, these extra data can be shown as a way of parameterize the transformation. In this context, the first option was to extend the source metamodel to support the modeling of these extra data. However this meant polluting the metamodel with concepts not relevant for the domain that it represents. We needed a different way to collect these extra data that was related with the source model but not included in it. That is, we just needed a way to annotate the input model [8]. Moreover, since this information, *parameters* or annotations had to be available from the ATL program that executes the transformation and considering that we were in a MDE context, the best option was to use another model (and thus to define a new metamodel): an annotation model. The idea behind the use of model annotations for model transformation is the following: suppose we have a source and a target metamodel, a terminal model conforming to the former and the corresponding model transformation. Then, for each annotation model used to execute the transformation, different target models are generated.

Finally, instead of defining a completely new metamodel for our annotation models, we use a weaving model to annotate the input model. A *Weaving Model* is a special kind of model used to establish and handle the links between models elements [1]. This model stores the links (i.e., the relationships) between the elements of the (from now on) *woven* models.

To create and handle the weaving models used in this work we have used the ATLAS Model Weaver (AMW). The model weaver workbench provides a set of standard facilities for the management of weaving models and metamodels [8]. Moreover, it supports an extension mechanism based on a Core Weaving Metamodel [7]. The Core Weaving metamodel contains a set of abstract classes to represent information about links between model elements. Typically, these classes are extended to define new weaving metamodels for specific contexts

So, we extended the aforementioned core weaving metamodel to obtain a new weaving metamodel for annotating Extended Use Case models. The weaving models conforming to the new metamodel serve as the annotation models for the execution of the model transformation. The process is summarized in the picture below.

For each execution of the ATL program, i.e., for each source model, we define a weaving model that conforms to an annotation extension of the core weaving metamodel. The weaving model contains a set of annotations that represent the information needed to execute the transformation, that is, the *parameters* used by some of the rules of the ATL program that encodes the mapping. Both, the source (the woven model) and the weaving model are taking as inputs to generate the output model. This way, given an Extended Use Case model, we can generate different Service Process models depending on the weaving/annotation model attached.

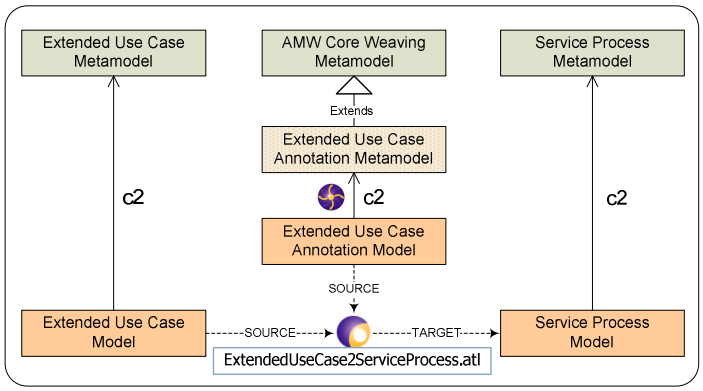


Fig. 2. Using weaving models to annotate Extended Use Case models

The following section sets out how this approach has been applied in a concrete scenario. The complete case study can be downloadable as an Eclipse project [22].

3 Case Study

To illustrate this work we use the models from a conference management system we have developed

3.1 The metamodels

As mentioned before here we address the mapping between the Extended Use Case metamodel (a simplified version of the UML 2.0 Use Case metamodel) and the Service Process metamodel (a simplified version of the UML activity package). To carry out this task we define a new extension of the core weaving metamodel for annotating Extended Use Case models: the Extended Use Case Annotation metamodel. Here we will introduce just the later since the formers can be found at [22].

Extended Use Case Annotation metamodel. If we want to use an Extended Use Case model as input in a model transformation, we need some extra information. For instance, if a use case includes two or more use cases, the order in which they should be executed in the including use case should be specified, since the mapping rule for <<*include*>> relationships [5] gives the two options showed in the picture below.

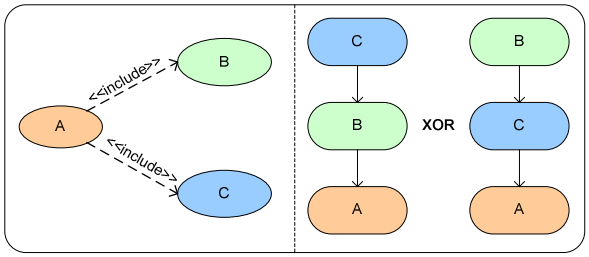


Fig. 3. Include Relationships mapping rule

As already mentioned, these data are not relevant to the model itself, so we use another model to collect them. More specifically, since these data represents relations between the elements of the Extended Use Case model, we use a weaving model. This process is known as annotation and the weaving model is known as the annotation model. Then, each link in the weaving model represents an annotation for the woven model. All this given, Fig. 4 shows the new weaving metamodel for annotating Extended Use Case models.

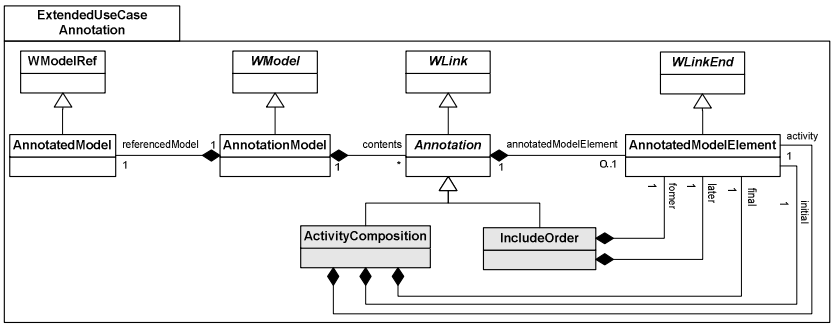


Fig. 4. Weaving metamodel for annotating Extended Use Case models

Each annotated use case in the Extended Use case model will be represented by an *AnnotatedModelElement* in the weaving model. We distinguish two types of annotations.

The *IncludeOrder* element helps in the mapping of several *include* relationships attached to the same use case. It relates the included use cases in groups of two, defining the order in which they should be executed (*former* and *later AnnotatedModelElement*). This information is used in the transformation rule that maps use cases to service activities to know which service activity has to precede the other.

On the other hand, the *ActivityComposition* elements serve to identify which use cases correspond to complex services and thus have to be mapped to activities in the Service Process models (*activity AnnotatedModelElement*) and which use cases have to be mapped to the initial and the last service activity for each activity (*initial* and *final AnnotatedModelElement*).

3.2 The models

The source model. Here we will refer just to the Extended Use Case model. As a matter of fact there will be two source models, this one (the woven model) and the annotation model, that can be thought of as an auxiliary source model. The model is showed in the left-hand side of the figure below.

The Web system for conference management offers three different services: *Submit an Article*, *View Submitted Articles* and *Edit Author Data*. In order to provide with this complex services,

some basic services are needed, as the *Log-In* or the *Register* ones. To model the relation between the different Use Cases two types of associations are used: <<*include*>> and <<*extend*>>. The former implies that the behavior of the included Use Case is contained in the including Use Case, while the later specifies how and when the behavior defined in the extending use case can be inserted into the behavior defined in the extended use case.

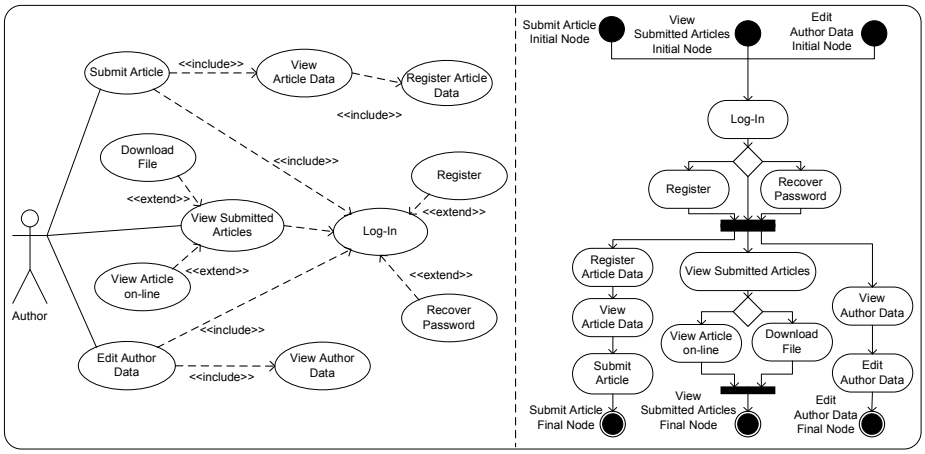


Fig. 5. Extended Use Case model and Service Process models for the conference management Web system

The target model. The Service Process model for our case study is shown in Fig. 5 (right-hand side). Each complex service identified in the previous model is mapped to an activity, while the basic services that it uses are represented as service activities. This way, we have three different activities that use a set of service activities. For instance the Log-In service activity is used by the three activities, Submit Article, View Submitted Articles and Edit Author Data.

Notice that the previous model (Extended Use Case model) did not show which the complex services were, those that had to be mapped to activities, or the order in which included use cases had to be executed. This kind of information is not conceptually relevant to be part of the corresponding metamodel. So it will be collected in the annotation model.

The annotation model. We have to annotate the *main* source model (i.e. the Extended Use Case model) to provide with the additional information needed to execute the transformation. To that end, we use the AMW tool to create a new weaving model conforming to the Extended Use Case annotation metamodel.

As mentioned before, a weaving model conforming to this metamodel includes two different types of *WLinks* (in fact, of annotations): *ActivityComposition* and *IncludeOrder*. The former serves to identify which of the use cases from the Extended Use Case model correspond to a Service provided by the system, as well as the entry and exit points to carry out the service. The later helps to solve the problem about the mapping of several include relationships attached to the same use case. The annotation model for the case study is showed in Fig. 6 (the whole process for obtaining the model as well as the metamodel is explained in detail in [22]).

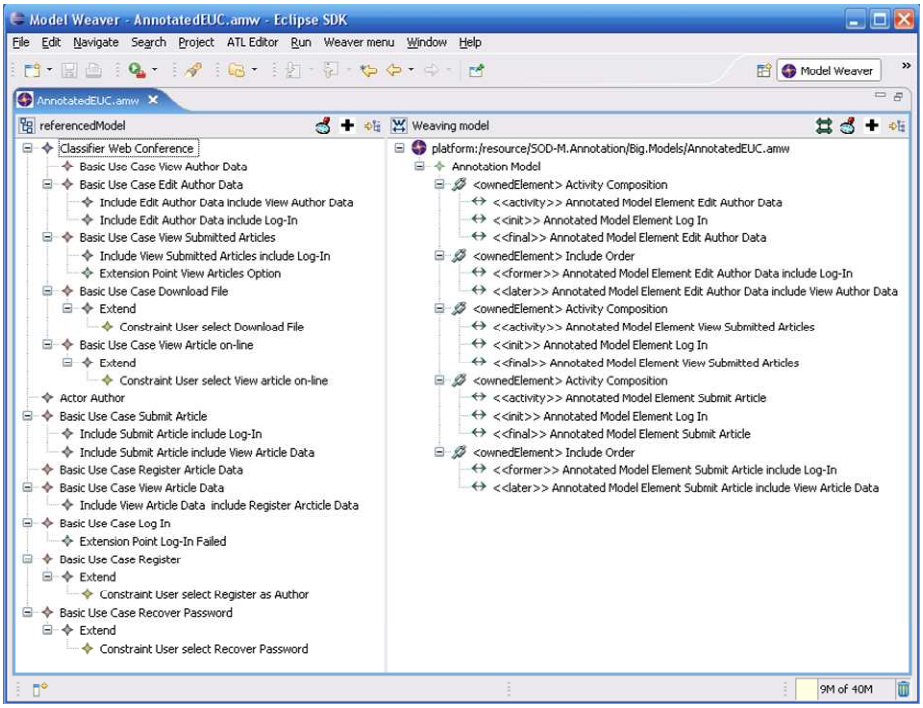


Fig. 6. Extended Use Case model and Extended Use Case Annotation model for the conference management Web system

We add an *Activity Composition* annotation for each service provided by the system. It serves to identify the complex services provided by the system, as well as its entry and exit points. For instance, the first one identifies the Edit Author Data as a complex service. It means that the corresponding Service Process model has to include an Edit Author Data activity. Moreover, the <<init>> and <<final>> annotations indicates that the Edit Author Data service starts by Log-In in the system and finishes by Editing the Author Data.

In the same way, we add an *Include Order* annotation for each pair of include relationships attached to a same use case. The annotation defines the order in which the included use cases should be executed to carry out the including use case. For example, according to the Extended Use Case model, both the Log-In and the View Article Data use cases are included in the Edit Author Data use case. However, there is no way to identify which one has to be completed first. To that end, the annotation (or weaving) model contains an *Include Order* annotation setting that the Log-In basic use case must precede the View Article Data use case. Therefore, as Fig. 6 shows, the Edit Author Data activity includes the Log-In and View Author Data service activities in the right order.

3.3 Using the annotations to customize the transformation

Once we have defined the *main* input model (the Extended Use Case model) and the *auxiliary* input model (the annotation model), it is time to code the model transformation program. Here we will focus just in showing an example of how to use the information provided by the annotation model. In this point some skills in the coding of model transformations are supposed to the reader.

The annotation of the input model allows us to add the missing data we need to execute the transformation. In order to use this information, we just have to include some helpers (auxiliary

functions) in the ATL program. For instance, to map an *include* object we have to know if it is related with other *include* objects (remember the ambiguity about the mapping of include relationships captured in Fig. 3). Thus, we define the following helper. It navigates the weaving model (the annotation model) looking for *IncludeOrder* annotations, whose *former* element is the same that was being mapped when the helper was invoked. To identify the *include* object, the *_xmiID_* property is used. It serves as the identifier of a woven element.

```
-- Returns the link in the Weaving model refering to the 'inc' ServiceProcess!Include
helper context ExtendedUseCase!Include def: getIncludeOrderLink() : AMW!IncludeOrder =
        AMW!IncludeOrder.allInstances()->asSequence()->
                select(link | link.former.element.ref = self.__xmiID__)->first();
```

Then, we define two different rules for mapping *include* objects: one for those not related with other include objects and one for those related. We include a guard in those rules to distinguish which rule should be used in each specific case. The guard invokes the helper we have just showed to make the decision.

```
-- Normal mapping of Include relationships
rule IncludeSimple2ServiceProcess {
        from
                inc : ExtendedUseCase!Include(inc.getIncludeOrderLink().oclIsUndefined())
        to
                edge : ServiceProcess!ControlFlow (
                        name <- ('[' + inc.addition.getFinalActionName() + ' to '+ inc.includingCase.name + ']'),
                        source <- inc.addition.getFinalAction(),
                        target <- inc.includingCase
                )
}

-- Mapping of Include relationships involved in a multi-option selection
rule Include2ServiceProcess {
        from
                inc : ExtendedUseCase!Include(not inc.getIncludeOrderLink().oclIsUndefined())
        to
                edge : ServiceProcess!ControlFlow (
                        name <- ('[' + inc.getSourceNodeName() + ' to ' +
                                        inc.getIncludeOrderLink().getNextInclude().addition.getPrevUseCase().name + ']'),
                        source <- inc.addition.getFinalAction(),
                        target <- inc.getIncludeOrderLink().getNextInclude().addition.getPrevUseCase()
                )
}
```

As the rules show, in both cases the *include* object is mapped to an *Edge* object. The difference comes in which is the target of the edge. If the include object is not related to other include objects, the target of the corresponding edge will be the service activity that maps the including use case (identified by *inc.includingCase* property). Otherwise, the edge target will be the Service Activity that maps the use case linked to the include object that was set to be subsequent to the one that is being mapped. To identify that service activity we code a new helper that given am *InludeOrder* link, returns the reference to the *later* include.

```
-- Given the 'self' IncludeOrderLink, it returns the 'include' referred as next by the link
helper context AMW!IncludeOrder def: getNextInclude() : ExtendedUseCase!Include =
        ExtendedUseCase!Include.allInstances()->asSequence()->select(inc | inc.__xmiID__ = self.later.element.ref)->first();
```

Concluding this section, we can say that we are able to customize the execution of the model transformation and consequently obtain different outputs, by modifying the annotation or weaving model used to run the transformation.

4 Conclusion

This work has presented a technique to address the development of model transformations in the context of Model-Driven Web Engineering (MDWE). The special nature of the models handled at the initial stages of a MDWE process, used to model the business logic, complicates the coding of valid transformations. On the one hand, the gap between models uses to be bigger than usual. On

the other hand, model transformations are generic by nature (they have to work for any terminal model conforming to a given metamodel). To fill the afore-mentioned gap we propose to introduce some design decisions to drive the transformation. This way, we keep generic nature of the model transformation, since we are just *parameterizing* it. Thus, we produce a different target model depending on the design decisions taken.

In this study we have showed how weaving models can be used as annotation models to introduce design decisions by the time of executing a model transformation. We have applied this technique to develop the mapping from the Extended Use Case model to the Service Process model of SOD-M, a service-oriented development method for Web Information Systems. To that end, we have defined a new weaving metamodel for annotating Extended Use Case models. Next, we have coded a model transformation that takes as input, not only the desired Extended Use Case model, but also a weaving model that annotates the former. Then, depending on which the annotation model is we obtain a different Service Process model. As a result, we are able to customize the mapping process and keep the generic nature of the model transformation without polluting the source model.

This technique contributes to improve the accuracy of the models used at the initial stages early stages of WIS development and can help to increase the quality of the models built as well as the subsequent code generated from them. These activities are especially important in proposals aligned with MDA (like SOD-M) because it proposes the models to be used as a mechanism to carry out the whole software development process.

The approach presented here can be easily generalized to other domains and contexts. The proof of concept has been achieved and the results are available as open source. We may qualify this technique of simple or complex according to the point of view. In comparison with the benefits it brings, the solution may be considered as quite simple as has been discussed in the paper. This is mainly due to the simplicity, the genericity and power of the AMW tool and its good coupling with the ATL model transformation solution.

At the present time we are working in the integration of this model transformation process in M2DAT (MIDAS MDA Tool), a case tool which integrates all the techniques proposed in MIDAS for the semiautomatic generation of Service-Oriented applications. The tool is now under development in our research group and its early functionalities have been already presented in previous works [22]. Besides, the open issue of automating the rest of mappings in the MIDAS methodology using the approach presented here is been tackled.

Acknowledgments. This research is partially granted by the GOLD project financed by the Ministry of Science and Technology of Spain (Ref. TIN2005-00010) and the M-DOS project (URJC-CM-2007-CET-1607) cofinanced by the University Rey Juan Carlos and the Regional Government of Madrid.

5 References

1. Bernstein, P A. Applying Model Management to Classical Meta Data Problems. First Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA. 2003.
2. Bézivin, J.: In search of a Basic Principle for Model Driven Engineering. Novatica/Upgrade. V, 21-24 (2004).
3. Bézivin, J.: Some Lessons Learnt in the Building of a Model Engineering Platform. 4th Workshop in Software Model Engineering (WISME), Montego Bay, Jamaica (2005).
4. Caceres, P., De Castro, V., Vara, J.M., Marcos, E.: Model transformations for hypertext modeling on web information systems. ACM Symposium on Applied computing (SAC). ACM Press, Dijon, France (2006) 1232-1239.

5. De Castro, V., Vara, J.M., Marcos, E. Model Transformation for Service-Oriented Web Applications Development. 7th International Conference on Web Engineering. July 2007.
6. De Castro, V., Marcos, E., López-Sanz M.: A Model Driven Method for Service Composi-tion Modeling: A Case Study. Int. Journal of Web Engineering and Technology. 2006 - Vol. 2, No.4, pp. 335 - 353.
7. Didonet Del Fabro, M.: Metadata management using model weaving and model transformation. Ph.D. thesis. University of Nantes (2007).
8. Didonet Del Fabro, M., Bézivin, J. and Valduriez P. (2006). Weaving Models with the Eclipse AMW plugin. Eclipse Modeling Symposium, Eclipse Summit Europe, Esslingen, Germany. 2006.
9. Flore, F.: MDA: The Proof is in Automating Transformations between Models. In OptimalJ White Paper, pages 1–4, 2003.
10. Gerber, A., Lawley, M., Raymond, K., Steel, J., Wood, A.: Transformation: The Missing Link of MDA. International conference on Graph Transformation (2002) 90-105.
11. Harmon, P.: The OMG's Model Driven Architecture and BPM. Newsletter of Business Process Trends (May 2004). http://www.bptrends.com/publications.cfm
12. Jouault, F., Kurtev, I.: Transforming Models with ATL. Satellite Events at the MoDELS 2005 Conference (2006) 128-138.
13. Mellor, S., Scott, K., Uhl, A., Weise, D.: Model-Driven Architecture. In: Advances in Object-Oriented Information Systems. pp. 233-239 (2002).
14. Moreno, N., Romero, J., Vallecillo, A.: An Overview of Model-Driven Web Engineering and the MDA. In: Web Engineering: Modelling and Implementing Web Applications. pp. 353-382 (2008).
15. Murzek, M., Kramler, G.: Business Process Model Transformation Issues - The Top 7 Adversaries Encountered at Defining Model Transformations. In: International Conference on Enterprise Information Systems (2007) 144-151.
16. OMG. MDA Guide V1.0.1. Miller, J., Mukerji, J. (eds.) Document Nº omg/2003-06-01 (2001). Accessible in: http://www.omg.org/cgi-bin/doc?omg/03-06-01.
17. Strommer, M., Murzek, M., Wimmer, M.: Applying Model Transformation By-Example on Business Process Modeling Languages. Advances in Conceptual Modeling -Foundations and Applications (2007) 116-125.
18. Papazoglou, M. et al.: Service-Oriented Computing: A Research Roadmap. In: Dagstuhl Seminar Proceedings (2006).
19. Rozenberg, G.: Handbook of graph grammars and computing by graph transformation: Volume I. Foundations. World Scientific Publishing Co., Inc., River Edge, NJ, USA (1997).
20. Selic, B.: The Pragmatics of Model-Driven Development. IEEE Software. 20, 19-25 (2003).
21. Vara, J.M., Vela, B., Cavero, J.M., Marcos, E.: Model transformation for object-relational database development. SAC '07: Proceedings of the 2007 ACM symposium on Applied computing (2007) 1012-1019.
22. Vara, J.M., Didonet Del Fabro, M.: Web Applications Modelling. ATL Use Cases. http://www.eclipse.org/m2m/atl/usecases/webapp.modeling/.
23. Varró, D., Pataricza, A.: Generic and Meta-Transformations for Model Transformation Engineering. UML 2004 - The Unified Modeling Language. Model Languages and Applications. 7th International Conference, Vol. 3273. Springer (2004) 290-304.
24. Verner, L.: BPM: The Promise and the Challenge. Queue of ACM Vol. 2, Nº 4 (2004) 82-91.

ESTUDIO EMPÍRICO DE LA SITUACIÓN DEL GOBIERNO DIGITAL MUNICIPAL COSTARRICENSE

Arnoldo Zambrano Madrigal[1], Cristina Cachero Castro[1] , Mª Ángeles Moraga[2]

[1] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Alicante, España
azambram@hotmail.com
ccachero@dlsi.ua.es

[2] Instituto de Sistemas y Tecnologías de la Información, Universidad de Castilla-La-Mancha, España
MariaAngeles.Moraga@uclm.es

RESUMEN

Es un hecho reconocido cómo la universalización de Internet y la WWW está influyendo en el modo en que los gobiernos proporcionan servicios e interaccionan con los ciudadanos. Esta nueva forma de interacción es especialmente relevante en países con poblaciones dispersas y barreras geográficas importantes, como es el caso de Costa Rica. Sin embargo, el estado del Gobierno Digital en las municipalidades costarricenses está todavía en un estado incipiente. Es por ello que es especialmente relevante desarrollar estudios detallados tanto de las demandas de los ciudadanos como de los factores que están impidiendo que éstas sean cubiertas, para a partir de estos datos proponer soluciones. En este artículo se presenta un estudio de estas características, que ha servido de base para la elaboración, en el seno del proyecto nacional *FOMUDE*, del conjunto de requisitos funcionales que deben cubrir los portales municipales costarricenses si pretenden satisfacer la demanda ciudadana. Estos requisitos han sido además clasificados de acuerdo con el mínimo nivel de digitalización necesario para considerarlos satisfactorios. Además, los resultados de este estudio han servido para detectar un conjunto de riesgos asociados a las condiciones actuales, tanto técnicas como organizativas, de las municipalidades. Para paliar estos riesgos, proponemos una serie de recomendaciones orientadas a incrementar las probabilidades de éxito del proyecto.

PALABRAS CLAVES

Municipalidad, Portal web municipal, Accesibilidad, Gobierno Digital Municipal, Gobierno local.

1. INTRODUCCIÓN

Los avances tecnológicos que ha experimentado la sociedad en los últimos años [8] y principalmente la implantación generalizada de Internet, ha abierto todo un nuevo abanico de posibilidades en cuanto al modo en que los gobiernos proporcionan servicios e interaccionan con los ciudadanos, y ha acuñado el término Gobierno Digital (E-Government). Según la definición de Gobierno Digital formulada en [9], es el uso de las Tecnologías de la Información y Comunicación (TIC) para promover un gobierno más eficiente y eficaz, facilitando y permitiendo mayor acceso a los servicios públicos por parte de los ciudadanos, así como la prestación de servicios a través de Internet, teléfono, centros comunitarios, dispositivos inalámbricos u otros sistemas de comunicaciones [10] [12].

La implantación del Gobierno Digital es especialmente relevante en países con poblaciones dispersas y/o barreras geográficas importantes [9] como es el caso de Costa Rica, cuya geografía cubierta de amplias zonas boscosas y un clima extremo dificulta el acceso presencial a los gobiernos locales. En estos países, y en Costa Rica en particular, la difusión de las tecnologías de información y comunicación (TIC's) han redefinido las expectativas de los ciudadanos acerca del gobierno y sus servicios [1] [5] [6] [14]. Esto ha obligado a que las autoridades nacionales busquen nuevas y mejores alternativas para ofrecer servicios ágiles y oportunos a los ciudadanos por medio de Internet. Para ello, el Gobierno de la República a través de su Secretaría Técnica de Gobierno Digital (www.gobiernofacil.go.cr), ha implementado durante los últimos años un ambicioso plan de trabajo que ha permitido la conexión de las distintas instituciones públicas con la ciudadanía por medio de un único portal que ha propiciado un nuevo modo de gestión administrativa, que implica entre otras cosas la oferta de más y mejores servicios a la población, así como la mejora de la eficiencia, la transparencia y la reducción de costos.

Sin embargo, este cambio en el modo de interacción del ciudadano con el gobierno nacional no se ha trasladado a las municipalidades, lo que está causando numerosos retrasos y quejas entre la población. Conscientes de este hecho, el Ministerio de Planificación (MIDEPLAN) y el Instituto de Fomento y Asesoría Municipal (IFAM) , instituciones a cargo de velar por el desarrollo de los municipios costarricenses, han abordado un Proyecto de Fortalecimiento Municipal y Descentralización (FOMUDE) cuyo principal objetivo es el fortalecimiento del Gobierno Digital Municipal (GDM) [15][16]. Para ello, FOMUDE [4] aboga por la definición y coordinación de distintas tareas fundamentales, que actualmente están siendo llevadas a cabo por distintos equipos de trabajo y que pretenden ser puestas a disposición de la ciudadanía a través de un portal único. Estas tareas incluyen (ver Figura 1):

- El desarrollo del Sistema Integrado de Administración Tributaria Municipal (SITRIMU) cuyo objetivo es la recaudación de impuestos de bienes inmuebles.
- El desarrollo del Sistema Bid-Catastro que permite la digitalización de planos de catastro de las propiedades ubicadas en las distintas partes del territorio nacional y que deben ser administradas por los gobiernos locales.
- La adquisición del Sistema Nacional de Pagos Electrónicos (SINPE) que logrará la transferencia de fondos entre distintas entidades bancarias lo que a su vez hará que el usuario pueda cancelar sus tributos sin importar dónde tenga su cuenta bancaria.
- El desarrollo del plan de capacitación bajo modalidad Elearning para aquellos funcionarios municipales que por razones de tiempo, ubicación geográfica y costos de desplazamiento se les dificulta obtener dicha actualización de conocimientos.
- La interconexión con el Sistema Clasificado de Puestos Municipales, sistema informático que la Dirección General de Servicio Civil tiene planeado desarrollar próximamente y que permitirá definir y consultar las clases de puestos de trabajo ofertados por las municipalidades.

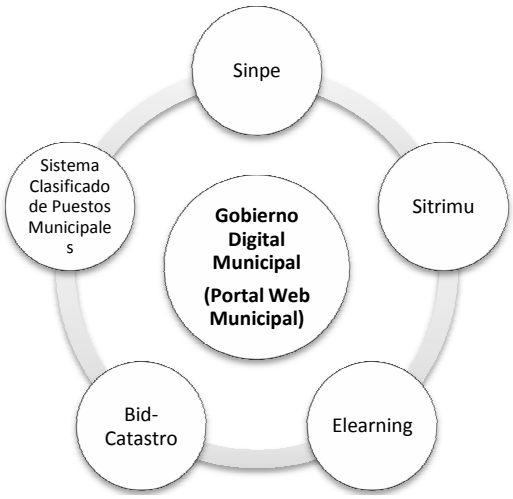


Figura 1: Proyectos contemplados para formar parte de un Gobierno Digital Municipal.

Además, el desarrollo adecuado de este proyecto requiere contar con los ciudadanos que van a interactuar con él: por un lado es necesario saber cuáles son sus expectativas con respecto a los portales municipales, y en qué medida éstas se alinean con cada una de las áreas del proyecto. Por otro lado, es necesario prever qué barreras existen que pueden dificultar el éxito del proyecto. Con el fin de resolver estas dos cuestiones, en la Sección 2 se presenta el diseño de un estudio empírico, orientado a conocer el estado real de las municipalidades costarricenses en materia digital. La Sección 3 presenta los resultados del estudio, resultados que han servido de base para presentar, en la Sección 4 recomendaciones tras analizar detalladamente la información recabada. Por último en la sección 5 se ofrecen las conclusiones orientadas a facilitar que las municipalidades realicen las mejoras necesarias y así logren incorporarse con garantías de éxito a un GDM.

2. ESTUDIO EMPÍRICO DEL ESTADO DEL GOBIERNO DIGITAL EN LAS MUNICIPALIDADES

De acuerdo con [2] el método experimental tiene como finalidad establecer relaciones causales que sirven de explicación entre los hechos observados y los factores que los producen.
Dentro de los métodos experimentales, la encuesta [11] [13] puede conceptualizarse como una estrategia de investigación, es decir, "un procedimiento que implica un proceso completo de investigación, que abarca la formulación del problema, el establecimiento de los objetivos, la selección de los sujetos y el diseño y análisis de los datos". Las diferentes técnicas de encuestas tienen características comunes, como: (1) recogen información de una porción de la población de interés, dependiendo del tamaño de la muestra en el propósito del estudio; (2) la muestra es seleccionada científicamente de manera que cada persona en la población tenga una oportunidad medible de ser seleccionada (de esta manera los resultados pueden ser proyectados con seguridad, de la muestra a la población mayor); (3) la información es recogida usando procedimientos estandarizados, de modo que a cada individuo se le hacen las mismas preguntas en más o menos la misma manera; (4) la intención de la encuesta no es describir los individuos particulares quienes, por azar o algún otro criterio de selección, son parte de la muestra; sino obtener un perfil compuesto de la población. De manera general, se puede decir que las encuestas son una fuente importante de conocimiento científico básico y pueden ser clasificadas por su método de recolección de datos (por correo, telefónica, mediante entrevista personal, por internet, etc.).

La siguiente tabla muestra el proceso experimental que debe seguirse para la elaboración y ejecución de un cuestionario. [7]

Tabla 1. Proceso experimental para elaboración y ejecución de cuestionarios.

Fase	Actividades involucradas
Definición de requisitos	¿Por qué, dónde y con qué objetivos la encuesta ha sido diseñada?
Diseño de planificación	Cómo va a ser llevada a cabo la encuesta: definición de parámetros de contexto, formulación de hipótesis, diseño de experimento, definición del proceso de recolección de datos, selección de las técnicas de análisis, instrumentación
Ejecución de la entrevista	Recogida de datos, motivación de los participantes, validación de datos, presentación de informes
Análisis de los datos	Descripción de datos, reducción del conjunto de datos, pruebas de las hipótesis
Agrupamiento de resultados	Interpretación de resultados, identificación de amenazas, inferencia, identificación de lecciones aprendidas

En nuestro caso, el objetivo de la encuesta que hemos realizado ha sido doble: por un lado, se pretendían identificar las necesidades de los ciudadanos en materia de gobierno digital; por otro lado, se querían identificar las barreras existentes para la implantación de dicha forma de gobierno.

Dado lo limitado de la población (89 municipalidades y 13 federaciones), se planeó la ejecución de la encuesta por parte de todas las instituciones involucradas en el proceso. Para ello se designó una persona de contacto (enlace) en cada municipalidad. Los requisitos para la selección de enlaces fueron (1) vivir en la zona y (2) llevar un mínimo de 5 años trabajando en la institución. Con ello garantizamos un conocimiento detallado de la situación particular de cada institución por parte del enlace.

Para el estudio se definieron cinco variables: 1-.Grado de desarrollo actual de la web municipal, 2-.Servicios ofertados a la ciudadanía por parte de los municipios costarricenses, 3-. Cumplimiento de la Ley 7600 sobre accesibilidad, 4-.Servicios requeridos por parte de la ciudadanía y 5-. Grado de especialización del personal técnico contratado por la municipalidad. Esta selección de variables se baso en la importancia de estos de cara a determinar las posibles barreras objetivas [9] a la implantación del GDM. Por un lado, el grado de desarrollo de las webs municipales (incluidos sus servicios y su nivel de accesibilidad) influye directamente en el esfuerzo necesario para introducir las prácticas de Gobierno Digital en las municipalidades. Por otro lado, una inadecuada respuesta a las demandas de los ciudadanos puede limitar la usabilidad y credibilidad del sistema. Por último, no podemos olvidar que el factor humano es básico de cara a desarrollar y mantener los sistemas implantados, por lo que es necesario asegurar su correcta formación. En este estudio es importante hacer notar que los factores de infraestructuras, también barreras objetivas importantes en Costa Rica, no han sido tenidos en cuenta en este estudio debido a que existe una acción paralela orientada a garantizar unos requisitos tecnológicos mínimos en todas las municipalidades.

Para cada una de estas variables se formularon un conjunto de preguntas que pueden ser vistas en la Tabla 2.

Tabla 2. Clasificación de preguntas del cuestionario a las municipalidades.

VARIABLES	PREGUNTAS
Grupo 1 (Posicionamiento web municipal)	¿Posee su institución un sitio web?
Grupo 2 (Servicios ofertados a la ciudadanía por parte de los municipios costarricenses)	¿Su sitio web oferta servicios de tipo (Informativa básica, Informativa, Interactiva, Transaccional, inter-relacionada [1])?
Grupo 3 (Cumplimiento de la Ley 7600 sobre accesibilidad)	¿Conoce usted las reglas de accesibilidad, conocidas como W3C, para personas discapacitadas?
	¿Lo aplica actualmente en su sitio web?
Grupo 4 (Servicios requeridos por parte de la ciudadanía)	¿Sabe usted que tipos de servicios estarían interesados en tener sus usuarios?
	Por favor indicar los servicios que brinda su institución.
Grupo 5 (Recurso humano especializado)	¿Posee su institución Área de Tecnologías de Información (TI)?
	¿Cuántas personas componen TI?
	¿El diseño, desarrollo y mantenimiento del sitio web es realizado por outsourcing o diseño propio?
	¿Número de funcionarios de la institución que dan mantenimiento al sitio web?

La encuesta fue validada por un equipo de expertos que, tras revisar la formulación de las preguntas y las variables consideradas en el cuestionario, sugirió una serie de mejoras que fueron tenidas en cuenta a la hora de elaborar el cuestionario final.

3. RESULTADOS DEL ESTUDIO EMPÍRICO DEL GOBIERNO DIGITAL EN LAS MUNICIPALIDADES

Una vez recibidas y procesadas las respuestas que suministraron los distintos enlaces existentes en cada una de las municipalidades, obtuvimos un total de 77 respuestas, lo que supone un 75 % de tasa de respuesta. A continuación ofrecemos los resultados de dicho estudio.

3.1 Municipalidades con representación web

En la siguiente figura se puede notar el claro incumplimiento de la orden girada por el Ministerio de Ciencia y Tecnología [17], sobre la obligación que tienen todas las instituciones estatales de poseer un sitio web para ofrecer sus servicios al ciudadano. Únicamente 33 municipalidades, de un total de 77 encuestadas, cumplen la orden del MICIT, eso indica un 43% de representatividad web municipal.

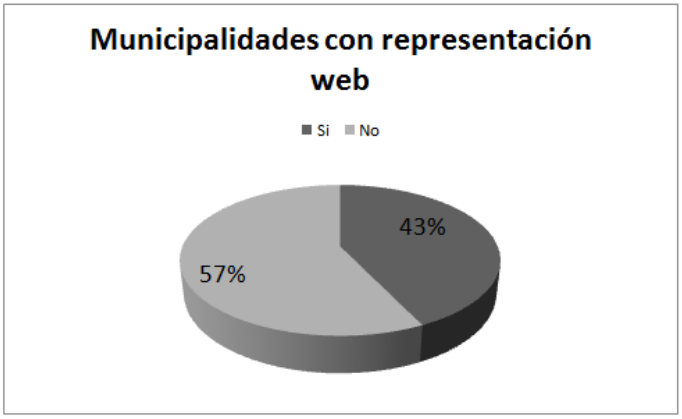


Figura 2. Representatividad web municipal

3.2 Servicios ofertados por los sitios web municipales

Según [1], el grado de implantación del Gobierno Digital no sólo depende del número de servicios ofertados si no que también de su nivel de desarrollo. El estudio distingue cinco niveles (informativa básica, informativa, interactiva, transaccional, interrelacionada). La tabla 3 muestra los resultados sobre el nivel de digitalización de los servicios brindados por las municipalidades, sus sitios web desempeñan labores informativas demasiado básicas (74 % ubicados en el primer y segundo nivel).

Tabla 3. Tipos de servicios de los sitios web municipales

Municipalidad	N1	N2	N3	N4	N5	Total	Municipalidad	N1	N2	N3	N4	N5	Total
Santa Ana	1	1				2	Goicoechea	1					1
Coronado		1				1	Buenos Aires	1	1				2
Tibas	1					1	Belén		1	1			2
Montes de Oca		1	1			2	Santa Barbara	1					1
Moravia		1				1	San Rafael	1	1				2
Puntarenas	1	1				2	Heredia		1				1
Esparza		1		1		2	San Isidro	1					1
Carrillo	1	1				2	Coto Brus	1	1	1			3
Santa Cruz	1					1	Liberia	1	1	1	1		4
Naranjo		1				1	Monteverde	1		1			2
San Carlos		1	1			2	Osa	1			1		2
San Ramón			1		1	2	La Unión		1	1			2
Cóbano	1	1				2	Turrialba	1	1	1			3
Santo Domingo		1	1			2	Golfito			1			1
Grecia		1				1	Cartago		1				1
Cañas	1	1				2	Total	19	24	11	3	1	58
Limón	1	1				2	%	33	41	19	5	2	100
Pérez Zeledón	1	1				2							

3.3 Accesibilidad para la ciudadanía con discapacidad

Costa Rica, por medio de la Ley 7600 sobre Igualdad de Oportunidades para Personas con Discapacidad [3], establece que los ciudadanos con discapacidad deben tener las mismas posibilidades de acceso a la información contenida en sitios web estatales. La siguiente gráfica demuestra que esto no se está aplicando con las municipalidades debido a que de las 33 instituciones que indicaron poseer sitio web, el 100% no aplica la W3C garantizando así la navegabilidad de los ciudadanos con discapacidad a través de sus sitios web.

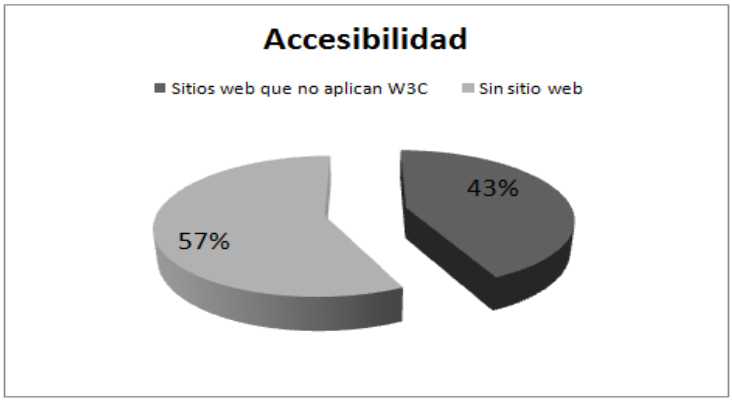


Figura 3. Accesibilidad a sitios web municipales

3.4 Demanda de servicios por parte de la ciudadanía

Conocer qué tipos de servicios demanda la población a las instituciones públicas es un paso importante hacia un GDM eficiente. La siguiente tabla recoge las opiniones expresadas por los usuarios ante consultas realizadas por parte de los municipios.

Tabla 4. Servicio solicitados por los ciudadanos

Pago de impuestos	Descarga de formularios	Foros
Información turística	Trámites por internet	Bolsa de empleo
Verificación de planos para construcción	Agenda municipal	Consulta de tributos

3.5 Obstáculos para el desarrollo de sitios web municipales

Al analizar los datos de las figuras 4 y 5 sobre la cantidad de personal de TI ubicado en las municipalidades y las opciones de mantenimiento a los sitios web municipales, se identifica una razón de su bajo posicionamiento ante la ciudadanía. Aunque la mayoría de instituciones tiene un departamento de informática (91%), el número de personas que integran este departamento es muy bajo, ya que solo se cuenta con una o dos personas que atienden a todo el personal de la municipalidad y además, combina las labores de soporte con el de desarrollo y mantenimiento de sus sistemas. De los 33 municipios que contestaron afirmativamente contar con presencia web, 30 poseen personal informático en las condiciones indicadas anteriormente. Esta razón ha obligado a los municipios a subcontratar dicho mantenimiento por outsourcing, lo cual también ha sido insuficiente debido a que no todas son poseedoras de un presupuesto amplio que garantice dicha labor o debido a que no disponen de un plan de aseguramiento de calidad que permita que los requisitos especificados sean correctos y los productos finalmente entregados cubran las necesidades de las municipalidad. De los 33 municipios con presencia web, 14 aplican outsourcing, 18 atienden el sitio directamente con personal de la municipalidad y sólo en un caso se manejan las dos opciones.

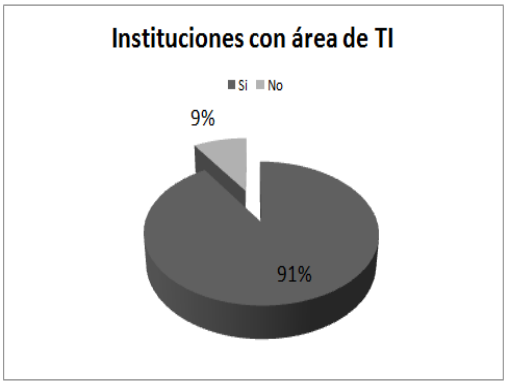


Figura 4. Municipales con RRHH informático

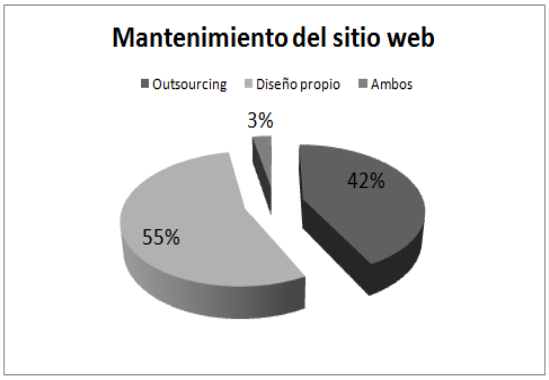


Figura 5. Mantenimiento a sitios web municipales

3.6 Riesgos del estudio y del proyecto

A pesar de lo exhaustivo del estudio (que ha sido realizado sobre el total de la población objetivo), existen algunos riesgos a su validez que conviene considerar. En primer lugar, y a pesar de las restricciones impuestas sobre el enlace municipal seleccionado en cada caso, existen variaciones en cuanto a su grado de experiencia técnica, que puede haber sesgado o introducido errores en las respuestas más técnicas. Otro riesgo se refiere a la identificación de los servicios demandados por los ciudadanos. Estos servicios han sido recabados de manera indirecta a través del enlace de cada municipalidad. Pensamos que una pregunta directa a los ciudadanos habría producido resultados más fiables. Por otro lado, los datos recogidos nos hacen plantearnos ciertos riesgos que hay que tener en cuenta a la hora de llevar a cabo el proyecto. En primer lugar la ubicación geográfica de las municipalidades provoca que no siempre sea fácil la llegada de infraestructuras como es Internet de banda ancha. Por otro lado, la baja presencia de Internet y el escaso número de servicios ofrecidos por parte de las municipalidades puede comprometer la buena acogida por parte de éstas de los nuevos servicios. Para paliar esta situación, pensamos que es necesario que se disponga de personal capacitado y de programas que conciencien a los trabajadores de la necesidad y ventajas de cambiar su modo de trabajo. Además, hay que concienciar a los equipos de desarrollo de la necesidad de contemplar aspectos de usabilidad y accesibilidad en los servicios ofertados.

Por último, hay que tener en cuenta que la implantación de un Gobierno Digital Municipal no sólo requiere personal capacitado y aplicaciones preparadas, sino un ciudadano dispuesto a hacer un uso eficiente de las mismas. Así, aunque en este estudio nos hemos centrado en evaluar a las municipalidades, una evaluación de las condiciones actuales (tanto a nivel de infraestructura como de conocimiento) del usuario objetivo de los portales municipales es vital de cara a asegurar el éxito del proyecto.

4. RECOMENDACIONES

Una vez recibidos, analizados y procesados los datos suministrados por las municipalidades, ofrecemos una clasificación de los nuevos servicios que deben ser incluidos cuanto antes en el

portal web municipal (figura 6), a su vez es importante indicar que dicho portal web ha de estar ubicado en las instalaciones del IFAM, con lo cual se aprovecha la infraestructura tecnológica que tiene dicha institución. Otra recomendación importante es detectar servicios que puedan ser reutilizables en distintas municipalidades. De esta forma se fomenta la reutilización y se ahorran costes y recursos.

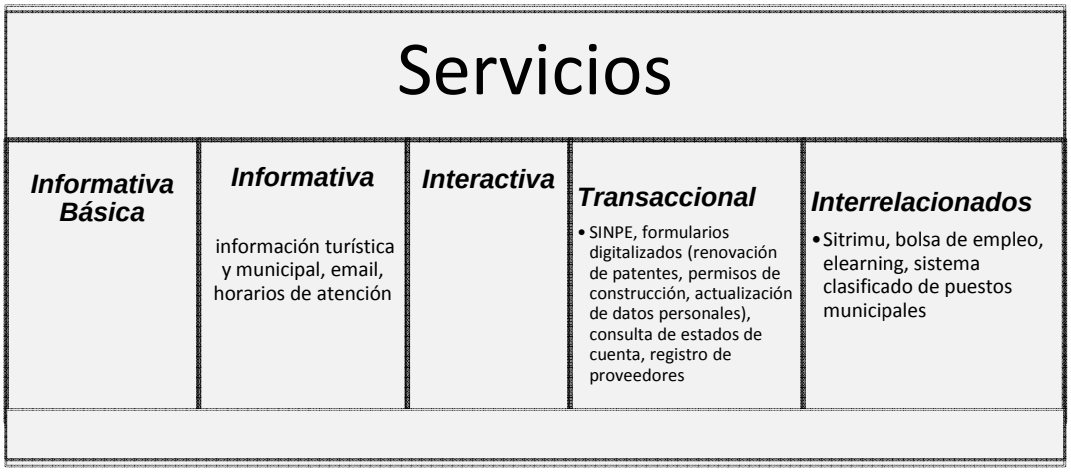


Figura 6. Propuesta de clasificación de servicios para un Portal Web Municipal

5. CONCLUSIONES

El presente estudio apunta a que las municipalidades costarricenses realicen algunas modificaciones en su forma de interactuar con el ciudadano, a continuación detallamos estas:

- Las municipalidades deben invertir tiempo y dinero en mejorar sus centros de tecnologías de información dotándoles del personal necesario que pueda hacer frente a las demandas propias de la institución en materia de soporte técnico, desarrollo y mantenimiento, así como velar por el debido manejo de sus sitios web, por cuanto, esto representa una ventana al mundo y sobre todo permite la prestación de más y mejores servicios a la ciudadanía en general. No es posible que las municipalidades dispongan de uno o dos funcionarios informáticos para atender todas las tareas propias de sus respectivas instituciones.
- Es necesario que los gobiernos locales realicen esfuerzos mayores para incrementar su presencia web y de esa forma lograr llegar a la ciudadanía con mayores servicios.
- En materia de accesibilidad los sitios web municipales poseen grandes problemas, ya que es nula la utilización de estándares y al ser una Ley de la República es urgente que se hagan los ajustes necesarios para dotar a los sitios web municipales con las reglas necesarias que cumplan con los estándares internacionales de accesibilidad [18]. También, es importante capacitar al personal informático de dichas instituciones en el manejo de dichas reglas.
- Como se pudo constatar en la investigación realizada y así fue indicado en este artículo se tienen proyectados varios desarrollos que pretenden ofrecer servicios web a la ciudadanía, razón por la cual es fundamental que las municipalidades tomen las previsiones del caso para elevar el nivel de seguridad tanto física como lógica de sus datos. Además, es

importante capacitar al personal dotándoles del conocimiento necesario que permita poner en práctica técnicas para el manejo seguro de información.

- También, se requiere que las municipalidades cuenten con un ancho de banda adecuado que les provea una buena navegación y de esa forma sus servicios sean ágiles.
- Para FOMUDE, disponer de un resultado que, a partir de la encuesta mencionada, permitiera conocer la situación real que viven las municipalidades, ha sido de vital importancia, debido a que al clarificar el panorama real de los municipios, se pueden fijar objetivos y metas más claras para todos.
- Por último, es fundamental y de acuerdo con los resultados obtenidos en cuanto a los tipos de servicios brindados por las municipalidades poder variar dichos porcentajes y ofrecer servicios de mayor nivel de desarrollo digital. Para ello es importante tomar como punto de partida las etapas de desarrollo de la presencia digital suministradas por las Naciones Unidas, las cuales deben servir como punta de lanza para evaluar todo servicio que en un futuro se desee colocar en el portal web municipal.

AGRADECIMIENTOS

Agradecemos al Ministerio de Ciencia y Tecnología de Costa Rica y al Consejo Nacional para Investigaciones Científicas y Tecnológicas por la gran colaboración que han brindado a este estudio otorgando todas las facilidades necesarias para llevar a cabo esta labor y que se espera contribuya en gran medida al fortalecimiento de los gobiernos locales de Costa Rica.
También, se desea agradecer a doña Florita Azofeifa, Directora del Proyecto FOMUDE por permitirnos participar dentro del equipo de trabajo que dirige.
Este trabajo ha sido parcialmente auspiciado por el Proyecto ESPIA (TIN2007-67078) y por el Proyecto QUASIMODO (PAC08-0157-0668) de Castilla-La Mancha, Ministerio de Educación y Ciencia (España).
Por último, queremos externar un sincero agradecimiento al señor José Joaquín Arguedas Herrera, Director de Servicio Civil, por cuanto ha permitido poder aplicar los conocimientos adquiridos en dicha institución y ha facilitado toda la ayuda necesaria para seguir adelante con este proyecto tan importante.

REFERENCIAS

1. Cubillo Mora Mayela, Guzmán Villarreal Olman, 2006, *Informe final del Proyecto Diagnóstico sobre el Gobierno Digital en Costa Rica*, San José, Costa Rica
2. Chica Alaminos, Costa Castejón J.L: Elaboración, análisis e interpretación de encuestas, cuestionarios y escalas de opinión (2006), Universidad de Alicante (España)
3. Figueres Olsen José María, 1996, Ley 7600 Igualdad de oportunidades para las personas con discapacidad, San José, Costa Rica, pp 1-31
4. Fomude, 2007, Expediente de licitación EuropeAid/125237/D/SUP/CR, San José, Costa Rica, pp 1-108
5. Gil-Garcia J. Ramon, *Exploring the Success Factors of State Website Functionality: an Empirical Investigation,* pp 121-130.
6. Gil-Garcia J. Ramon, 2006, *Enacting State Websites: A Mixed method study exploring E-Government success in multi-organizational settings,* Proceedings of the 39th Hawaii International Conference on System Sciences 2006, Estados Unidos, pp. 1-10
7. Goulao, M. and Brito e Abreu F., Modeling the experimental software engineering process. 6th International Conference on the Quality of Information and Communication Technology. IEEE, 2007.
8. Hans J Scholl: *Employing to Mobility Paradigm: The next big leap in Digital Government?,* pp. 1-2
9. Li Baoling, 2005, *On the barriers to the development of e-government in China,* Xi'an, China, pp. 549.
10. Mendes Emilia, 2005, *A systematic review of web engineering research, pp 498-507.*
11. Martínez Espinosa Yulkeidi: Mejores prácticas para la realización de encuestas Web en línea. In. Universidad de Alicante (2008), Alicante (España)

12. Peters M. Rob, Janssen Marijn, Van Engers M Tom, 2004, *Measuring E-Government Impact existing practices and shortcomings, Icec 04, Sixth International Conference on Electronic Commerce,* pp. 480-489
13. Petricek Vaclav, Escher Tobias, Cox J. Ingemar, Margetts Helen, 2006, *The web structure of E-Government developing a methodology for quantitative evaluation, International World Wide Web Conference 2006,* Edinburgo, Escocia, pp. 669-678.
14. Punia K. Devendra & Saxena K.B.C., 2004, *Managing inter-organizational workflows in EGovernment Services,* pp 500
15. Rojas Molina Fabio, 2008, *Gobierno Digital Municipal,* San José, Costa Rica
16. Vargas Polinaris Jorge, Alfaro Echeverría Juan José, 2004, *Convenio de Cooperación Interinstitucional entre el Ministerio de Planificación y Política Económica y el Instituto de Fomento y Asesoría Municipal para la delegación de la unidad de gestión del Proyecto de Fortalecimiento Municipal y Descentralización según convenio de financiación acordado con la Comunidad Europea*, San José, Costa Rica
17. http://66.102.9.104/search?q=cache:UQKcuF8F8AMJ:www.hacienda.go.cr/centro/datos/Articulo/Factores%2520que%2520inciden%2520en%2520el%2520desarrollo%2520del%2520Gobierno%2520Electr%C3%B3nico.doc+conatic&hl=es&ct=clnk&cd=10&gl=cr
18. http://www.w3c.es/

Verificación de Wrappers Web: Nuevas Ideas

José Luís Arjona[1] y José Luís Álvarez[2] e Iñaki Fernández de Viana[2]

Departamento de Tecnologías de la Información, Universidad de Huelva
Campus Universitario de la Rábida, 21071, Palos de la Frontera (Huelva)
{jose.arjona, alvarez, i.fviana}@dti.uhu.es

Resumen. Actualmente una de las principales fuentes de información es la Web. Lamentablemente la mayoría de sus contenidos están orientados a una interacción humana lo que hace muy difícil su procesamiento automático. Un wrapper es un sistema que simula la interacción humana con la Web e intenta estructurar sus contenidos para así facilitar un posterior procesamiento. Como todo proceso automático puede fallar, por lo que es necesario verificar que la información se va extrayendo adecuadamente. En el presente artículo se exponen diversos métodos de verificación así con un anális de las inconvenientes que estos presentan.

Palabras clave: Verificación, Wrapper, Extracción de Información, Web, Selección de Características, Clustering, Outliers.

1 Introducción

La Web se está convirtiendo, si no lo es ya, en una de las principales fuentes de cualquier tipo de información. De una forma abstracta, se la puede ver como un conjunto de islas de información independientes que muestran sus contenidos siguiendo una serie de estándares web. Cuando hablamos de fuentes independientes de información nos referimos a que es muy habitual que tengamos que recurrir a más de una de estas islas para obtener una información completa sobre nuestro objeto de consulta. Por ejemplo, supongamos que estamos interesados en la compra de algún tipo de dispositivo electrónico y sabemos que la única manera de conseguirlo es comprándolo en las web de una empresa localizada en un país no perteneciente a la Comunidad Económica Europea. Si queremos saber el montante de toda la compra tendríamos que consultar tanto la página de dicha empresa como otra que nos indique los costos arancelarios.

El caso anteriormente expuesto intenta enfatizar que la obtención de información desde la Web es, a día de hoy, prácticamente manual debido a que dicha información está dispersa y a que su representación está orientada al usuario final y no a su procesamiento automático.

Un wrapper web es un sistema que proporciona un interfaz de programación para acceder a los datos proporcionados por islas web simulando la interacción humana. Por lo general, este toma como entrada una consulta, localiza el formulario adecuado [25] [38], lo rellena [23], navega por la página de resultados [17], extráe los atributos de interés [13] y devuelve dichos atributos como un conjunto de resultados.

En principio un wrapper web no es tolerante a cambios, esto es, cualquier modificación en los formularios de consulta, en la forma de navegación o en la forma de mostrar los resultados pueden hacer que los datos que este devuelve no sean correctos. A no ser que la información generada por el wrapper sea verificado de una forma automática, esta podrían pasar desapercibida para las aplicaciones que la utilicen.

Este artículo se organiza de la manera siguiente: primero haremos un resumen del proceso de verificación, luego indicaremos las carencias que hemos detectado en dicho proceso y acabaremos presentando las futuras líneas de investigación.

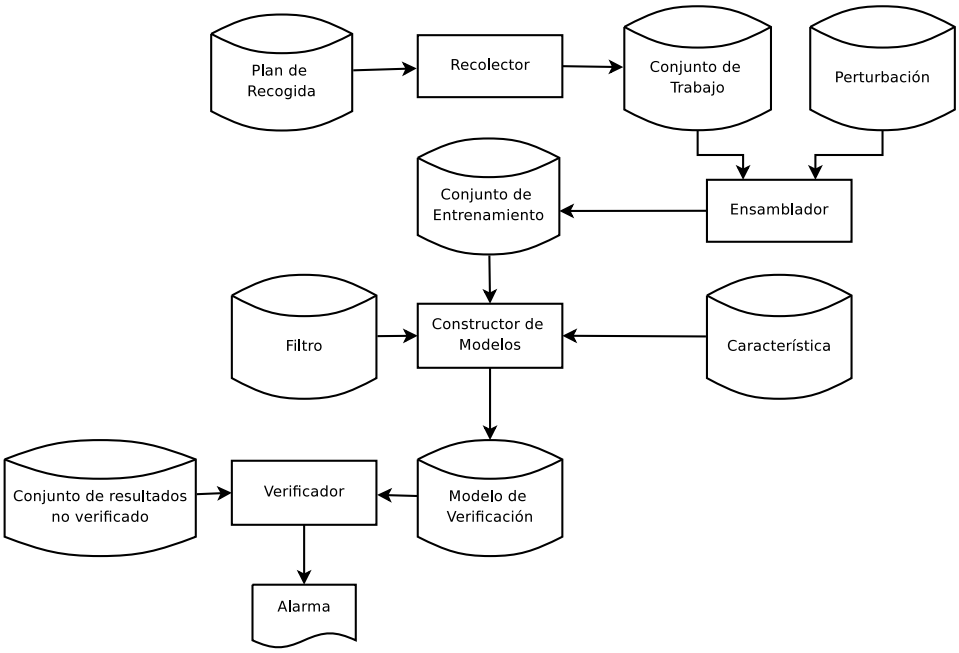


Fig. 1. Marco de trabajo para el desarrollo de un verificador

2 Marco de trabajo para el desarrollo de un verificador

En la figura 1 mostramos el marco de trabajo que usaremos para explicar el funcionamiento de la verificación de wrappers web. A groso modo el proceso empieza invocando al wrapper para obtener un conjunto de resultados que usaremos para formar el conjunto de entrenamiento. De este conjunto de entrenamiento se extráen una serie de características tanto numéricas como categóricas. Dichas características se modelarán y combinarán para que representen al conjunto de entrenamiento. Cuando recibamos un conjunto de resultados a verificar se comprobará si las características que este presenta siguen el modelo obtenido para el conjunto de entrenamiento. En los siguiente puntos describimos con mayor detalle cada uno de los pasos que se siguen en la verificación.

2.1 Obtención de los conjuntos de datos

El primer paso que tenemos que realizar para construir un verificador es obtener un conjunto de datos válidos. El **recolector** ejecuta un plan de recolección que no es más que un conjunto de consultas que serán ejecutadas por el wrapper que es objeto de verificación.

El recolector obtendrá del wrapper un conjunto de datos que los usará para crear el **conjunto de trabajo** que es una estructura que almacena la siguiente información: el wrapper usado, las consultas ejecutadas y los conjuntos de datos obtenidos por cada una de ellas. Estos conjuntos de datos pueden estar formado por un único slot o un conjunto de ellos. El término slot se usa indistintamente para hablar de atributos o registros. Cada uno de estos slots está etiquetado con el nombre de su clase.

2.2 Creación de los conjuntos de entrenamiento

Cada uno de los conjuntos de resultados devueltos por un wrapper deben de ser examinados por un experto que los clasificará como válidos o inválidos. Por tanto, el conjunto de entrenamiento estará formado por dos conjuntos de datos: uno válido y otro inválido.

Como el wrapper fue creado antes que el verificador, todos los conjuntos de datos que aquel devuelve suelen ser válidos. En [34] se demuestra que aprender sólo de casos positivos conlleva un sobreaprendizaje que desemboca en la detección errónea de conjuntos de datos inválido. Es pues necesario generar conjuntos de datos inválidos aplicando una serie de **permutaciones** a los datos que conforman el conjunto de entrenamiento. En [21] se proponen tres tipos de permutaciones denominadas: cambio en el interfaz de consulta, cambio en las fuentes de datos y cambio en la presentación. El problema de estas permutaciones es que el conjunto de datos inválidos que se genera es claramente inválido, demasiado artificial. Otra propuesta descrita en [21] se basa en usar como conjunto de datos inválidos, los datos obtenidos de otras islas de datos semánticamente equivalentes.

2.3 Creación de los modelos de verificación

Un modelo de verificación es una caracterización del conjunto de entrenamiento basada en el análisis de un conjunto de características. Una característica no es mas que un rasgo cuantificable o bien de un slot o del conjunto de datos. Los valores que tengan cada una de estas características se usarán para obtener evidencias de la validez o no de los datos.

Estas características pueden clasificarse como numéricos u ortogonales, y si son aplicables a slots o a conjuntos de datos. Una característica decimos que es numérica si permite representar a un conjunto de datos mediante un número. En [15, 16, 18, 26, 36, 38, 5] podemos encontrar características de este tipo, van desde aquellas que únicamente cuenta el número de atributos de una clase que cumplen una serie de patrones de comienzo y de fin, hasta las que tienen en cuenta el conjunto de caracteres de un atributo, pasando por los que miden la similaridad entre atributos.

Las características categóricas engloban tanto a patrones que describen la estructura de un registro, hasta restricciones matemáticas de los valores que puede tener un atributo o relaciones entre ellos. En [21, 5] se describen más detalladamente diversas características de este tipo.

2.4 Modelos de verificación

En la literatura encontramos distintos modelos de verificacón, quizás los más conocidos son los que indicamos a continuación.

Modelo de Lerman: la técnica propuesta por Lerman [18] se puede usar si tenemos conjuntos de datos sin ningún dato válido y si las características que queremos analizar son numéricas. Este algoritmo se basa en caracterizar, tanto el conjunto de entrenamiento como los datos no verificados, como un vector cuya dimensionalidad se corresponde con el número de características. Cada elemento de dichos vectores contiene el valor medio que dicha característica tiene, respectivamente, en el conjunto de entrenamiento y en el conjuntos de datos. Para verificar el conjunto simplemente comprueba si dichos vectores son estadísticamente iguales para lo cual usa el estadístico de Pearson.

Modelo de Kusmerick: por otro lado Kusmerick presenta dos modelo que son práctica idénticos [15, 16]. Al igual que Lerman el conjunto de entrenamiento del que parte no tiene valores inválidos y, en este caso, permite trabajar con cualquier tipo de característica. Este

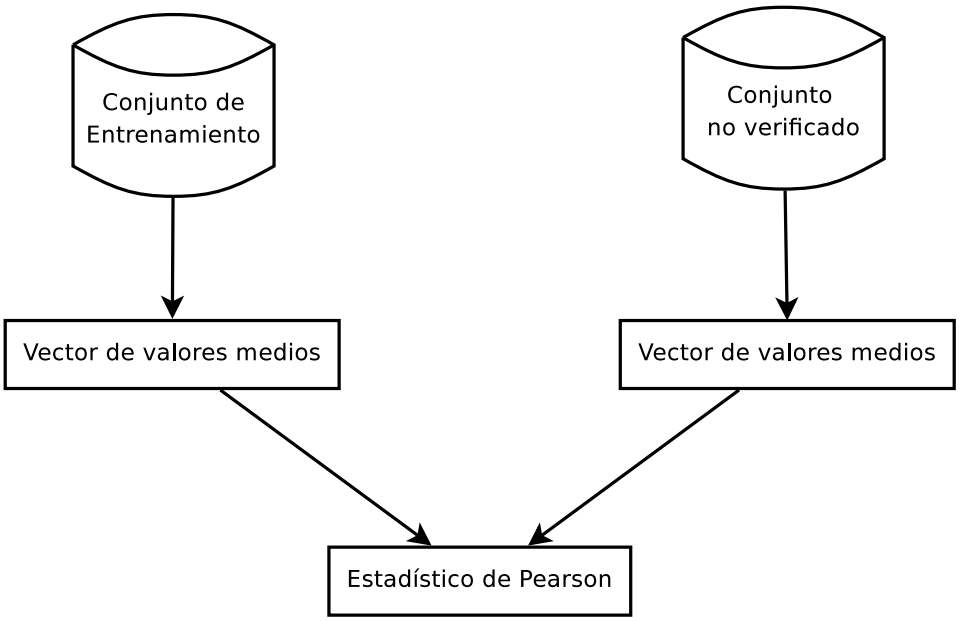


Fig. 2. Obtención de conjuntos homogéneos

método intenta modelar cada una de características como si fueran variables aleatorias que siguen una distribución de probabilidad. Para las variables numéricas usa una distribución normal mientras que para las categóricas usa una empírica que la crea basándose en los datos que estas variables toman en el conjunto de entrenamiento. Cuando queramos validar un conjunto lo único que hacemos es calcular la probabilidad de que los valores de las características de dicho conjunto sigan las distribuciones de probabilidad del conjunto de entrenamiento. Para combinar cada una de las probabilidades en un único valor Kusmerick propone tres alternativas:

- **Independencia:** supone que las variables son independiente y calculamos la probabilidad combinada como la multiplicación de todas ellas.
- **Vinculación:** supone que hay una variable que vincula a otras y, por tanto, la probabilidad combinada es igual a su valor mínimo.
- **Equivalencia:** todas las variables tienen la misma importancia así que la probabilidad combinada es igual a la media geométrica.

Modelo de MacCann: otra técnica muy conocida es la usada por MacCann [21], al igual que Kushmerick modeliza las características como distribuciones gausianas. El proceso de verificación propuesto se basa en cuatro características fundamentales:

1. Las estimaciones de las características se calculan mediante densidades normalizas, calcula la probabilidad de que una característica tenga un valor con más probabilidad.
2. Usa tanto conjuntos de valores válidos como inválidos. Cada característica se perfila mediante dos distribuciones normalizadas, una para los conjuntos de valores válidos y la otra para los inválidos. Aunque los autores proponen dos formas de combinar dichas distribuciones ninguna de ellas parece satisfactoria.
3. Calcula la probabilidad combinada de las distintas características teniendo en cuenta el peso discriminatorio de cada una de ellas. Para los conjuntos de datos válidos la probabilidad combinada se calcula como $p^+ = \sum_{p \in ps} wm(p)p(rs)$ y para los inválidos

como $p^- = \sum_{p \in ps} wm(p)(1 - p(rs))$ donde wm es una función que mapea pesos con estimaciones, rs es un conjunto de datos y ps es el conjunto de características.
4. El usuario define un límite fijo que se usará para rechazar o no el conjunto a verificar.

2.5 Filtrado de los datos

En ciertas ocasiones, al comprobar la validez de un conjunto de datos, el verificador devuelve un falso negativo debido a que dicho conjunto presenta alguna característica no deseadas, como por ejemplo la presencia de outliers. Los filtros son unos elementos que, dentro del marco de desarrollo de un verificador, se encargan de sanear el conjunto de datos explorando sus valores en busca de elementos que puedan provocar una errónea clasificación del mismo.

En [21] se proponen dos tipos de filtros, el primero se basa en el uso de una serie de fuente de datos externas semánticamente equivalentes. Dichas fuentes nos servirán para crear un nuevo modelo que se usará para comprobar la validez de aquellos atributos conflictivos del conjunto de datos a validar. El segundo filtro se basa en hacer un aprendizaje desde la web. En este caso parte tanto de los valores de los atributos como los patrones léxicos de cada uno de los atributos de la clase. Si se quiere comprobar la validez de ciertos atributos, se usa cualquier motor de búsqueda para localizar tanto los patrones instanciados como los atributos. Si en dicha búsqueda el número de páginas que contienen el patrón instanciado es superior al número de veces que aparece simplemente el atributo entonces dicho atributo se considerará válido.

3 Carencias y nuevas ideas

En el punto anterior describimos muy someramente los distintos pasos y algoritmos que se usan en el proceso de verificación de un wrapper. En este sección pretendemos enumerar algunas de las carencias que hemos detectado en varios de estas propuestas e intentaremos dar algunas nuevas ideas que intenten solvendar dichas desventajas.

3.1 Conjuntos de entrenamiento heterogéneos

Las técnicas de modelado de verificadores descritas en [18, 21] presuponen que los conjuntos de datos devueltos por el proceso de recolección son homogéneos. Dicho de otro modo, suponen que no existe dependencia entre la consulta que generó el conjunto de datos y los valores de las características. Por lo general, esto no es así y, por tanto, se debería asegurar lo homogeneidad de los mismos.

Para trabajar realmente con conjuntos de datos homogéneos, proponemos analizar los datos que forman el conjunto de entrenamiento y obtener una serie de nuevos conjuntos de datos que, ahora sí, serán homogéneos. Todo esto permitiría tener más de un conjunto de entrenamiento y, a cada uno de los cuales, se le asignaría un modelo de verificación. Cuando tengamos que verificar un conjunto de datos lo primero que tendremos que hacer es buscar aquel conjunto de entrenamiento al que más se asemeje y aplicar el modelo de verificación oportuno (figura 3).

La idea que hay detrás de todo este proceso es buscar clusters entre los conjuntos de datos, para lo cual podemos usa cualquiera de las técnicas descritas en [37, 24], luego describir cada una de estas agrupaciones mediante un prototipo, usando el algoritmo de la k-medias con la mejora propuesta en [20], y calcular lo cerca que está el conjunto sin verificar de cada una de estas agrupaciones. En principio puede parecer que esta técnica sólo es válida cuando trabajamos con características numéricas pero en [28, 30] se presentan una serie de resultados que permiten trabajar también con rasgos categóricos. En estos casos los cluster queda definidos por un centro y un radio.

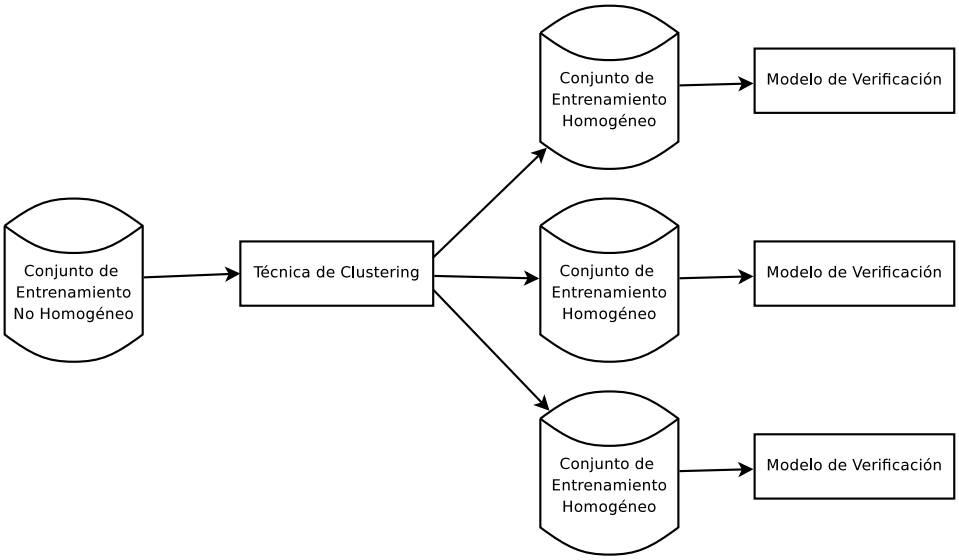


Fig. 3. Obtención de conjuntos homogéneos.

3.2 Conjuntos de datos inválidos

Como ya hemos visto en secciones anteriores, para obtener los modelos de verificación tenemos que partir de un conjunto de entremiento. Para que el aprendizaje sea adecuado es aconsejable que dicho conjunto contenga tanto ejemplos válidos como inválidos. Para nosotros esto es un problema ya que el conjunto de entrenamiento sólo dispone de ejemplos válidos. Aunque existen distintas técnicas que me permiten generar datos inválidos, no queda claro si estos se corresponden con la realidad o son demasiado artificiales. Una primera propuesta bastante sencilla consistiría en esperar a detectar el primer conjunto de datos inválido y hacer que formara parte de nuestro conjunto de entrenamiento. Según [4, 21] esta propuesta presenta dos problemas: el primer conjunto inválido tardará en detectarse y, cuando este se detecte, el wrapper se verá modificado.

Una forma de crear buenos ejemplo inválidos es usando clasificadores tal y como se indica en [5, 36]. El proceso sería el siguiente: primero construimos modelos de clasificación para los atributos de una clase; luego aplicamos dicho modelo a una serie de datos extraidos de la web; si el modelo clasifica correctamente dichos datos, pero estos no se corresponden con el atributo que estamos considerando, entonces podemos usar este conjunto de datos para crear ejemplos inválidos.

Si tenemos en nuestro conjunto de entrenamiento tanto datos válidos como inválidos aparece un nuevo problena: cómo combinar la información que nos da cada uno de los dos conjuntos. Si el número de conjuntos de datos inválidos es bastante grande podemos usar técnicas basadas en árboles de decisión, sietemas basados en reglas, máquinas de vector soporte, redes neuronales [35], algoritmo de los k-vecinos más cercanos [33], árboles indexados que mejoran considerablemente la eficiencia el método de los k-vecinos más o el algoritmo de las k-medias [22]. Otra técnica muy simple pero con unos resultados excelentes consiste en proyectar aleatoriamente los valores de las características en un susbespacio con un número de dimensiones preestablecido. Resultados teóricos demuestran que estas proyecciones preservan las distancias entre las relaciones. Una vez hechas las proyecciones podemos usar árboles indexados para aplicar la técnica del vecino cercano [10].

3.3 Independencia entre rasgos

Otro de los problemas que tiene el proceso de verificación es encontrar las características que mejor caractericen a un conjunto de datos. Cuantos más usemos la dimensionalidad del problema será mayor y, por tanto, los costes computaciones que conlleva serán más elevados.

Antes de crear el modelo de verificación, sería interesante estudiar el conjunto de características candidatas en busca de disminuir su tamaño. En una primera fase podrías obtener un conjunto de reglas de asociación [29] que nos permita detectar dependencias entre variables. Una vez que tenemos características independientes, podemos reducir aún más el conjunto si aplicamos técnicas de selección de características. En general este proceso se divide en cuatro etapas esenciales [6]:

- **Procedimiento de selección:** donde se selecciona el posible subconjunto de características.
- **Función de evaluación:** donde se evalúa el subconjunto de características.
- **Criterio de detención:** donde se comprueba si el subconjunto cumple diversas características deseables. Como criterio general se toma la tasa de error generada por el clasificador (C4.5, Naive Bayesian, Máquinas de Vector Soporte, Redes de propagación) que se genere con este subconjunto de características.
- **Procedimiento de validación:** en esta última fase validamos la calidad del subconjunto de características. La podemos medir calculando la correlación entre las variables o usando métodos de validacón cruzada.

Básicamente existen tres formas de ir generando los subconjuntos de características [6]:

1. **Métodos Completos:** estos métodos examinan todas las posibles combinaciones de características.
2. **Métodos Heurísticos:** utilizan una metodología de búsqueda de forma tal que no es necesario evaluar todos los subconjuntos de características. Por lo general lo que se hace es ordenar las características según su poder de clasificación [21, 35].
3. **Métodos Aleatorios:** son aquellos métodos que no tienen una forma específica de definir el subconjunto de características a analizar, sino que utilizan metodologías aleatorias.

Desde el punto de vista de la función de evaluación, los procedimiento de selección de características se pueden clasificar en dos categorías [12]:

1. **Métodos de filtraje:** el procedimiento de selección es realizado de forma independiente a la función de evaluación.
2. **Métodos dependientes:** el algoritmo de selección utiliza como medida la tasa de error del clasificador. Se obtienen generalmente mejores resultados que en el caso anterior, pero trae consigo un costo computacional mucho mayor.

3.4 Estimando variables

Kusmerick usa una distribución de probabilidad normal para caracterizar las características numéricas y una distribución empírica para las categóricas. Cuando quiere combinar la información de las distintas características lo que hace es presuponer que las variables aleatorias son independientes y multiplica sus valores. Ahora bien, si uno de los elementos del producto es muy pequeño, representaría un valor poco frecuente, entonces la probabilidad combinada sería muy pequeña Para evitar un problema similar con características categóricos se propone usar el estimador de Laplace. Presupone que los rasgos categóricos toman valores en

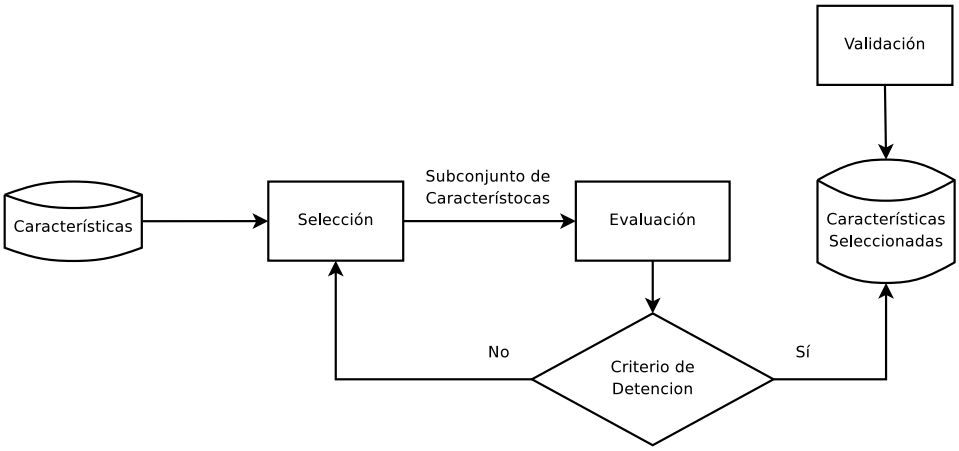


Fig. 4. Selección de características.

el conjunto $\{p_1, p_2, ..., p_n\}$ y las frecuencias de cada uno de estos en el conjunto de entrenamiento viene dada por $\{p_1 \longmapsto f_1, p_2 \longmapsto f_2, ..., , p_n \longmapsto f_n\}$. La frecuencia de cada patrón la calcula como $\frac{f_i}{\sum_{i=1}^{n} f_i}$.

Por otro lado MaCann usa densidades normalizadas para modelar los conjuntos de características (no usa características de slots). Las bondades de este tipo de densidades no se han demostrado aún en la práctica.

En [27] se presenta una técnica que se comporta muy bien cuando el número de casos inválidos de los que disponemos es pequeño. En esta técnica un valor de una característica que se encuentre en la cola de la distribuciíon se considera inválido. Presenta una fórmula que es capaz de calcular la probabilidad de que un valor esté en los extremos de la distribución.

Por lo general, no podemos suponer que las características siguen una distribución normal y, por tanto, tenemos que buscar técnicas que permitan modelarlas sin hacer suposiciones sobre su distribucíon. En [7] se propone el uso de la inecuación de Chebyshev de forma que el valor de una característica se considerara 1.0 si no se aleja de la media más de tres veces la desviación y 0 en otro caso (obviamente se puede asociar valores intermedios dependiendo de la distancia de separación respecto a la media). En [3] se propone el uso de una técnica denominada estimación de densidad del núcleo para calcular la función de densidad de probabilidad de una variable aleatoria, en nuestro caso una característica.

3.5 Combinando modelos

El proceso de verificación puede ser mejorado si, como se propone en [9], usamos un conjunto de modelos de verificación independientes, hasta ahora se ha considerado un único modelo. Si logramos obtener más de un modelo nos surge un nuevo problema, cómo combinar las decisiones tomadas por cada uno de ellos. Algunas de las técnicas que dan solución a este problema son: *bagging*, *boosting*, *staking*.

El bagging es una técnica muy simple pero requiere que los modelos sean homogéneos. Cuando queremos validar un conjunto lo pasamos como entrada de cada uno de los modelos que votará indicando si el conjunto es o no válido. El conjunto será válido si así lo dictamina la mayoría. Los distintos conjuntos de entrenamiento que se necesitan para obtener los modelos de verificación se logran tomando muestras del conjunto de entrenamiento inicial.

El principal problema que presenta la técnica de bagging es que todos los modelos son ponderados de la misma manera y, por tanto, se les da la misma importancia. Esto es un

error ya que al haberse construido los modelo de forma independiente lo más probable es que no sean complementarios y, por tanto, alguno cubran más espacio de búsqueda que otros. El boosting es similar al bagging pero en este caso la creación de los modelos es iterativo de tal forma que se van creando modelos que cubran aquellas zonas no cubiertas por el modelo anterior.

Staking es un método que no requiere que los modelos sean homogéneos. Esta técnica reemplaza el mecanismo de votación por la creación de un clasificador a partir de las salidas de los distintos modelos.

3.6 Técnicas no exploradas

Si reformuláramos el problema de la verificación de wrappers como la estimación de una serie de vectores y el cálculo de distancias dentro de dicho espacio vectorial, entonces la verificación se asemeja mucho a un problema ampliamente conocido en el campo del reconocimiento de outliers: el reconocimiento de novedades. Por tanto parece, factible utilizar algunas de las técnicas de dicho campo al ámbito de la verificación.

En [14] se propone un método eficiente para la detección de outliers que es fácilmente adaptable a nuestro problema. En este algoritmo si m de los k vecinos más cercanos de un conjunto no verificado están dentro de una distancia umbral, consideramos que dicho conjunto es válido. En [2] se sigue la misma metodología pero sólo se tiene en cuenta el m-ésimo vecino más cercano para así disminuir el costo computacional. El hecho de que el usuario tenga que definir una distancia umbral hace desaconsejable el uso de cualquiera de estas dos técnica. En [19] se presenta un modelo alternativo que se comporta bien en espacios multidimensionales grandes y, además, no hace ningún tipo de suposición acerca de la distribución que siguen las características.

En el campo de la detección de outliers es frecuente el uso de la máquina de vector soporte ya que han demostrado su buen funcionamiento cuando el número de conjuntos de datos inválidos en muy pequeño [32]. La idea es que se proyectan los vectores que modelan los conjuntos de datos en un espacio n-dimensional usando las denominadas funciones núcleo. El objetivo es definir un hiperplano que separe los datos válidos y los inválidos. Aunque en [8] se proponen distintas heuríscticas para facilitar la obtención de las funciones núcleo, el uso de máquinas vector soporte es computacionalmente complejo y encontrar las funciones núcleo adecuadas no es, en absoluto, trivial.

Otras técnicas relacionadas con la detección de outliers son las basadas en redes neuronales. En [1, 31] se usan un perceptrón multicapa, una de las cuales oculta. La red es entrenada mediante un conjunto de entrenamiento que cubre todos los posible valores válidos para el conjunto de datos. Si introducimos el conjunto aún no verificado al perceptrón y este no lo reconoce entonces se disparará una señal de alarma. En [11] se propone el uso de redes neuronales no asociativas que no son mas que redes basadas en perceptrones que intentan reducir la dimensionalidad del conjunto de caraterísticas. La idea es que se entrene a la red de tal forma que sólo reproduce los valores de entrada si estos son válidos. Todas estas propuestas sólo nos permiten trabajar con rasgos numéricos lo que parece ser una desventaja a no ser que consideremos vectores de estimaciones en vez de los vectores de características.

4 Conclusiones y trabajos

El proceso de verificación de un wrapper es largo y complejo, formado por un conjuro de etapas que realizan una tareas muy bien definidas. La dificultad de cada una de ellas a llevar a los autores a proponer diversas soluciones que alcancen los objetivos deseados. En general, ninguna de las propuestas soluciona los problemas completamente ya que suelen ser bastante sencillas y, a nuestro juicio, carecen de una visión global del problema. El aparente

aislamiento entre técnicas se pone de manifiesto por el simple hecho de no encontrar ningún trabajo que las compare en igualdad de condiciones.

Durante este artículo se han ido presentando las diversas técnicas que actualmente existen y se ha hecho especial hincapié en las carencias que estas presentaban. En algunas ocasiones son los propios autores los que las ponen de relieve comentándola en sus propios artículos ya sea de forma expresa o al no justificar las decisiones de diseño tomadas. No sólo hemos intentado localizar los posibles problemas sino que hemos propuesto posibles soluciones que deberán de ser evaluadas en trabajos futuros.

Referencias

1. C Bishop. Novelty detection and neural network validation. In *IEE Conf. on Vision and Image Signal Processing*, volume 4, pages 217–222, 1994.
2. Simon Byers and Adrian E. Raftery. Nearest neighbor clutter removal for estimating features in spatial point processes. *Journal of the American Statistical Association*, 93:577–584, 1998.
3. S. Chen, X. Hong, and C. J. Harris. An orthogonal forward regression technique for sparse kernel density estimation. *Neurocomput.*, 71(4-6):931–943, 2008.
4. Boris Chidlovskii. Automatic repairing of web wrappers by combining redundant views. In *ICTAI '02: Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'02)*, page 399, Washington, DC, USA, 2002. IEEE Computer Society.
5. Boris Chidlovskii, Bruno Roustant, and Marc Brette. Documentum eci self-repairing wrappers: performance analysis. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 708–717, New York, NY, USA, 2006. ACM.
6. M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1:131–156, 1997.
7. O. de Oliveira and A. da Silva. Automatic verification of data extracted from the web. In *Proceedings of the Brazilian Symposium on Databases*, page 5671, 2003. Journal version in AIJ, available at http://citeseer.nj.nec.com/13663.html.
8. Dennis DeCoste and Marie Levine. Automated event detection in space instruments: A case study using ipex-2 data and support vector machines. In *SPIE Conference Astronomical Telescopes and Instrumentation*, 2000. Journal version in AIJ, available at http://citeseer.nj.nec.com/13663.html.
9. Thomas G. Dietterich. Ensemble methods in machine learning. pages 1–15. Springer-Verlag, 2000.
10. Dmitriy Fradkin and David Madigan. Experiments with random projections for machine learning. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 517–522, New York, NY, USA, 2003. ACM.
11. Nathalie Japkowicz, Catherine Myers, and Mark Gluck. A novelty detection approach to classification. In *In Proceedings of the Fourteenth Joint Conference on Artificial Intelligence*, pages 518–523, 1995.
12. George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *International Conference on Machine Learning*, pages 121–129, 1994. Journal version in AIJ, available at http://citeseer.nj.nec.com/13663.html.
13. Mohammed Kayed and Khaled F. Shaalan. A survey of web information extraction systems. *IEEE Trans. on Knowl. and Data Eng.*, 18(10):1411–1428, 2006. Member-Chia-Hui Chang and Member-Moheb Ramzy Girgis.
14. Edwin M. Knorr and Raymond T. Ng. Algorithms for mining distance-based outliers in large datasets. pages 392–403, 1998.
15. Nicholas Kushmerick. Regression testing for wrapper maintenance. In *AAAI '99/IAAI '99: Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, pages 74–79, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence.
16. Nicholas Kushmerick. Wrapper verification. *World Wide Web*, 3(2):79–94, 2000.
17. Juliano Palmieri Lage, Altigran S. da Silva, Paulo B. Golgher, and Alberto H. F. Laender. Automatic generation of agents for collecting hidden web pages for data extraction. *Data Knowl. Eng.*, 49(2):177–196, 2004.

18. Kristina Lerman, Steven N. Minton, and Craig A. Knoblock. Wrapper maintenance: A machine learning approach. *Journal of Artificial Intelligence Research*, 18:2003, 2003.
19. Manuel Castejón Limas, Joaquín B. Ordieres Meré, Francisco J. Martínez De Pisón Ascacibar, and Eliseo P. Vergara González. Outlier detection and data cleaning in multivariate non-normal samples: The paella algorithm. *Data Min. Knowl. Discov.*, 9(2):171–187, 2004.
20. S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.
21. Robert McCann, Bedoor AlShebli, Quoc Le, Hoa Nguyen, Long Vu, and AnHai Doan. Mapping maintenance for data integration systems. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1018–1029. VLDB Endowment, 2005.
22. Andrew W. Moore. The anchors hierarchy: Using the triangle inequality to survive high dimensional data. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 397–405, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.
23. Alberto Pan, Juan Raposo, Manuel Álvarez, Víctor Carneiro, and Fernando Bellas. Automatically maintaining navigation sequences for querying semi-structured web sources. *Data Knowl. Eng.*, 63(3):795–810, 2007.
24. Dan Pelleg and Andrew W. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.
25. Sriram Raghavan and Hector Garcia-Molina. Crawling the hidden web. In *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, pages 129–138, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.
26. Orna Raz. Semantic anomaly detection in online data sources. In *In ICSE*, pages 302–312, 2002.
27. Stephen J. Roberts. Novelty detection using extreme value statistics. In *IEE Proceedings on Vision, Image and Signal Processing*, pages 124–129, 1999.
28. Dan Simovici, Namita Singla, and Michael Kuperberg. Metric incremental clustering of nominal data. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 523–526, Washington, DC, USA, 2004. IEEE Computer Society.
29. Ming syan Chen, Jiawei Hun, Philip S. Yu, Ibm T. J, and Watson Res Ctr. Data mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8:866–883, 1996.
30. Pablo Tamayo, Charles Berger, Marcos M. Campos, Joseph Yarmus, Boriana L. Milenova, Ari Mozes, Margaret Taft, Mark F. Hornick, Ramkumar Krishnan, Shiby Thomas, Mark Kelly, Denis Mukhin, Robert Haberstroh, Susie Stephens, and Jacek Myczkowsji. Oracle data mining - data mining in the database environment. In Oded Maimon and Lior Rokach, editors, *The Data Mining and Knowledge Discovery Handbook*, pages 1315–1329. Springer, 2005.
31. L. Tarassenko, A. Nairac, N. Townsend, I. Buxton, and P. Cowley. Novelty detection for the identification of abnormalities. *Int. J. System Science*, 31(11):1427–1439, 2000.
32. David M. J. Tax, Er Ypma, and Robert P. W. Duin. Support vector data description applied to machine vibration analysis, 1999.
33. Dietrich Wettschereck. *A study of distance-based machine learning algorithms.* PhD thesis, Corvallis, OR, USA, 1994. Adviser-Thomas G. Dietterich.
34. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, October 1999.
35. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann Series in Data Management Sys. Morgan Kaufmann, second edition, June 2005.
36. Tak-Lam Wong and Wai Lam. Adapting web information extraction knowledge via mining site-invariant and site-dependent features. *ACM Trans. Interet Technol.*, 7(1):6, 2007.
37. Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: A new data clustering algorithm and its applications. *Data Min. Knowl. Discov.*, 1(2):141–182, 1997.
38. Zhen Zhang, Bin He, and Kevin Chen-Chuan Chang. Understanding web query interfaces: best-effort parsing with hidden syntax. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 107–118, New York, NY, USA, 2004. ACM.