Economic Insights – Trends and Challenges	Vol.V(LXVIII) No. 3/2016	39 - 46

Knowledge Acquisition by Document Annotation

Aurelia Pătraşcu

Faculty of Economic Sciences, Petroleum-Gas University of Ploieşti, Bd. Bucureşti 39, 100680, Ploieşti, Romania
e-mail: patrascuaura@yahoo.com

Abstract

Association model of document viewing with ontology viewing is the most important component of the semantic documents approach. Storing, in the same place, the ontologies and documents allows an efficient grouping in packages and their manipulation.
Thus, the purpose of this paper is to give an overview on the exponential development of information technologies based on two findings: on one hand, using information technology as various applications, as a means of improving the decisional process and raise the data processing speed and on the other hand, the risks associated with the integration of information technologies in every company which can lead to substantial losses if they are ignored.

Keywords*: document annotation; semantic documents; information technology*

JEL Classification: *C81; C61*

Introduction

In recent years, so many technologies have developed that enable the document management that the main issue that deserves attention is their integration.

Document management system means a computer system which is increasingly exploited by businesses in various sectors of socio-economic activity, its primary characteristic being automation of the computerized and human activities, especially those that involve interaction with applications and information technology tools.

In general, three large components can be retrieved from any document management system existing on the market: document repository, workflow engine and indexing technology-search.

In order to extract concepts and relationships in the text, the researchers tested various methods, from the automatic analysis of natural language, to the manual document annotation.

One of the semantic annotation tools described in literature (Vargas-Vera et al., 2002) provides the means for building ontologies from web pages. This system allows designers to achieve knowledge acquisition in 4 steps: browse the web, semantic annotation, learning rules and extracting information. Thus, this approach represents a systematic gradual extraction method of ontology knowledge from text. This tool was developed in the annotation tool *MnM* (Vargas-Vera et al, 2002)**.**

Smore is an editor that allows users to add semantic markup while creating web pages (Kalyanpur et all, 2006).

Another annotation tool is *CREAM* (Handschuh et al, 2002) that provides support for simultaneous achieving of metadata creation activity and web page content creation activity, as well as for preexisting web pages' annotation. The framework *S-Cream* along with its reference implementation allows semi-automatic annotation of web pages.

Onto-h (Benjamins et al., 2005) is a collaborative and semi-automatic tool for document annotation and for binding annotations to ontologies. Annotation tool assist designers, providing them with a smart editor that is implemented as a plugin in Protégé.

Asbru and guideline markup tool (GMT) are systems for annotation of clinical guidelines in the form of text and gradual conversion into an executable representation (Kosara et al, 2002, Votruba et al, 2004).

It is noted that the objective of semantic document does not consist in the acquisition of knowledge as such. Rather, these methods complement and help build semantic document. A possible scenario is to use a method of knowledge acquisition from text to build ontologies that are a component of semantic documents. For example, a developer of medical guidelines might use Asbru to analyse a document constituting a pre-existing medical guide and build the knowledge base that becomes part of the semantic document which represents a medical guide.

Annotations and Metadata in PDF

Currently, the most widely used format for implementation of semantic documents is PDF format (Adobe Acrobat Reader DC, 2015). PDF documents are available throughout the world, being used for the document storing and printing.

Many organizations keep their online documents archives in PDF format, thus allowing access to them within the internal networks and the Internet.

The most important advantages of PDF documents are the possibility to document and the opportunity to improve the format with additional information (Eriksson, 2007). In addition, there are a few commercial tools and open-source software that create, analyse and modify PDF files. They can be used, naturally, as a basis for semantic documents and other document formats, such as Word or RTF. It is possible to convert most formats of documents in PDF format.

Adobe PDF format is an open format, as it is published and it can be developed customized applications that create, modify and read this format. Indeed, there are few PDF commercial and open-source applications. Initially, Adobe created the PDF format in the early 1990s as the format for Acrobat product, in order to facilitate the exchange of documents and remove the compatibility problems generated by PostScript documents. These documents cannot be printed in similar conditions on different printers and font configurations. Like PostScript files, PDF documents use text and graphs descriptions that are oriented on the page and provide support for device or resolution independent interpretation. One of the PDF goals was getting an archiving format that allows printing of documents properly over a long period of time. Another objective was to provide a support for viewing on screen, interactive of documents. PDF uses a special pattern file in order to meet these requirements (Figure 1 shows the storage format).

PDF files are made up of a collection of numbered objects containing the information needed to interpret the document's pages.

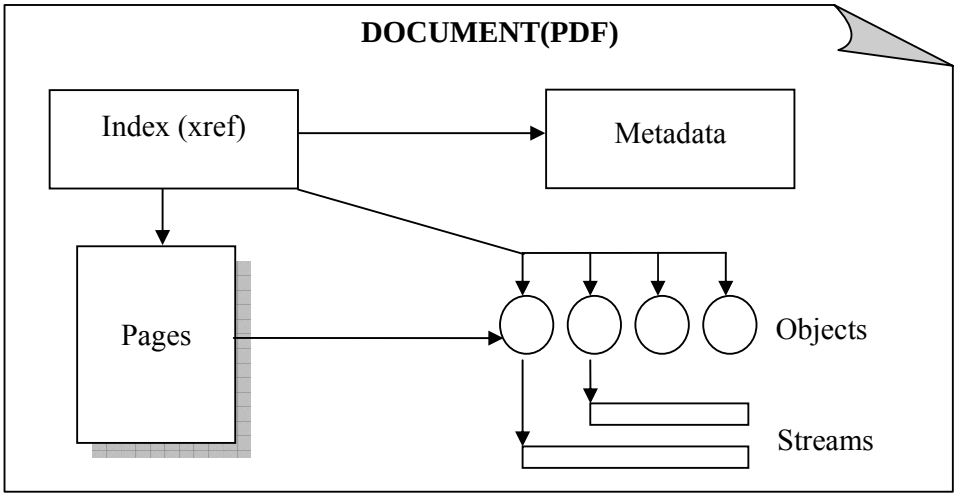

Fig. 1 The internal components of the PDF document.

Source: Eriksson, 2007, 627.

These objects provide support both for textual data and binary data, such as: the images encoded in different formats. A table of cross-references (cross reference) provides an index of object positions in the file. This approach allows random access to file that is important to achieve a viewing on the screen and a quick scroll of the file. Thus, it can be concluded that this format represents a file system within another file. In addition, PDF ensures objects compression and encryption. Because PDF provides this functionality at the object level, it is possible to control the compression and encryption of certain parts belonging to the document. For example, a publisher may restrict, through encryption, the reading and printing of certain pages from a document.

The evolution degree of PDF is moderated. In the last decade, Adobe has continued to add new features to the PDF specification. While the original PDF version focus on the transfer of documents, recent versions support web links, bookmarks, annotations, notes, forms, fields, metadata, transparency etc. Thus, the main functionality remains, largely, unchanged and current versions offer improved support in the areas of interactive visualization, collaborative work and online connectivity.

XMP Protocol is a method to add metadata to documents and other files, such as Adobe FrameMaker and AdobePhotoshop files. The purpose of this protocol is to allow flexible searching and retrieving and to facilitate automation based on workflow metadata. XMP stores metadata such as title, subject, author and data in the form of RDF triples. For example, Adobe Acrobat keeps in RDF a subset of elements from the Dublin Core metadata which is stored within PDF files.

XMP (XMPS, 2016) is shown both for rapid scanning of multiple files, as well as for the detailed analysis of the files. XMP metadata are relatively easy to accomplish because RDF statements are "wrapped" into a syntax that starts with a sequence of distinct markups. Computer applications can scan these tag sequences and, then, analyse the RDF-based metadata included in those tags. Another advantage of XMP Protocol is that applications that do not recognize this protocol usually ignore metadata.

This protocol is definitely, an important step in ensuring universal support for the semantic web and its adaptation. However, PDF files with XMP protocol there are not semantic documents, because it does not provide sufficient support for RDF and cannot connect the document annotation to knowledge representation. XMP restricts the meaning of RDF instructions,

allowing only to the document itself to be a subject of RDF triples (for example, use the attribute "rdf: about" to the "rdf: Description" elements to refer to a document unique identifier). This restriction means that it is not possible to write RDF/RDFS instructions in XMP. Accordingly, it is not possible to use OWL language in the header of the RDF statement in XMP, since OWL use rdf: about and rdf: ID attributes to identify the classes. In other words, XMP is inappropriate for RDF/RDFS and OWL ontologies.

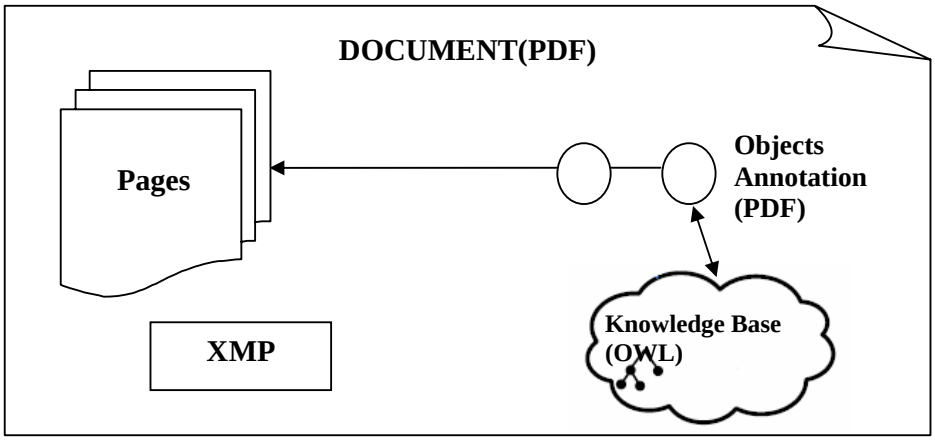

Fig. 2. Knowledge Base and metadata XMP of PDF document.

Source: Eriksson, 2007, 628.

The XMP aim is to represent the metadata from the document in an extensible manner, which is totally different from the semantic documents objective that consists in combining documents with knowledge representation. The following shall be considered as basic example that XMP is unsatisfactory. In XMP, documents may have a string value for the author property. You cannot insert objects independent and cannot define the relationships between them, because it is not possible to define properties than for this document. For example, you cannot define relationships between multiples authors of a document, such as a student – leader relationship. As a language for adding metadata to document, XMP presents advantages. However, in the current version of the Acrobat implementation, XMP language cannot be used for semantic documents. Although, initially, it was thought that the XMP should be used as a basic element of knowledge representation in semantic documents, in the end, this idea was abandoned in favour a separate area for storage of RDF/RDFS and OWL expressions in PDF files. Currently, extensibility can be used to insert RDF/RDFS and OWL ontologies in PDF files in parallel with XMP by defining the objects and RDF/RDFS flows and OWL (fig. 2.). PDF structure allows both formats for metadata to coexist in the same file.

The key idea of the semantic document annotation is that the texts and graphs should be placed in conjunction with the ontology and vice versa. The advantage of this association is represented by the fact that users can move easily between ontology and document views and that the search tools based on ontology can accurately locate text. As mentioned previously, PDF ensures annotation of documents. Through this feature you can highlight and comment words and regions from PDF pages and, also, you can save these comments in the PDF file. You can use this type of annotation to bind the text and graphics of ontologies.

Acrobat provides annotation tools to highlight text and add notes and stamps to preexisting PDF documents (MS Word has similar functions to its own formats). In addition, Acrobat enables document check and electronic signatures.

In Figure 3 it is presented the operation of PDF document annotation. Acrobat uses several types of annotations and it can be created a custom annotation type through an extended

mechanism of plug-ins. This plugin interface ensures implementation of semantic annotation tools that allow users to highlight text and regions, as well as create links to the ontology definitions.

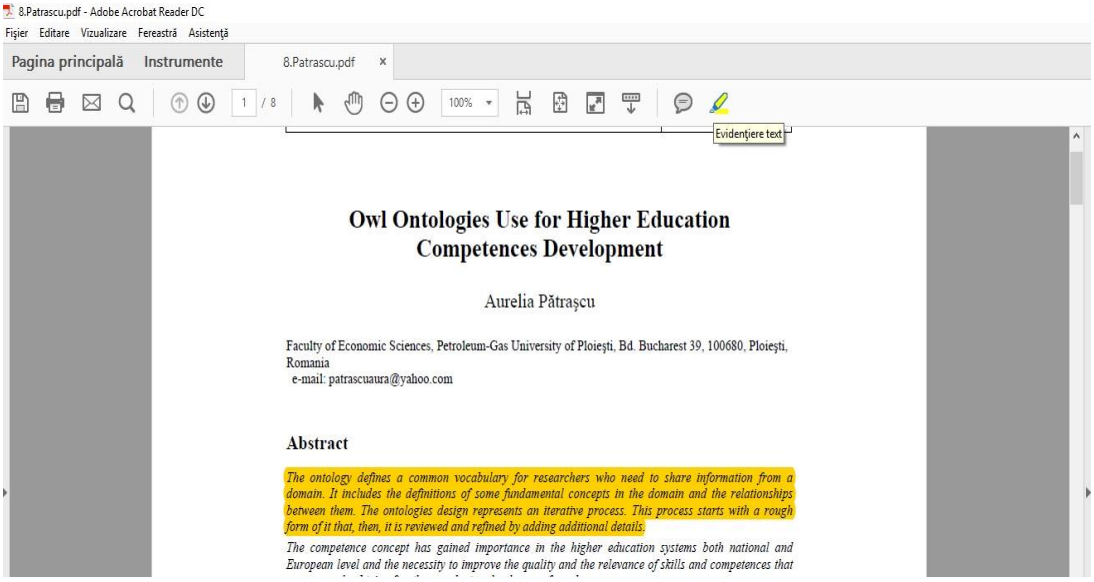

Fig. 3. PDF document annotation

Source: made by the author.

Ontologies and Documents Integration

The association model of document view with ontology view is the most important component of the semantic document approach. Storing in the same place the ontologies and documents allows an efficient grouping in packages and their manipulation. This type of storage is similar to, in essence, with the entire document annotation with ontology-based metadata and the DublinCore metadata set for a document. However, the strength of semantic documents is drawn from documents and ontologies integration through a common model. The goal of this integration model is to enable readers, authors, developers and programmers to alternate between documents and ontologies and use them in combination.

In order to integrate documents and ontologies, several methods can be used. Integration models should recognize all documents and ontologies features. This requirement means that is important to use the recognized formats for documents and that the integration model should not restrict the document format. For example, the usual tools for managing the format should work as before. Similarly, the integration model should enable the ontological language retains its expressiveness.

There are three models that can be used for integrating ontologies and documents:

1. Inserted active objects

In this method, the document authors include in the document an element of knowledge representation, similar to the inclusion of a table or figure. This item occupies an area of the page and is part of the text flow. KWrite tool (KDEA *) uses this strategy. In addition, this method is similar to the Java applet and ActiveX objects from HTML documents.

2. Document outline related to ontologies

According to this method, parts of the structural elements from the document, such as: title, summary, abstract, pages and figures refer to ontology elements. Although many word processors can manipulate these structures, information is not always kept in the final documents.

3. Document annotations connected to ontologies

In this case, the arbitrary annotation of text and document regions refers to ontology elements. It is similar to the use of structured links between documents for hypertext. Annotation tools of the semantic web, such as the Annotea [Anno**], use document annotations connected to ontologies. This approach is compatible with the second method (document outline related to ontologies), because you can annotate the document elements.

The ontologies that achieve mapping document annotations to domain ontologies can be grouped into three categories:

1. Ontology for annotating

Ontology in this category describes document annotations. It consists of classes necessary to instantiate the annotation individuals. Also, this ontology contains the annotation properties such as number of pages, selected text and position. For example, the PDFTab plugin uses annotation ontology with classes that correspond to annotations from the text, on the graphics or in a specific region. PDFtab tab is a plugin extension of Protégé ontology editor managing semantic documents. The purpose of the tab is to allow developers to import PDF documents and connect them to the ontologies using annotations. Another purpose of this tab consists of supporting the visualization and annotation activities for large documents. PDFTab plugin meets these goals by incorporating Acrobat product in the Protégé ontology editor, in a new manner. In other words, it connects the ontologies domains and document management and allows developers to use the Protégé environment to create the semantic documents ontologies.

2. Document Ontology

The purpose of this ontology is to shape the document structure independent of annotations. The document ontology depends on the document format and contains concepts for major parts of the document. For example, an ontology document for reports can contain classes for chapters, sections, paragraphs, figures and tables.

3. Domain Ontology

They contain the document content model. In fact, a domain ontology describes the terminology used in document text and shapes, at least, some of the fundamental semantics. For example, a semantic document representing an annotated history book can use such an ontology consisting of classes such as: people, events and locations. Depending on the application, it is possible that a domain ontology consists of, in fact, several ontologies with different purposes.

Through these ontologies can achieve a demarcation between the domain scope and implementation of the annotation-document link.

Figure 4 shows these ontologies, the links between them and exemplifies for an annotation the association between the instances of different ontologies.

In this example, PDFTab links the annotation in the text with the instance annotation (“Annotation #80A”) which in turn refers to the instance “Tabel #80”. This is an instance of the class “Tabel” from the document's ontology (Protégé generates an implicit identifier for instances). Then, the Tabel instance refers to the instances from the domain ontology (in this example, the operator “City population” and the class “Citizens”). Also, Figure 4 shows the demarcation between the ontology and the system.

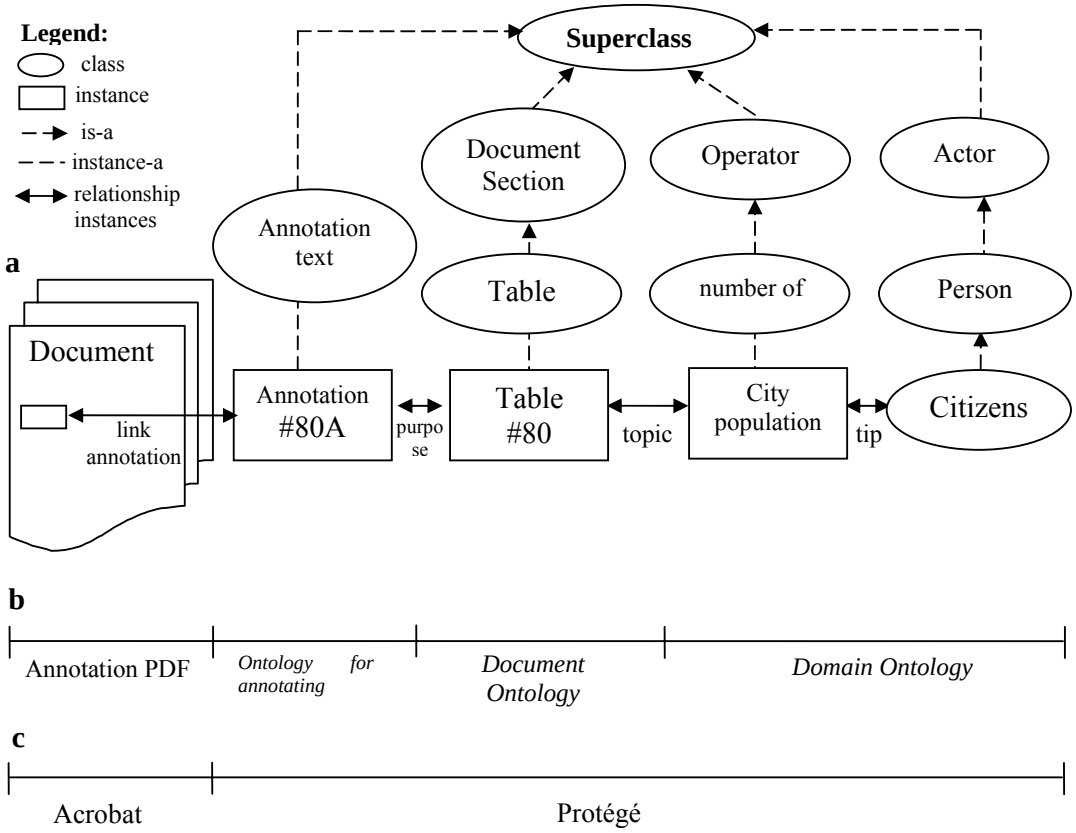

Fig. 4 Support ontologies for shaping the document and domain
a. An example of annotation with the corresponding links to instances
b. Length of annotation, document and domain ontologies
c. Demarcation between Acrobat and Protégé.
Source: (Eriksson, 2007, 633)

The annotation and document ontologies are compatible with the semantic web, as the basic terminologies can be used on the web (e.g. AnnotationText and Table) and because for these types of ontologies spaces for names can be defined. It is possible to use different document ontologies and reuse document ontologies created by others (such as: document ontologies for books, reports, facts charts, design documents, articles and PhD theses), as different documents have different components. In addition, the domain ontology can consist of a combination of ontologies, for example, a high level ontology, a collection of recycled ontologies and an ontology specific to the application.

Conclusions

These integration strategies have both advantages and disadvantages. The advantage of active inserted objects is that ontology can be edited directly in the document. However, it is not clear how you can combine different active objects (that are restricted to specific areas) from a large document into a consistent ontology that describes the document. Inserted objects are unsuitable for documents written for other purposes and for pre-existing documents. The document

outlines connection to ontologies presents the advantage of a clear structure of the sections that can be mapped into an ontology in a direct way. However, there are many situations in which the outline mapping is too crude to be able to associate the document and ontologies. Document annotations linked to the ontology provides the best level of detail to link text and text areas with ontologies. Detailed connection is essential when the application requires the existence of explicit links between terms and phrases in your document, on the one hand, and concepts from ontology, on the other hand. A disadvantage of this method is that the process of annotation can be especially difficult, because there is no outline structure that can be observed.

It is, also, easy to imagine the combinations of these integration strategies. For example, one possibility consists in combining document outlines with arbitrary annotations. Finally, the integration model must be able to carry out the inverse references from ontologies to documents, such as links to annotations and document elements. Currently, they are used document annotations connected to the ontologies, as first integration strategy because it requires a concept integration between documents and ontologies.

References

1. Adobe Acrobat Reader DC, 15.017.20053, August, 2016 http://www.adobe.com/devnet-docs/acrobatetk/tools/ReleaseNotes/DC/dccontinuousaugust2016ooc.html
2. Benjamins, V.R., Contreras, J., Blazquez, M., Nino, M., Garcia, A., Navas, E., Rodriguez, J., Wert, C., Millan, R. and Dodero, J., 2005. ONTO-H: a collaborative semiautomatic annotation tool, *In Proceedings of the Eighth International Protégé Conference*, Madrid, Spain, July 18-21, http://protege.stanford.edu/conference/2005/submissions/abstracts/accepted-abstract-benjamins.pdf.
3. Eriksson, H., 2007. The semantic-document approach to combining documents and ontologies, *International Journal of Human-Computer Studies*, vol. 65, nr. 7, pg. 624-639, ISSN 1071-5819.
4. Extensible Metadata Platform (XMP), 2016. *Specification: Part 3*, Additing Intelligence to Media, Storage in Files.
5. http://wwwimages.adobe.com/content/dam/Adobe/en/devnet/xmp/pdfs/XMP%20SDK%20Release%20cc-2016-08/XMPSpecificationPart3.pdf
6. Kalyanpur, A., Hendler, J., Parsia, B. and Golbeck, J., 2006. *SMORE: semantic markup, ontology, and RDF, editor.* Technical Report, University of Maryland, http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA447989&Location=U2&doc=GetTRDoc.pdf
7. Kosara, R., Miksch, S., Seyfang, A. and Votruba, P., 2002. Tools for Acquiring Clinical Guidelines in Asbru", *In Proceedings of Sixth World Conference on Integrated Design and Process Technology (IDPT'02),* Pasadena (California), USA, June 23 - 28, 2002, pg. 22-27.
8. Handschuh, S. and Staab, S., 2002. Authoring and annotation of web pages in CREAM, in *Proceedings of the 11th International World Wide Web Conferenc*e, Honolulu, Hawaii, USA, 7-11 May, pg. 462 – 473, ISBN 1-58113-449-5, http://www2002.org/CDROM/refereed/506/index.html.
9. Vargas-Vera, M., Motta, E., Domingue, J., Lanzoni, M., Stutt A. and Ciravegna, F., 2002. *MnM:* "Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup", in *Proceedings The 13th International Conference on Knowledge Engineering and Management* (EKAW 2002), Siguenza, Spain, October 1-4, 2002, Lecture Notes in Computer Science, Springer, vol. 2473, pg. 379-391, ISBN 3-540-44268-5.