# Use of Linked Data principles for semantic management of scanned documents

# *Emprego dos princípios Linked Data para gestão semântica de documentos digitalizados*

Luciane Lena Pessanha MONTEIRO[1]
Mark Douglas de Azevedo JACYNTHO[2]

## Abstract

The study addresses the use of the Semantic Web and Linked Data principles proposed by the World Wide Web Consortium for the development of Web application for semantic management of scanned documents. The main goal is to record scanned documents describing them in a way the machine is able to understand and process them, filtering content and assisting us in searching for such documents when a decision-making process is in course. To this end, machine-understandable metadata, created through the use of reference Linked Data ontologies, are associated to documents, creating a knowledge base. To further enrich the process, (semi)automatic mashup of these metadata with data from the new Web of Linked Data is carried out, considerably increasing the scope of the knowledge base and enabling to extract new data related to the content of stored documents from the Web and combine them, without the user making any effort or perceiving the complexity of the whole process.

**Keywords**: Linked data. Digitalization. Digital documents. Metadata. RDF. Semantic Web.

## Resumo

*Este trabalho aborda o uso da Web Semântica e dos princípios Linked Data propostos pelo World Wide Web Consortium no desenvolvimento de uma aplicação Web de gestão semântica de documentos digitalizados. O objetivo principal é registrar documentos digitalizados, descrevendo-os de maneira que a máquina consiga compreendê-los e processá-los, realizando filtragem de conteúdo e nos auxiliando na busca por tais documentos quando de uma tomada de decisão. Para tal, metadados inteligíveis por máquina, criados por meio do emprego de ontologias Linked Data de referência, são associados aos documentos, criando uma base de conhecimento. Para enriquecer ainda mais, é realizado, de forma (semi)automática, o mashup destes metadados com dados provenientes da nova Web de Dados Ligados, aumentando, sobremaneira, a abrangência desta base de conhecimento e possibilitando extrair da Web novos dados relacionados ao conteúdo dos documentos armazenados e combiná-los, sem a necessidade de que o usuário da aplicação faça qualquer esforço ou perceba a complexidade de todo o processo.*

***Palavras-chave***: *Dados ligados. Digitalização. Documentos digitais. Metadados. RDF. Web semântica.*

## Introduction

Over the few years, there has been an increasing interest in the movement of the Semantic Web. This is an extension of the original Web proposed by Berners-Lee *et al.* (2001). The main idea is to use the Web not only to share information, but to share the meaning (semantics) of information. In other words, documents (html pages)

[1] Universidade Candido Mendes, Departamento de Computação, Coordenação de Computação. Campos dos Goytacazes, RJ, Brasil.

[2] Universidade Candido Mendes, Centro de Pesquisa Candido Mendes, Programa de Pós-Graduação em Pesquisa Operacional e Inteligência Computacional. R. Anita Peçanha, 100, Pq. São Caetano, 28030-335, Campos dos Goytacazes, RJ, Brasil. Correspondence to/*Correspondência para*: M.D.A. JACYNTHO. *E-mail*: <markjacyntho@gmail.com>.

Received in 5/22/2015 and approved in 9/1/2015.

http://dx.doi.org/10.1590/2318-08892016000200010

containing a collection of structured machine-comprehensible statements (metadata) that allow the machine to understand the content of the documents, making intelligent decisions to assist us.

As more pragmatic subset of the Semantic Web, there is the so-called Web of Linked Data. The Linked Data concept, proposed by Berners-Lee (2006), consists of publishing data directly, rather than just publishing metadata associated with documents, and interconnect them through semantic links (relationships), creating a global data space. A Web of structured data, which is fully understood by software agents, making the search for information more accurate and consistent (Heath & Bizer, 2011).

Thus, the purpose of this study is to build a Web application for semantic management of scanned documents in which the machine can not only store scanned documents, but also process/understand the meaning (semantics) of the content of documents, making "smart" decisions to help us by filtering content or even giving us the answer to a question. To this end, a very promising solution is to semantically annotate these documents. In other words, associating a collection of structured statements (metadata) with documents in a way the machine can understand the meaning of the documents from these statements. Statements are made by means of a standard data model and using constructs (concepts and properties) of a standard ontology, where ontology is nothing more than a formal representation model of a particular area of knowledge.

This paper is structured as follows: first, we present the framework of the Semantic Web; the following section describes the methodological path for the construction of the proposed document management application; then an example of use is shown; some related studies are listed in the next section; and, in the last section, we discuss the conclusions and suggest further studies.

*Semantic Web*: The Semantic Web is a project initiated in 2001 by the World Wide Web Consortium (W3C) <http://www.w3.org> that comprises a set of technologies and standards able to extend the current Web of Documents, turning it into the Web of Data. On the Web of Data, each entity or resource (representing, for example, a person, a place, a company, in short, any real-world entity) is identified by a Web address, a Uniform Resource Identifier (URI) that uniquely identifies it in the world. Dereferencing (accessing) a given resource URI, a file is provided containing a representation (description) of this resource in a standard structured language understandable by machine. In this new Web, resources have semantic relationships (connections) with other resources or literal values (such as text or dates) through Web links representing properties, the so-called typified links, which describe and relate representations of geographically distributed resources.

*Resource Description Framework* (RDF): is the standard data model of the Semantic Web. The RDF model is based on graphs and composed of statements (Cyganiak *et al.*, 2014).

A statement represents a triple *resource-property-value* (also known as *subject-predicate-object*). Resource, identified by a URI (Web address), is any real-world entity you want to describe, such as a person, a place or a theme, for instance. *Property* is a property of some ontology, also identified by a URI, which establishes a relationship between a resource and a value. Value may be represented by another resource or a literal value.

Taking this article as example and assuming it were to be identified by the URI <http://periodicos.puc-campinas.edu.br/seer/index.php/transinfo/article/view/0001>, we could set its title by associating the string that represents the title to the URI of the article, using the property <http://purl.org/dc/terms/title>, thus forming a triple, where the *resource* is the URI of the article, the *property* is <http://purl.org/dc/terms/title> and the *value* is the title content.

In addition, we could establish that this article has "Semantic Web" as its subject by associating the URI of the article with a third-party URI such as <http://dbpedia.org/resource/Semantic_Web>, using the property <http://purl.org/dc/terms/subject>, generating the triple: *resource*: <http://periodicos.puc-campinas.edu.br/seer/index.php/transinfo/article/view/0001>, *property*: <http://purl.org/dc/terms/subject>, *value*: <http://dbpedia.org/resource/Semantic_Web>.

Note that related resources do not need to reside in the same data source. For example, the RDF representation of the resource <http://periodicos.puc-campinas.edu.br/

seer/index.php/transinfo/article/view/0001> would be published on the server <http://periodicos.puc-campinas.edu.br> however the RDF representation for the resource <http://dbpedia.org/resource/Semantic_Web> would be on the server <http://dbpedia.org>. The relationship between resources described by different data servers (data reuse or Linked Data mashup) is the essence of the Web of Linked Data since the machine can navigate from one resource to another, acquiring more information, regardless of data location.

*Ontologies* (*semantics*): the statements (triples) contained in different websites cannot be arbitrary. They must be created by using common terms and relationships, *i.e.* a common vocabulary for a given knowledge domain, an ontology (Jacyntho, 2012). Heflin (2004) defines ontology as a common set of terms that are used to describe and represent a domain. An ontology defines terms used to describe and represent an area of knowledge.

An ontology is created for a specific domain, such as science, education, and people, for example. An ontology defines concepts or classes, establishes relationships between these classes and also defines properties, which describe various features and attributes of the classes. It can be said that an ontology encodes the knowledge of a given domain, so that this knowledge can be understood by machines, making the Semantic Web possible (Jacyntho, 2012). By means of knowledge described by an ontology, the machine can perform inferences, reason and deduce new triples based on the relationships between terms described in ontologies.

It is essential that there exist languages for building ontologies, the so-called meta-ontologies. Existing meta-ontologies are RDF-Schema (rdfs) (Brickley & Guha, 2014) and Web Ontology Language (OWL) (McGuiness & Harmelen, 2004).

*Sparql Protocol and RDF Query Language* (SPARQL): is the standard query language and access protocol for RDF data (Harris & Seaborne, 2013). RDF data is not only serialized in static documents, they reside in RDF databases (RDF Data Store or Triple Store), where RDF triples can be store and retrieved using a query language (usually SPARQL) (Jacyntho, 2012).

*Syntaxes*: RDF is an abstract data model. The use of concrete standard syntaxes for the representation of RDF triples is required before these graphs to be actually published on the Web. The most popular syntaxes are: RDF/XML (Gandon & Shreiber, 2014) and Turtle (Beckett *et al.*, 2014).

*Web of Data topology*: a significant number of individuals and organizations have been publishing their data as Linked Data and, hence, the Web of Data is growing rapidly. The Web of Data topology (Cyganiak & Jentzsch, 2014) consists of data sources connected by RDF links. A link between two data sources means that there is at least one triple in which the subject resides in the source data source and the object resides in the target data source. One of the most reused data sources is DBpedia <http:// dbpedia.org>, which is the Linked Data version of the Wikipedia data.

## Methodological procedures

This section covers the proposed Linked Data application for semantic management of scanned documents, comprising the methodological procedures used throughout its development cycle.

The application consists of a digital document management system based on the Semantic Web that has the following functional requirements: *register author*, *register document*, *search author*, and *search document*.

To register a document the user enters the title, the authors, the language, a small description and a few keywords related to the document. After this, the user submits the document file, which must be in RDF format, in English or Portuguese. To register an author, the user provides the author's name and email.

In both cases for documents or authors, the user requests the system to analyze the entered data to perform the *Linked Data mashup* with resources from *DBpedia*. *DBpedia Lookup* <http://wiki.dbpedia.org/Lookup> and *DBpedia Spotlight* <https://github.com/dbpedia-spotlight/dbpedia-spotlight/wiki> Web services are used to accomplish this task. *Lookup* analyzes individual words, such as title and keywords of the document, to find candidate resources related to the document and uses the name of the author to find candidate resources related to author. *Spotlight* is used to make natural language processing of texts, as the description of the document, and extracts candidate resources related to the text. For both documents or authors, the system displays a list containing, for each resource returned from the Web services, its description, label, and URI that allow the user to approve the resources that are related to the document or author.

By registering a new document or author, a URI is automatically generated for each one, which allows their identification on the system and the whole Web of Data. Each information inserted is mapped by the system to a property of an ontology. The ontology used for documents is *Dublin Core* (DC) (Dublin Core Metadata Initiative, 2012), and *Friend of a Friend* (FOAF) (Brickley & Miller, 2014) for people (authors).

In addition, each *DBpedia* resource returned through the title, keywords and description, and which was approved by the user, is related to the document by using the *rdfs:seeAlso* (which means a related thing), a property of the *RDF Schema* (RDFS) ontology. Similarly, each *DBpedia* resource based on the author's name is related to the author through the *owl:sameAs* (which means the same thing), a property of the *Web Ontology Language* (OWL) ontology.

The system allows searching for documents by title, description or keyword. The retrieved documents have their data displayed in a list, as well as a link to view more details. The detail link activates a new page which displays all the information of the selected document, allowing the visualization of the PDF file, editing and deleting of the document. Links to *DBpedia* resources (obtained through *Lookup* and *Spotlight* Web services) offer more information about the document and author(s).

It is also possible to perform advanced searches, such as documents by author and semantic/Linked Data searching. When searching for documents by author, a list of authors is displayed and one can view the documents associated with each author by simply selecting the desired author. When an author is selected, a page containing the list of documents associated with him/her is displayed. In semantic/Linked Data searching, a list of topics related to documents registered in the system is displayed, enabling the selection of the desired topic. These topics are obtained through the *rdfs:label* (a representative label) property of resources associated with documents through the *rdfs: seeAlso* property. When we select one of these topics, the system displays a list of documents related to the topic. Semantic searching can be done in English or Portuguese.

The system allows one to search for the authors by name or email. The retrieved authors have their data displayed, as well as a link to view more details. The detail link displays all information of the selected author and links to resources from *DBpedia*, containing more information about the author. The page also allows the author to be edited and deleted.

It is noteworthy that all these interesting processes occur behind the scenes, without the end user being aware. Although it is a semantic Linked Data system, from the point of view of human-computer interaction, it works like a conventional Web application.

*Domain class model*: A conceptual class model was first built, formed by the following classes: *Domain Object, Author, Document and Language*.

The *Author* class has *name* and *email* attributes. The *Document* class has *title*, *keywords*, and *description* attributes. A document has one or more authors and a language (represented by the *Language* class), which can be Portuguese or English.

*DomainObject* is the superclass of *Author* and *Document* classes. *DomainObject* contains the characteristics common to all classes, which, for now, is only the *id* attribute (used to generate the URI), which identifies the document or author.

## Mapping of the class model into Linked Data ontologies

The classes (and their attributes) of the conventional model were mapped in the selected *Linked Data* ontologies (dc, foaf, rdfs and owl). This mapping means that when the object of the class is mapped to the RDF model, forming triples *resource-property-value*, the resource will be the object mapped to its URI (generated from the *id* attribute); the property will be a property of some ontology; and the value will be the attribute's value. The schema below demonstrates the domain model classes and their attributes and the ontological properties used in the mapping.

*DomainObject class* → owl: Thing
- *seeAlso attribute* → rdfs:seeAlso
- *sameAs attribute* → owl:sameAs.

*Author class* → foaf:Person
- *name attribute* → foaf:name
- *email attribute* → foaf:mbox.

Document class → dc:BibliographicResource
- *title attribute* → dc:title
- *keywords attribute* → dc:subject
- *language attribute* → dc:language
- *description attribute* → dc:description
- *identifier attribute* → dc:identifier
- *authors attribute* → dc:creator.

*Language class* → codes "pt" and "en" from the international standard ISO-639-1 <http://www.iso.org/iso/home/standards/language_codes.htm>, corresponding to the Portuguese and English languages, respectively.

Taking as example the *Author* class that has the *name* attribute associated with the *foaf:name* property, when an *Author* object is mapped to RDF, the triple regarding the author's name will have *foaf:name* as property and the *name* as value.

The *Document* class is mapped to the *dc:BibliographicResource* class of the *Dublin Core* ontology that classifies the document by using the *rdf:type* property, i.e. the document is classified by a triple formed by its URI, the *rdf:type* property and the *dc:BibliographicResource* class as value, expressing that documents are bibliographic resources. The *Dublin Core* properties were used as follows: *dc:title* (document's title); *dc:creator* (document's authors); *dc:language* (document's language); *dc:subject* (document's keywords); *dc:description* (document's description); and *dc:identifier* (defines an unambiguous ID for the document through the string representing the location (path) of the PDF document file on the server). The *Author* class, in turn, is mapped to the *foaf:Person* class of the FOAF ontology, which states that each author is a person through triples formed by the author's URI, the *rdf:type* property and the *foaf:Person* class as value. The FOAF properties were used as follows: *foaf:name* (author's name) and *foaf:mbox* (author's e-mail). The *rdfs:seeAlso* property of the *RDF Schema* ontology and *owl:sameAs* property of the OWL ontology were used to make the *Linked Data mashup* with the Web of Data (specifically with the *DBpedia* data source). The *rdfs:seeAlso* stores URIs from other Web resources containing information related to authors or documents. The *owl:sameAs* stores other URIs from the Web of Data that also identify the author or document in question.

*Example of application use*: this section demonstrates the main features of the application. The document data used as an example are: *URI*: <http://www.example. com/documents/1>; *title*: "Yet another optimization article"; *keywords*: "Software performance", "Software architecture", "Optimization"; *description*: "It is a short article that talks about the importance to optimize software performance."; *language*: "English"; *PDF file*: "YetOptimization.pdf"; *related URIs*: <http://www.dbpedia.org/resource/Software_architecture> and <http://www.dbpedia.org/resource/Software_performance_testing>. The author data are: *URI*: <http://www.example.com/authors/1>; *name*: Martin Fowler; and *email*: martin@gmail.com.

*Registering an author*: to register an author, the user fills in the name and e-mail of the author and asks the system to analyze them. The system accesses the *DBpedia* Web services, searching for information related to the author's name, and displays the candidate resources (represented by URIs from *DBpedia*) and shows their descriptions. Finally, the user selects the *DBpedia* resources that s/he wants to associate with the author to perform the Linked Data mashup and inserts the author into the system.

For the author "Martin Fowler," for example, the mashup candidate resources returned from *DBpedia* would be: <http://dbpedia.org/resource/Martin_Fowler>; <http://dbpedia.org/resource/Martin_ Fowler_(footballer)>; among others. The description of the resource represented by the URI <http://dbpedia.org/resource/Martin_Fowler> says that this resource represents a British software engineer named Martin Fowler; the description of the resource <http://db pedia.org/resource/Martin_Fowler_(footballer)> says that this resource represents a professional footballer.

Thus, at registration, the user should evaluate the descriptions and select the resources that, in fact, are related to the author registered, in this case, the software engineer. As author registration is relatively simple, for reasons of space, the screen has been omitted here.

*Registering a document*: to register a document, the user enters the document's data and associates its PDF file (Figure 1).

After filling out all the information, the user requests the system to analyze it. The system accesses the *DBpedia* Web services by searching for information related to the title, description and keywords of the document and displays the returned candidate resources, showing their descriptions and URIs. Finally, the user selects the resources that s/he wants to associate with the document and inserts the document into the system.

Chart 1 displays the RDF triples stored internally in the knowledge base of the system, after registration of the document shown in Figure 1. Note the use of the classes and properties of the ontologies, as well as the *Linked Data mashup* with the *DBpedia,* denoted by the *rdfs:seeAlso* property.

*Searching for an author*: To search for authors, the user provides the name or email of the author. After searching, a list of retrieved authors is displayed, and beside each author, a detail link leads the user to a page that displays information about the author, including *DBpedia* resources related to him/her. By means of the resources obtained in *DBpedia* (associated with the author at registration), links to *DBpedia* Web pages are shown, containing more information about the author. It is worth noting that the description of the author is enriched with *Linked Data* information from *DBpedia* without the user realizing, greatly improving the user experience.

*Searching for a document*: To search for the example document in the database (Figure 2), its title is partially informed (*yet*) and, below, the document data are displayed.

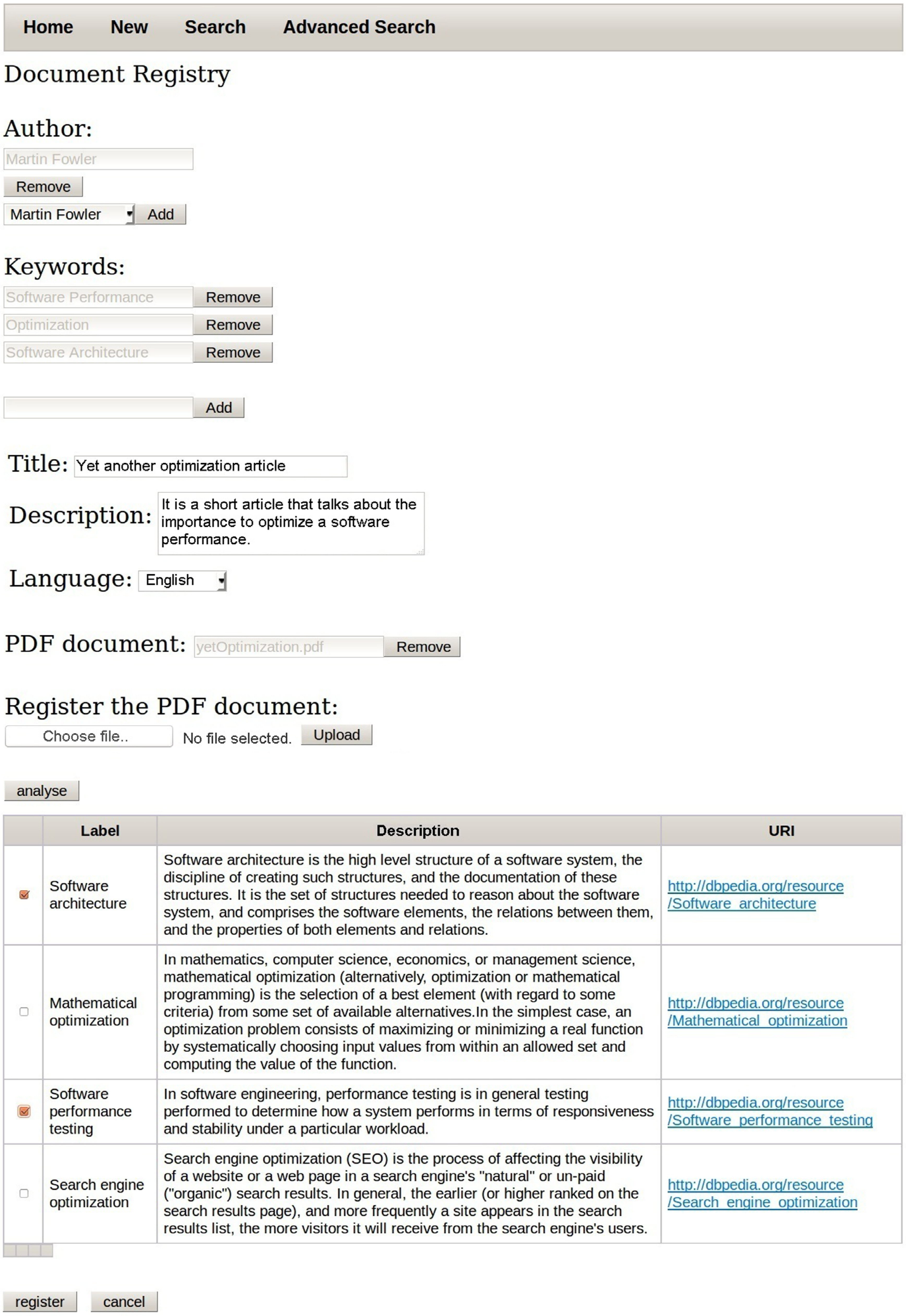
The detail page (Figure 3) displays all information about the document searched, including *DBpedia* resources related to the document and its authors, which are presented as pages containing additional information, accessed via links.

The association of these (meta)data from *DBpedia* to the document was made at registration and provides the user with easy access to Web data of interest, quickly and accurately and, above all, without the user being aware of the internal complexity of the mashup.

**Chart 1**. Resource Description Framework triples generated at registration of the document.

| Resource | Property | Value |
|---|---|---|
| http://www.example.com/documents/1 | dc:type | dc:BibliographicResource |
| http://www.example.com/documents/1 | dc:subject | "Software performance" |
| http://www.example.com/documents/1 | dc:subject | "Software architecture" |
| http://www.example.com/documents/1 | dc:subject | "Optimization" |
| http://www.example.com/documents/1 | dc:title | "Yet another optimization article" |
| http://www.example.com/documents/1 | dc:description | "It is a short article that talks about the importance to optimize a software performance." |
| http://www.example.com/documents/1 | dc:language | "en" |
| http://www.example.com/documents/1 | rdfs:seeAlso | http://www.dbpedia.org/resource/Software_architecture |
| http://www.example.com/documents/1 | rdfs:seeAlso | http://www.dbpedia.org/resource/Software_performance_testing |
| http://www.example.com/documents/1 | dc:creator | http://www.example.com/authors/1 |
| http://www.example.com/authors/1 | rdf:type | foaf:Person |
| http://www.example.com/authors/1 | foaf:name | "Martin Fowler" |
| http://www.example.com/authors/1 | foaf:mbox | mailto:martin@gmail.com |

Source: Prepared by the authors (2015).



Document Registry

Author: Martin Fowler

Martin Fowler

Keywords: Software Performance; Optimization; Software Architecture

Title: Yet another optimization article

Description: It is a short article that talks about the importance to optimize a software performance.

Language: English

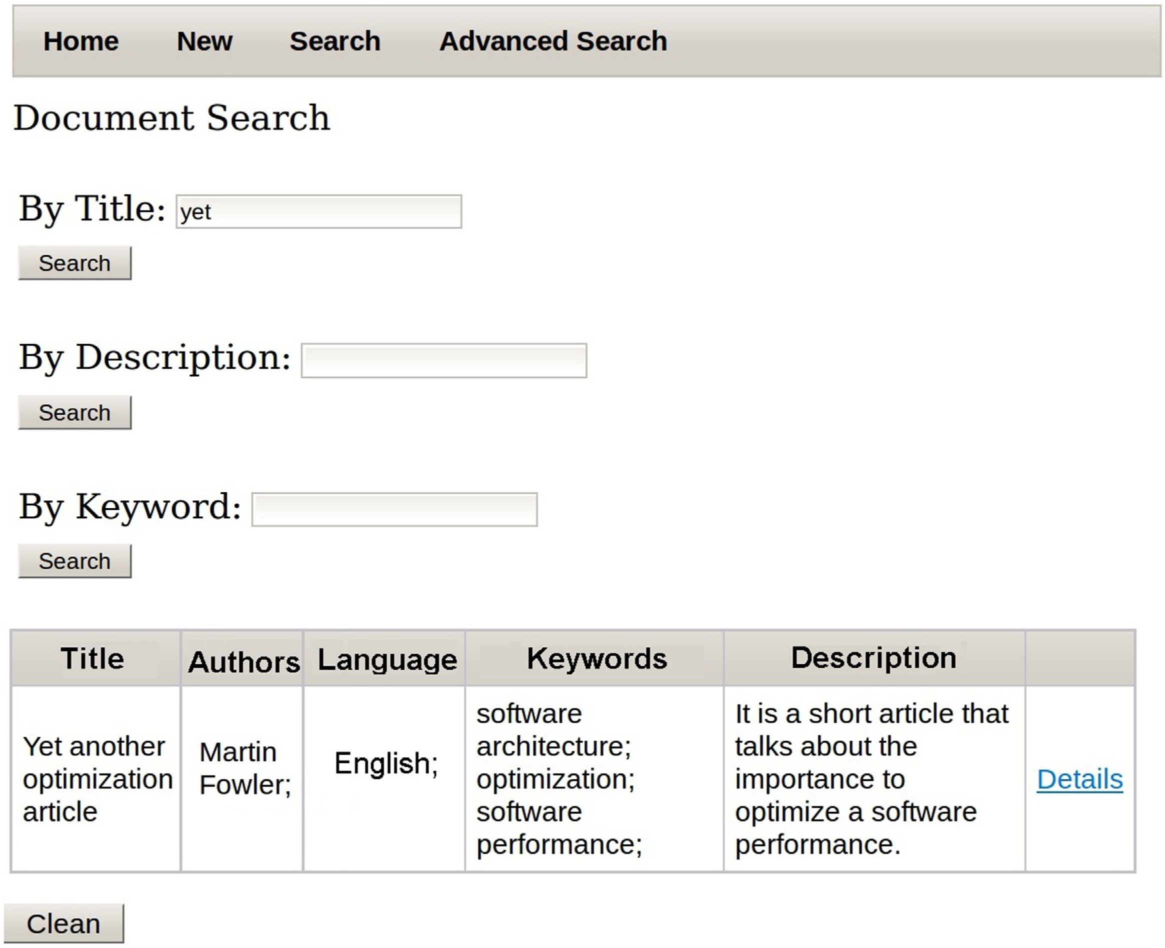
PDF document: yetOptimization.pdf

Register the PDF document:

| | Label | Description | URI |
|---|---|---|---|
| ☑ | Software architecture | Software architecture is the high level structure of a software system, the discipline of creating such structures, and the documentation of these structures. It is the set of structures needed to reason about the software system, and comprises the software elements, the relations between them, and the properties of both elements and relations. | http://dbpedia.org/resource/Software_architecture |
| □ | Mathematical optimization | In mathematics, computer science, economics, or management science, mathematical optimization (alternatively, optimization or mathematical programming) is the selection of a best element (with regard to some criteria) from some set of available alternatives.In the simplest case, an optimization problem consists of maximizing or minimizing a real function by systematically choosing input values from within an allowed set and computing the value of the function. | http://dbpedia.org/resource/Mathematical_optimization |
| ☑ | Software performance testing | In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. | http://dbpedia.org/resource/Software_performance_testing |
| □ | Search engine optimization | Search engine optimization (SEO) is the process of affecting the visibility of a website or a web page in a search engine's "natural" or un-paid ("organic") search results. In general, the earlier (or higher ranked on the search results page), and more frequently a site appears in the search results list, the more visitors it will receive from the search engine's users. | http://dbpedia.org/resource/Search_engine_optimization |

**Figure 1**. Registering a document.

Source: Prepared by the authors (2015).



Document Search

By Title: yet

By Description:

By Keyword:

| Title | Authors | Language | Keywords | Description | |
|---|---|---|---|---|---|
| Yet another optimization article | Martin Fowler; | English; | software architecture; optimization; software performance; | It is a short article that talks about the importance to optimize a software performance. | Details |

**Figure 2**. Searching for a document.

Source: Prepared by the authors (2015).

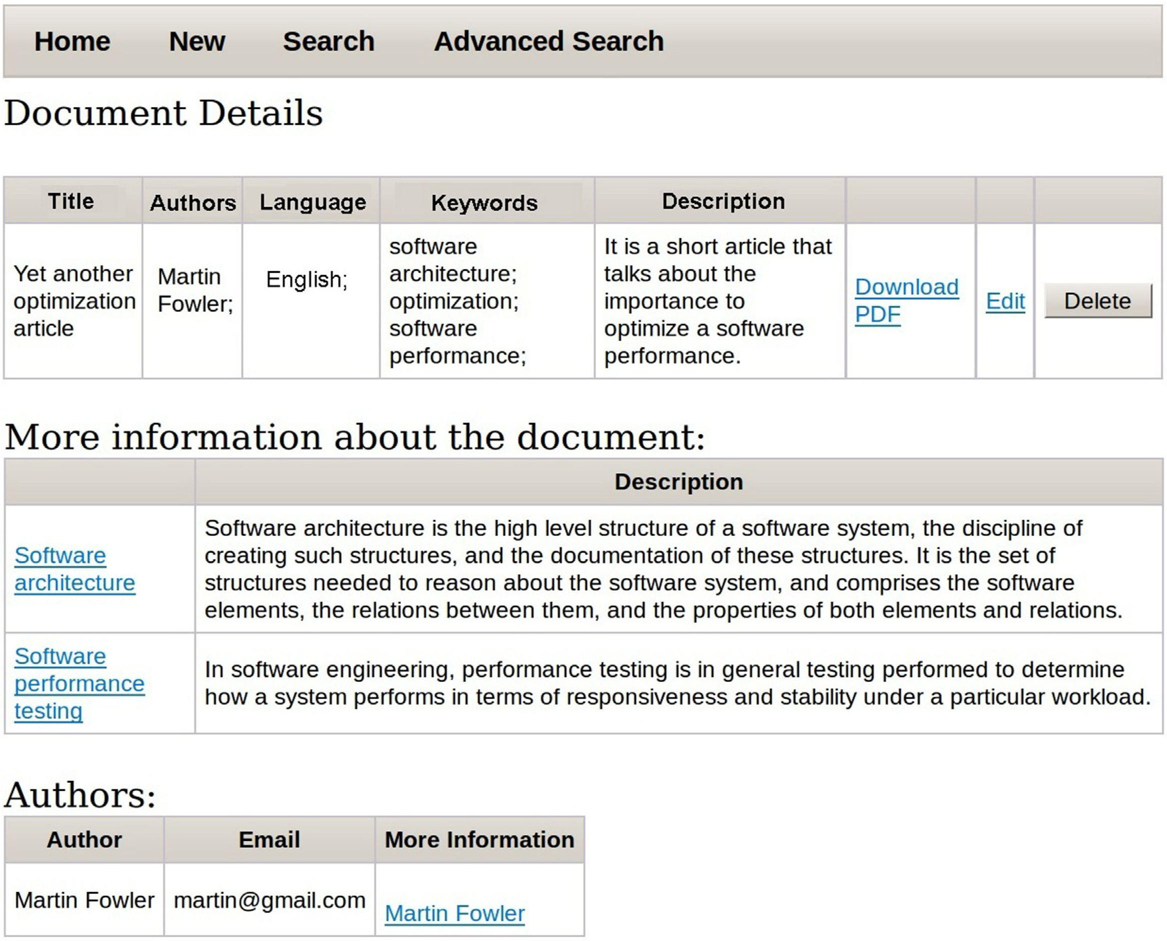
*Searching documents by author*: when searching for documents by author, the system displays a list containing the names of the registered authors as links. To view the documents, the user only needs to select the link of the desired author.

The precise searching for documents by author is possible thanks to the structured metadata description of the documents. The authors are related to documents via the *dc:creator* property, so to find the documents corresponding to an author, the machine searches for resources that comprise triples (*resource-property-value*) containing *dc:creator* as property and containing the selected author's URI as value. If the search were only based on the author's name as a simple keyword (literal value), the result would not be so accurate.

### Semantic and Linked Data searching in English

In the semantic and Linked Data searching in English (Figure 4), the system displays a list of topics (*DBpedia* resources) related (by the *rdfs:seeAlso* property) to at least one English document registered in the system. In Figure 4, by clicking on topic "Software architecture", a list of related documents is displayed, containing document information along with a detail link, which allows the visualization of all document information. In this example, after the selection of a topic, only one document was displayed, but each topic can be related to several different documents.

This is another example of searching that shows how the user benefits, when documents are described with semantic/structured metadata and enriched with



Document Details

| Title | Authors | Language | Keywords | Description | | | |
|---|---|---|---|---|---|---|---|
| Yet another optimization article | Martin Fowler; | English; | software architecture;<br>optimization;<br>software performance; | It is a short article that talks about the importance to optimize a software performance. | Download PDF | Edit | Delete |

More information about the document:

| | Description |
|---|---|
| Software architecture | Software architecture is the high level structure of a software system, the discipline of creating such structures, and the documentation of these structures. It is the set of structures needed to reason about the software system, and comprises the software elements, the relations between them, and the properties of both elements and relations. |
| Software performance testing | In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. |

Authors:

| Author | Email | More Information |
|---|---|---|
| Martin Fowler | martin@gmail.com | Martin Fowler |

**Figure 3**. Document details page.
Source: Prepared by the authors (2015).

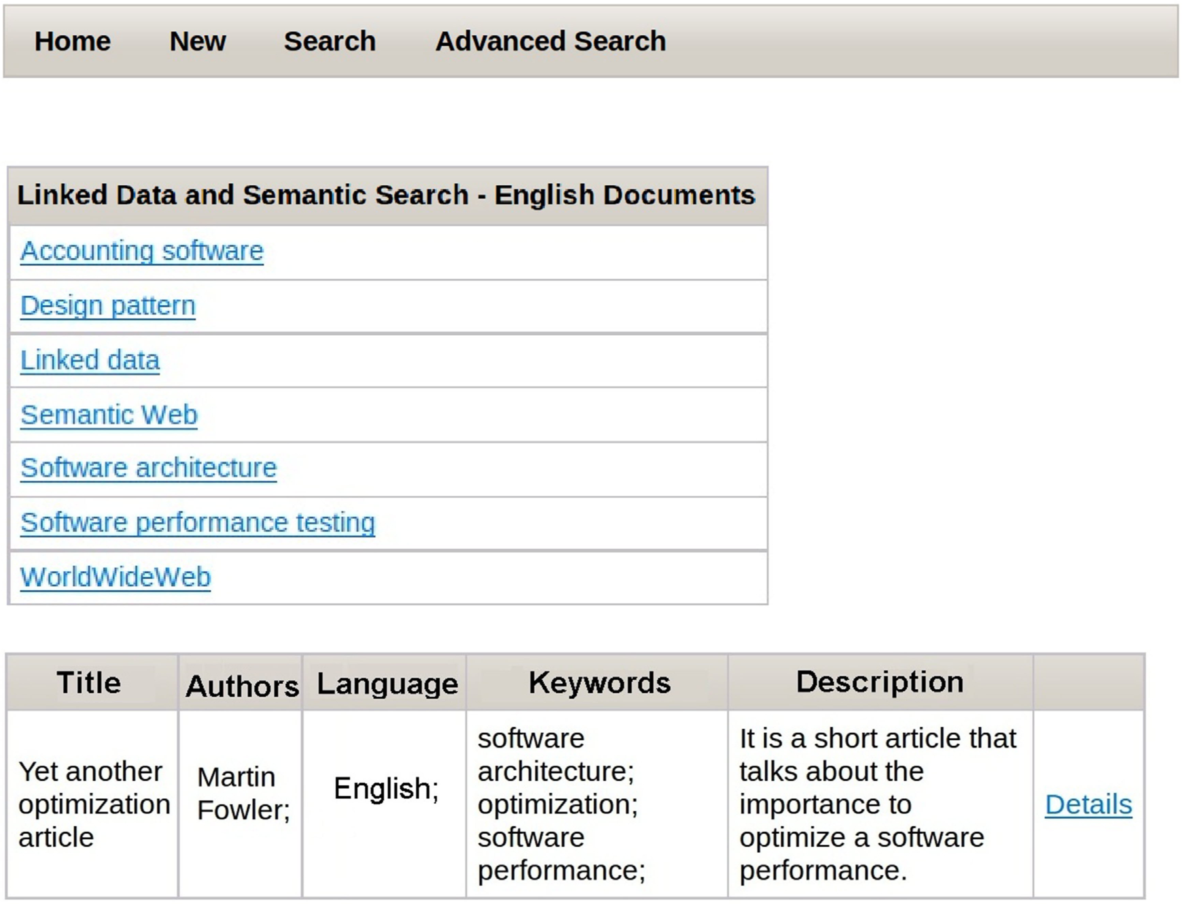
data obtained from the *DBpedia* (*Linked Data mashup*). Each topic in Figure 4 represents the *rdfs:label* property of a resource associated with at least one document registered in the system. Note that through reuse of data (resources) obtained from the Web of Data, we can associate documents to certain subjects (semi)automatically, providing the user with more complete and effective document search. In other words, a kind of semantic *folksonomy* was created, where each tag is not just a syntactic word but a *DBpedia* resource with explicit meaning described in RDF, thereby avoiding ambiguity.

The application also offers semantic searching in Portuguese, which works the same way as the semantic searching in English, differing only by the language of the topics and the returned documents.

*Related work*: This section presents some related work, comparing them with the proposed application.

*Alfresco* <http://www.alfresco.com> is an open source content management system, developed in Java, which has the main feature of managing digital documents. In *Alfresco DevCon*, 2012 event, the presentation *Alfresco & the Semantic Web* <http://www.zaizi.com/blog/semantic-technologies-in-alfresco> encouraged the use of *Alfresco* with semantic technologies.

*Nuxeo* <http://www.nuxeo.com> is an open source content management platform, which allows the management of digital documents. In *Solutions Linux Conference,* 2011 the presentation *Introduction to the Semantic Web* encouraged the development of Semantic Web-based systems using *Nuxeo*.



| Linked Data and Semantic Search - English Documents |
|---|
| Accounting software |
| Design pattern |
| Linked data |
| Semantic Web |
| Software architecture |
| Software performance testing |
| WorldWideWeb |

| Title | Authors | Language | Keywords | Description | |
|---|---|---|---|---|---|
| Yet another optimization article | Martin Fowler; | English; | software architecture; optimization; software performance; | It is a short article that talks about the importance to optimize a software performance. | Details |

**Figure 4**. Semantic and Linked Data searching in English.

Source: Prepared by the authors (2015).

*Differences and similarities*: *Nuxeo* and *Alfresco* are systems which have been designed without the use of Semantic Web technologies whose purpose is conventional management of scanned documents. In contrast, the application proposed in this study uses the Semantic Web and has the following peculiarities: the document data is stored in a machine-understandable way, enabling automatic (with the aid of machine) and more accurate searching; consumption of structured data from the new Web of Data; enrichment of the document semantic description; and improvement in the decision-making process.

*Nuxeo* and *Alfresco* developers encourage software engineers to use the Semantic Web to develop semantic document management applications along with these two open source platforms.

## Conclusion

The use of the Semantic Web in applications for management of scanned documents is very useful for knowledge management in organizations. The Semantic Web does not only store and retrieve information, but it does so with the aid of the machine, which can suggest documents and also deduce new knowledge from registered information, since this information is structured metadata (understandable by the machine) created on the basis of formal ontologies that describe the meaning of stored documents. In addition, the semantic metadata can and should be enriched through Linked Data mashup with other Web data sources such as *DBpedia*, expanding the searching and navigation power through documents.

In this article, a semantic Linked Data application that meets these requirements was described. With this application, we hope that the process of cataloguing and searching for scanned documents become more efficient and precise, transferring much of the work to the machine.

This study consumes (reuses) Linked Data information. However, in the future, the project will also publish RDF Linked Data for others to consume. To this end, two requirements are important: (1) making all URIs dereferenceable, returning according to HTTP content negotiation, HTML content (for humans) and RDF (for software agents); (2) providing a public SPARQL endpoint for querying the system knowledge base.

Performing mashup with other Linked Data sources in addition to *DBpedia*, such as *Freebase* <http://www.freebase.com/>, *RDF Book* Mashup <http://wifo5-03.informatik.uni-mannheim.de/bizer/bookmashup/>, *Geonames* <http://www.geonames.org/>, etc., is of paramount importance to increase the knowledge base.

We can still suggest SPARQL queries with inference from the ontologies used, as well as federated queries combining our application's database with the data sources used for mashup.

## Collaboration

All authors contributed to the conception and design of the study, data analysis and final essay.

## References

Beckett, D. *et al. RDF 1.1 Turtle*: Terse RDF triple language. W3C Recommendation 25 Feb. 2014. Available from: <http://www.w3.org/TR/turtle/>. Cited: Apr. 29, 2015.

Berners-Lee, T.; Hendler, J.; Lassila, O. The semantic Web. *Scientific American*, v.284, n.5, p.29-37, 2001.

Berners-Lee, T. *Linked data*: Design issues about Web architecture. 2006. Available from: <http://www.w3.org/DesignIssues/LinkedData.html>. Cited: Sept. 2, 2014.

Brickley, D.; Guha, R. V. *RDF Schema 1.1*. W3C Recommendation 25 Feb. 2014. Available from: <http://www.w3.org/TR/rdf-schema/>. Cited: 29 abr. 2015.

Brickley, D.; Miller, L. *FOAF vocabulary specification 0.99*. 2014. Available from: <http://xmlns.com/foaf/spec/>. Cited: May 11, 2015.

Cyganiak, R.; Jentzsch, A. *The linking open data cloud diagram*. 2014. Available from: <http://lod-cloud.net>. Cited: Nov. 26, 2014.

Cyganiak, R.; Wood, D.; Lanthaler, M. *RDF 1.1*: Concepts and abstract syntax. W3C Recommendation 25 Feb. 2014. Available from: <http://www.w3.org/TR/rdf11-concepts/>. Cited: 29 abr. 2014.

Dublin Core Metadata Initiative. *DCMI Metadata terms*. 2012. Available from: <http://dublincore.org/documents/dcmi-terms/>. Cited: May 11, 2015.

Gandon, F.; Shreiber, G. *RDF 1.1*: XML syntax. W3C Recommendation 25 Feb. 2014. Available from: <http://www.w3.org/TR/rdf-syntax-grammar/>. Cited: Apr. 29, 2015.

Harris, S.; Seaborne, A. *SPARQL 1.1*: Query language. W3C Recommendation 21 Mar. 2013. Available from: <http://www.w3.org/TR/sparql11-query/>. Cited: Apr. 18, 2015.

Heath, T.; Bizer, C. *Linked data*: Evolving the web into a global data space. California: Morgan & Claypool, 2011.

Heflin, J. *OWL web ontology language use cases and requirements*. W3C Recommendation 10 Febr. 2004. Available from: <http://www.w3.org/TR/webont-req/>. Acesso em: Sept. 2, 2014.

Jacyntho, M.D.A. *Um modelo de bloqueio multigranular para RDF*. 2012. Tese (Doutorado em Informática) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2012. Disponível em: <http://www.maxwell.vrac.puc-rio.br/Busca_etds.php? strSecao=resultado&nrSeq=20236@2>. Acesso em: 18 abr. 2015.

Mcguiness, D.L.; Harmelen, F.V. *OWL Web Ontology Language*: Overview. W3C Recommendation 10 Feb. 2004. Available from: <http://www.w3.org/TR/owl-features/>. Cited: Apr. 29, 2015.