Using Search Paradigms and Architecture Information Components to Consume Linked Data

Carlos H. Cordoví-García[1], Claudia Hernández-Rizo[1], Yusniel Hidalgo-Delgado[2], Liudmila Reyes-Álvarez[3]

[1] Electronic Goverment Center, University of Informatics Sciences, Havana, Cuba.
[2] Programming Techniques Department, Faculty 3, University of Informatics Sciences, Havana, Cuba.
[3] Industrial Informatic Center, University of Informatics Sciences, Havana, Cuba.
{ccordovi, crizo, yhdelgado, lreyes}@uci.cu

Abstract. The success of the Linked Open Data Initiative has increased the amount of information available on the Web. However, the Web content published under this initiative cannot be consumed by users who are unfamiliar with Semantic Web technologies (RDF, SPARQL, Ontologies, etc.), because they need to understand the structure, provenance and the way in which data are queried, and this can be complex for non-tech-users. In this paper the process development of a Web application is described, which uses components borrowed from Information Architecture and search paradigms applied to the task of consuming Linked Data by non-tech-users. These data are available via a SPARQL endpoint (it is a SPARQL protocol service that enables users to query a knowledge base known as RDF triples database or triplestore), which can be queried through a HTTP protocol from a Web browser. This proposal allows full-text search over a bibliographic dataset and faceted search, based on representative concepts of it (facets). Using these paradigms allows us to consume Linked Data in an intuitive and friendly way, which reduces the technical barrier that limits users in this process. In addition this proposal has been tested through a complexity analysis study. This study calculates complexity metrics, by measuring usability dimensions based on user experience for executing search tasks.

Keywords: Faceted Search, Full-Text Search, Linked Data, Ontologies

1 Introduction

Since its origins the current Web has represented one of the greatest technological advances for humanity. It radically changed the way in which socioeconomic activities were developed before the 90´s. However, the current Web (Web 2.0) has several limitations that do not allow users to enjoy its full potential: these limitations are format, integration and retrieval of resources available on the same[1]. This implies there are no mechanisms that allow automatic processing of information: the resources available on the Web are structured based on

Hypertext format, well known by its acronym HTML. Documents structured in this format (called Web pages) can be understood easily by humans and Web browsers; however they cannot be processed by a machine and it is therefore impossible to extract their semantic value automatically.

On the other hand, the information on the Web appears dispersed, and there are no explicit relationships between the different resources available: this produces ambiguity in the information[2]. Format and integration issues cause the information retrieval to be affected too, as is evidenced in the search engines most commonly used today. They are imprecise, and in many cases do not satisfy the users search needs, because they answer queries based on keywords. Therefore they are unable to retrieve information from queries expressed in natural language[3].

Taking into account these limitations and the need to solve them in order to achieve a Web evolution, where society´s current information challenges can be addressed. In 2001, Tim Berners Lee, the Web creator stated: "The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation"[4].

The transition to the Semantic Web requires information to be properly structured and integrated. In 2006, this led Tim Berners Lee himself to state that the concept of Linked Data: "...refers to a set of good practices for the publication and link structured data on the Web"[4].

Linked Data has become an active research area in recent years. This is due to the need to publish and consume structured data on the Web, thereby enhancing the development of the Semantic Web. This paper presents an approach for consuming Linked Data from the thematic domain of bibliographic metadata. This metadata has been collected from open access journals available on the Web and published as structured data following the Linked Data principles presented in[5].

Currently bibliographic metadata are published as Linked Data and they are accessible via SPARQL endpoints. However, for some users who are unfamiliar with Semantic Web technologies (RDF, SPARQL, Ontologies, vocabularies and metadata) commonly called non-tech-user or lay-user[2]; it is not very intuitive to use these data because they do not understand the nature and structure of the data, in addition they are not sure how to consult them.

When a user makes a request of a resource through its uniform resource identifier (URI), then useful information related to requested resource is returned[5]. So, when an URI is derefered[6] a response according to parameters specification in the request is returned: it can be returned in HTML format, in which case it is easily interpreted by a Web browser and displayed in an intuitive way to the user. However, it is usually returned in a type of RDF serialization [RDF / XML format[7], N3[8] Turtle[9]]. Then how to interpret and use this format is restricted to just tech-savvy users[2] and in some cases, for those users with knowledge about Semantic Web technologies.

This paper proposes a Web application that reuses and adapts components borrowed from Information Architecture (IA), to enable users who have no technical knowledge of Semantic Web technologies (non-tech-users), but who do have knowledge of computing and communications technologies. So, the user can consume (visualizing, browse, query, etc.) Linked Data through full text search and faceted search on a bibliographic metadata set that has previously been published as Linked Data. These IA components are present in most common Web pages, they are for instance: navigation facets, breadcrumbs, browsing menus, among others.

In addition two search paradigms have been implemented on this Web application: interactive-exploratory paradigm and key word search paradigm. Both are approaches commonly used in current Web applications; because they improve the user experience and make information easier to process, searching in unknown datasets specifically in new search scenarios.

The rest of this paper is organized as follows: First, the related work is presented in Subsection 1.1. Then, the materials and methods used for developing the proposed approach are detailed in Section 2. Section 3 presents the approach itself, while in Subsection 3.1 the use of IA components is discussed. Subsection 3.2 introduces the search paradigms used and Subsections 3.2.1 and 3.2.2 detail each of them. An evaluation based on usability and complexity indicators in Linked Data consumption processes has been carried out using a complexity analysis study. The results of this study are presented in Section 4. Finally, conclusions and future work are presented in Section 5.

2 Related Work

The way to promote the consumption (visualization, presentation and use) of data sources published as Linked Data in an intuitive and friendly way for web users has been a thematic considered in various projects. Software applications developed for this purpose (linked data applications) can be classified into two categories: generic applications and domain-specific applications. Generic linked data applications can process data from any thematic domain, for instance: library as well as life science data[10].

There are two basic types of generic linked data applications: linked data browsers and linked data search engines. Just as traditional web browsers allow users to navigate between HTML pages by following hypertext links, linked data browsers allow users to navigate between data sources by following RDF links[5].

Examples of linked data browsers are the Disco Hyperdata[5], Tabulator[11] and Marbles. However these applications have a common limitation: They do not allow a general view of the dataset to be navigated, because they only provide information on current resources. This means that they only visualize information on resources through the URI that represents them.

On the other hand linked data search engines improve the user experience compared to traditional searching methods, because it is performed over structured data. Nevertheless, in existing applications that belong to this group it is not possible to know a priori the general characteristics of a dataset such as its main resources, its properties and values. Examples of this type of application are Sig.ma[12] VisiNav[13] and Swoogle[10].

There are also various linked data applications that cover the needs of specific community users. In this category are to be found two types of applications: linked data mashups and other domain-specific applications. Linked data mashups are applications created with the objective of integrating information from heterogeneous sources (intertwined datasets on the Web of Data) to be used for satisfying the information needs of specific user communities[5]. Some applications that belong to this group are E.U: US Global Foreign Aid Mashup and Paggr[14]. These kinds of applications are powerful tools because they allow the recovery of interrelated information from different datasets. However they do not solve the issue of the characterization of datasets through a general view of their resources and properties.

Current tools used in Linked Data consumption, fail to help users to explore datasets so as to discover what kind of resources can be found there, what are their properties are or their interrelationships. Therefore, these tools are not able to show a general view of a Linked Dataset to users in order to describe its resources and properties.

3 Materials and Methods

The proposed approach has been developed using Semantic Web technologies such as: RDF , SPARQL and Ontologies[6]. It has also made use of the programming language on the server side, PHP, the algorithms library EasyRDF , the framework for developing JavaScript applications, JQuery, and the general purpose server, Virtuoso. In the Semantic Web field the way of representing, structuring and organizing the semantic of the data is fundamental. It is exactly here where the use of Ontologies is essential: An Ontology is a formal and explicit specification of a shared conceptualization[6].

So, in order to publish bibliographic metadata as Linked Data, in proposed approach an ontological model has been established (see Figure 1) using widely used Ontologies that cover the domain of bibliographic data.

As said, it is necessary to represent data through concepts contextualized in a domain. However, it is indispensable to keep data following linked data principles and to guarantee standardization and compatibility. That is why the W3C acronym for the World Wide Web Consortium, established as standard, the Resource Description Framework (RDF). It is a simple graph-based data model that has been designed for use in the context of the Web[7]. In RDF, a description of a resource is represented as a number of triples. The three parts of each triple are called its subject, predicate, and object[5].

Although RDF is the standard for representing Linked Data, it is still necessary to query data from this model: SPARQL is a query language for RDF graphs that has been defined by the W3C as a standard[8].Its syntax is similar to SQL, but oriented to RDF triples. Results of queries can be sets of RDF

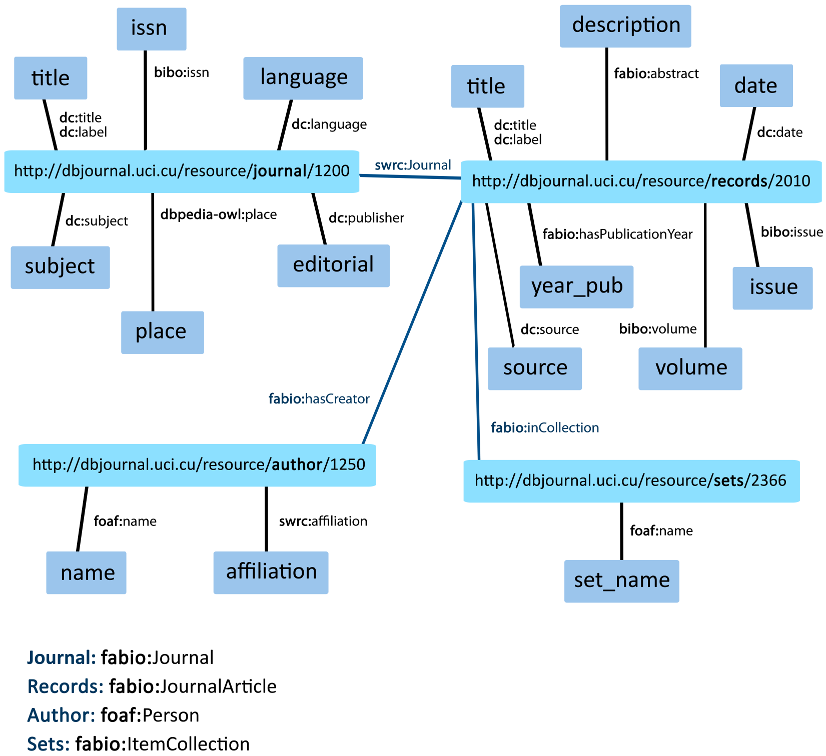

Fig. 1. Ontological Model

triples, RDF graphs, URIs of resources or values (for example literals). For this proposal SPARQL 1.1, has been used, which is the most recent specification of this standard.

In order to easily produce and consume data stored in RDF format EasyRDF[9] has been used, it is an algorithm library written in Object Oriented Programming language PHP. This library eases performing SPARQL queries via HTTP to a triplestore using the EasyRdf_Sparql_Client class that returns PHP objects as result.

Although JQuery is a frameworkfor developing JavaScript applications, it is a code library that contains ready-to- use processes and routines. In addition, it allows easy handling of asynchronous requests made to a Web server through AJAX. OpenLink Virtuoso Universal Server is a hybrid store solution for a range of data models; it includes relational data, RDF and XML as well as free text documents. It offers two use variants; the first one is Virtuoso Open Source Edition (mostly used by developers because it is free code) and the other is a commercial edition that requires paying for a license fee to use it. Virtuoso directly offers a SPARQL Endpoint that allows the resources contained in the server to be consulted.

Besides this, it provides a management tool through an interface called Virtuoso Conductor, which has complete access to all the features available. The proposal has been guided by a development agile approach using extreme Programming methodology (XP) because of certain features mentioned in[15] which makes it a suitable option for development.

4 Approach

The approach proposed here is a Web application which includes IA components and implements search paradigms in order to potentiate Linked Data consumption by both types of user: experts and non-experts. So, the web application architecture has been built on three layer and two physical levels (See Figure 2). The first one is the application level, it contains two stages: presentation layer and business layer including all technology support needed. The second one is the data level containing the data layer, here is where a special service called SPARQL endpoint is running and it allows requests to be answered that have been sent from the business layer as a result of a user s need for information formulated by them in the presentation layer.

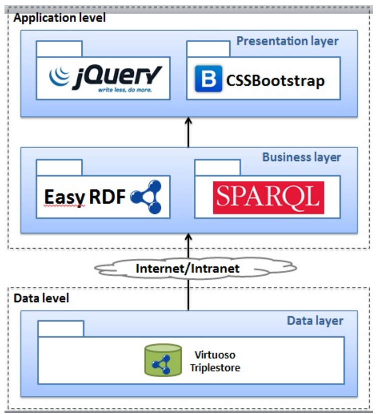

Fig. 2. Architecture view of solution proposal

An advantage of having three layers of architecture is to keep view and model separate. It allows having several views while keeping the same model. So, one of the weaknesses of current Linked Data consumption tools is that they fail to offer a general description of the dataset on which they are working. But if views can be shown, using graphs or another statistical element to describe the resources inside the dataset, this issue could be solved. The proposed approach builds a view including a bar graph showing the amount of articles written by year, also the number of each resource inside dataset is presented as Figure 3 shows.

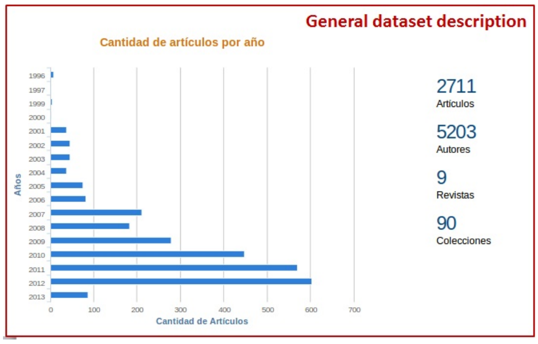

Fig. 3. General dataset description

4.1 Using IA Components

With the purpose of building friendly and intuitive visual interfaces that satisfy basic usability criteria in Web applications, the information has been structured based on interaction patterns common to the Web. These patterns can be easily extended to Linked Data context, and it is here where patterns make it easier for any user to perform search tasks using our proposal. Below are the main tasks to perform for the dataset resources analysis. Also shown is the pattern interaction used to solve this task and the IA component that implements this pattern. The relationship between these three elements is summarized in Table 1. In addition, Figure 4 shows how these IA components have been presented to users in a generated view.

1. **Filter items of interest, and ignore those that are not of interest to the user:** The proposal is to use a faceted navigation interaction pattern. Facets are a component of IA that allow users to filter items in a dataset, avoiding those that are not of interest. Using this strategy the search space is reduced by applying each restriction until the item of interest is retrieved.
2. **Show details of resources of interest:** Once the user has obtained the item or items required through faceted navigation, it is necessary to display more detailed information. This, in the Linked Data context is reduced to a list of the sets of properties and their respective values for each of the elements filtered. In this task a Details on demand interaction pattern is applied and implemented by the IA component dropdown menu. The dropdown menu shows the values and properties of each filtered element.
3. **Contextualizing the navigation space:** When the user navigates the dataset by means of facets or full-text search he/she needs to know in which

part of the search he/she is in each moment. With this purpose the Breadcrumbs pattern interaction is applied, which offers a reference for the user during the navigation process. This way, users can have direct links to previous positions of a transited path. This interaction pattern is implemented by an IA component of the same name.

Table 1. Relationship between Tasks, Interaction patterns and IA Components.

Task	**Pattern Interaction**	**IA Component**
Filter items of interest, and ignore those that are not of interest to the user	faceted navigation	facets
Show details of resources of interest	details on demand	dropdown menu
Contextualizing the navigation space	breadcrumbs	breadcrumbs

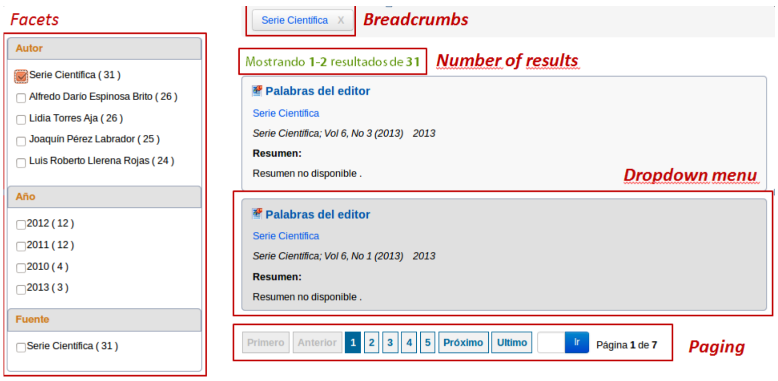

Fig. 4. Use of IA components

4.2 Search Paradigms

For non-tech-users who do not know the structure and form of querying data within a dataset, it is imperative to have mechanisms that allow searching for information without previous knowledge of these two elements. Therefore, it becomes necessary to develop interfaces that support intuitive search. They should also be easy to use and able to satisfy the information needs of the users interacting with them. A solution to this problem would be the inclusion of functionalities based on search paradigms inside user interfaces. In traditional Web conception (Web of documents) search paradigms, are used, whose application has been extended to the context of Linked Data. Existing search paradigms are classified into three categories: Keyword, Iterative-Exploratory, and Natural Language[11].

Keyword Search Paradigms Formulating information needs via keywords, is a paradigm that it has been applied in approaches in the Semantic Web that operate with Linked Data. Many semantic search engines perform search entities, among them: Falcons[11] and Sig.ma[12]. The main advantage of this paradigm is, as it allows user indexing mechanisms, it only needs indexing once, and then the index can be consulted each time a query is made, this streamlines and simplifies the search process.There is no standard syntax to express queries in SPARQL text search (it is only possible to use regular expressions to find matches with strings in RDF graph objects).

However, each triplestore provider provides a specific extension for textual search (Virtuoso , LARQ , Lucene Sail). It is therefore possible to perform hybrid queries combining SPARQL and full text search, these queries yield better results than those obtained by using queries expressed in SPARQL supported by regular expressions to perform textual search on RDF[13]. For a triplestore the proposed approach uses the Openlink Virtuoso server, as it allows text indexes to be created from the RDF graph stored on it. Once an index has been built a new namespace called bif is created. Whereby it can query the index directly using contains predicated in SPARQL hybrid queries. The proposed solution uses the previous development mechanism for implementing the full text search as is shown in Figure 5.

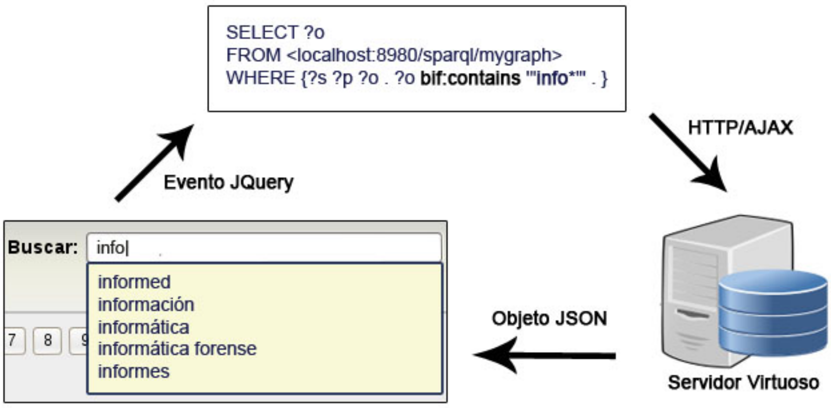

Fig. 5. Full text search: parameterized SPARQL query by authors name

The paradigm of searching through keywords has two fundamental disadvantages: the first one is the possible occurrence of ambiguities in searches (when a word has more than one meaning). The second disadvantage of the key word search paradigm is that it assumes a search scenario where the user actually knows what information they need. So, he requires previous knowledge in the domain of interest in order to search. This can be difficult in most common situations: in the case of scenarios where information needs are not precise or they are vaguely defined (user does not really know, what he should be searching for). In this kind of situation it becomes more useful to use interactive/exploratory paradigm.

Interactive/Exploratory Search Paradigm Iterative / exploratory paradigm, allows an iterative search by exploring and browsing of dataset. Navigation can be made through facets or using graph visualization[10]. Specifically the approach implements faceted navigation.

Faceted navigation is a technique for data structured exploration based on Facet theory[16], this technique allows data sets to be explored through conceptual orthogonal dimensions, also known as facets, which are nothing more than representations of the most relevant resources characteristics.

The main advantage of this technique is that it answers information needs from imprecise or vague search scenarios. In addition, previous knowledge of data schema is unnecessary; because exploratory paradigm is implemented and in this way user information needs are satisfied by exploring the dataset. Another advantage of faceted navigation is that it avoids ambiguity by building structured queries, using the SPARQL query language[17].

The proposed approach implements an interactive/ exploratory paradigm through parameterized SPARQL queries and complex Ajax interactions in Jquery. While the user is adding constraints by selecting facet values, SPARQL queries are automatically generated and they use these values as parameters for building the triple pattern of each SPARQL query. Parameters are sent through asynchronous requests to the server and the answer, received dynamically, updates visual components on the HTML interface. This process flow is detailed in Figure 6.

In contrast with other Linked Data consumption applications such as: DBLP and Rhizomer , our approach offers the possibility to filter results by more than one value in each facet. For instance: If we take the authors facet that lists dataset authors, it is possible to select any number of authors names, then all articles in which the previously selected authors names appear will be retrieved and they will be shown on a HTML interface. It does not even matter if authors in these articles appear as coauthors, they will be retrieved.

5 Evaluation

As explained in the Introduction, the objective of this research is for users to be able to overcome the technical barrier that limits them when consuming bibliographic metadata published as Linked Data available from a SPARQL endpoint service. This technical barrier depends on the complexity involved in Linked Data consumption process, complexity has been defined in different ways

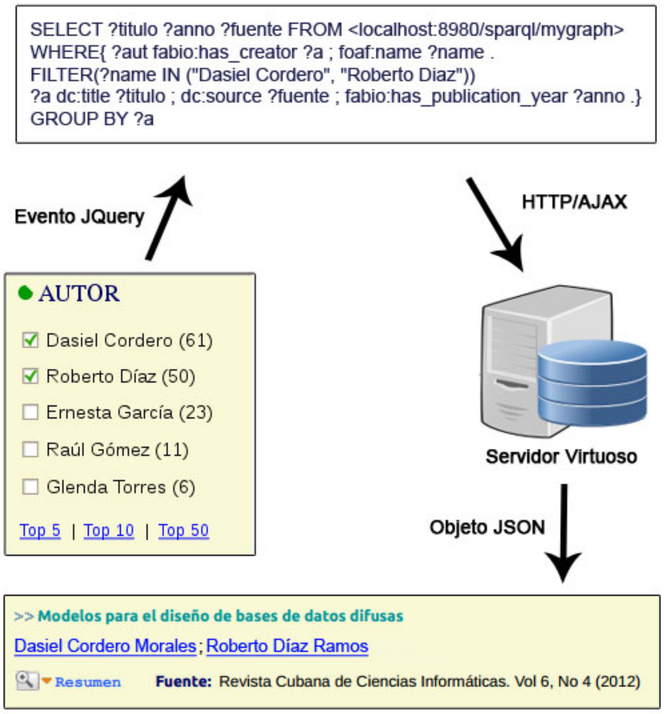

Fig. 6. Faceted search: parameterized SPARQL query by authors name

in several sciences, for instance: In the human computer interaction field it is defined as an aspect of a persons cognitive behavior[18]. This definition has been applied in[19] for analyzing user behavior while they solve a set of specific tasks.

On the other hand, the literature suggests that the complexity of a task has different characteristics such as: number of alternative routes, conflicts between different routes, capacity for analysis, varying time of task execution, among others. As can be seen, therefore, complexity has different meanings and definitions depending on the knowledge area that is being analyzed[15]. So, in the context of the proposed solution, complexity is defined in terms of the usability that it offers. In others words, the better the usability in the proposed approach, the less complex the process of performing search tasks on the bibliographic data published as Linked Data will be.

In order to demonstrate the relationship between usability and the aforementioned complexity indicators , a complexity analysis detailed in[20] has been applied to a set of search tasks, while a sample of non-tech-users are solving these tasks. First the tasks are executed, using the DBLP tool, and then, the same tasks are repeated, using the proposed solution. Both results are compared. DBLP has been chosen for the comparison because not only is it a Linked Data consumption application in bibliographic data domain, it also implements both paradigms: key word (full text search) and interactive/exploratory (faceted search) in an analogy with the proposed approach.

5.1 Phases of Complexity Analysis

1. Defining user roles
2. Defining targeted user tasks
3. Rating the complexity of each step in the targeted users tasks
4. Calculating complexity metrics for the targeted user task

Defining user roles Since usability is not absolute (a task can be simple to perform on software for one kind of user, however, the same task can be complex for another). So, it is necessary to define what role each user who will be executing the task will have. In this context, the user who runs the task will be a non-technical user.

Defining targeted user tasks **Task 1:** Select the X author that has more publications with Y author in year Z and in W journal.
Steps of task: The steps of the task define the simplest way for the user to move through the software application to meet his goal. Below the steps of Task 1 are shown.

1. Finding Y author
2. Getting the Xn author who has written with Y author
3. Finding Xi author that has written most articles with Y author
4. Checking whether Xi author has articles published in Z year
5. Checking whether Xi author has articles published in W journal

Rating the complexity of each step in the targeted user tasks Once the targeted user tasks have been defined then the next phase involves rating the complexity of each step in these tasks based on a set of complexity dimensions presented in Table 2. Each of these dimensions captures a potential source of complexity experienced by the user while interacting with software to perform a step in a user task.

Each of the complexity dimensions is associated with a rating scale that defines objective requirements for each point on the scale. For example, Table 3 provides the rating scale for the usability dimensions presented above.

Once the scale level for each complexity dimension has been defined, it is assigned an index (a value between 1 and 3) of complexity to each step of the task according the level to be assigned to each dimension. Table 4 and Table 5 show the complexity indexes for both DBLP and proposal approach.

Table 2. Complexity dimensions, meaning and measurement

Attribute	**Meaning**	**Measurement**
Ease of learning	It is a measure of how quickly and easily, can a user start to perform each step of a task using the software for first time	For how long a new user needs to use the software, before he can achieve the efficiency level of an expert user in accomplishing a task
Remember in time	It is the capacity of the software to permit the user to always perform the steps of a task, without having to consider how each step should be done.	Required time for finishing the task
Efficiency in use	User productivity using the software	Number of tasks by time unit that the user is able to do
Error rate	Errors detected in task accomplishment	Number of errors detected by user when trying to perform a specific task and the measure in which he recovers from the error

Table 3. Rating scale for usability dimensions

	Level constrains		
Dimensions	**Level 1**	**Level 2**	**Level 3**
Ease of learning	The step does not require previous orientation	The step requires partial orientation	The step requires detailed orientation
Remember in time	The step does not require having to remember the transited path	The step requires having to remember several parts of the process	The step requires remembering the entire process
Efficiency in use	The step requires at least a second to complete it	The step requires at least two seconds to complete it	The step requires at least 3 seconds or more to be completed
Error rate	The performed step does not create any errors	The performed step creates 2 errors maximum	The performed step generates 3 errors or more

6 Calculating complexity metrics for the targeted user tasks

In order to calculate the complexity metric of the whole user task, it is necessary to assign a value of between 1 and 6 for each complexity level to each one of

Table 4. Complexity indexes for each step in complexity dimensions of task using DBLP tool

	Index			
Task steps	**Ease of learning**	**Remember in time**	**Efficiency in use**	**Error rate**
1. Finding Y author	1	1	2	1
2. Getting the Xn author who has written with Y author	2	3	3	1
3. Finding Xi author who has written most articles with Y author	2	2	3	1
4. Checking whether Xi author has articles published in Z year	1	1	1	1
5. Checking whether Xi author has articles published in W journal	1	1	2	1

Table 5. Complexity indexes for each step in complexity dimensions of task using proposal approach

	Index			
Task steps	**Ease of learning**	**Remember in time**	**Efficiency in use**	**Error rate**
1. Finding Y author	1	1	1	1
2. Getting the Xn author who has written with Y author	1	2	1	1
3. Finding Xi author who has written most articles with Y author	2	2	3	1
4. Checking whether Xi author has articles published in Z year	1	1	1	1
5. Checking whether Xi author has articles published in W journal	1	1	2	1

the dimensions. It must be according to the influence or importance that each dimension has in accomplishing each step and according to the whole complexity of tasks. See Table 6. The results of the complexity metric values for Task 1 for both tools, by usability dimensions are shown in Figure 7.

Table 6. Complexity metric value for each complexity dimension

	Complexity metric		
Dimensions	**Level 1**	**Level 2**	**Level 3**
Ease of learning	1	4	6
Remember in time	1	2	5
Efficiency in use	1	3	6
Error rate	1	2	5

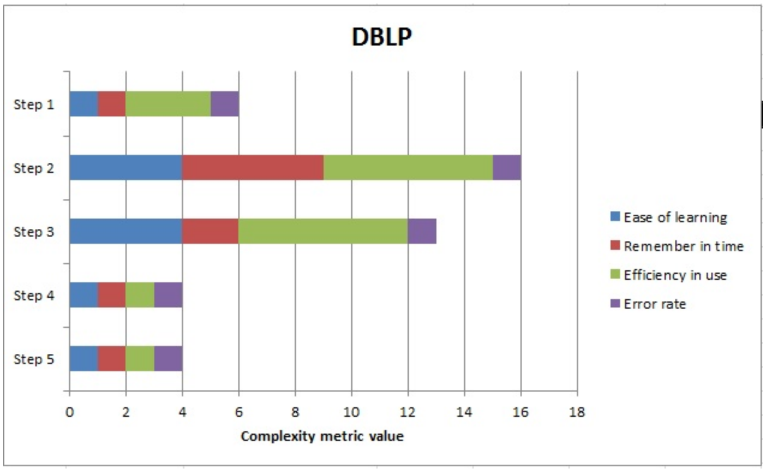

Fig. 7. Complexity metric results for task 1 using DBLP

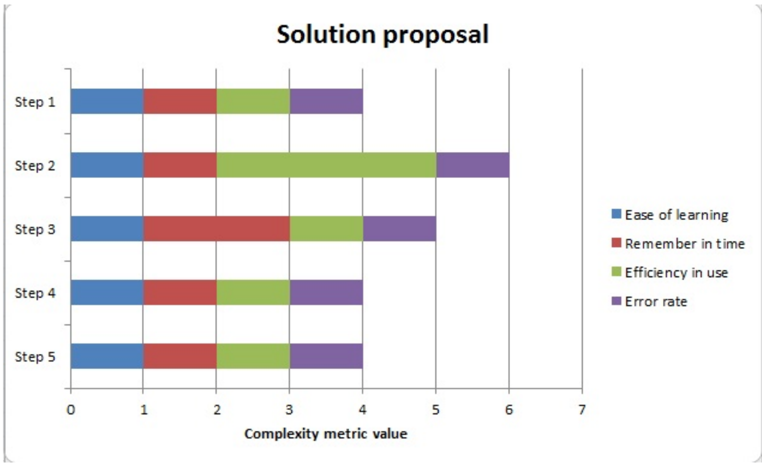

Fig. 8. Complexity metric results for task 1 using proposal approach

6.1 Results: interpretation of the complexity analysis

As Figure 6 and Figure 7 show, executing task 1 using the DBLP tool has a complexity metric of 43 (adding up the complexity of each step that task 1 includes). However, the complexity metric value for the same task using the proposed approach is 23, so there is a reduction by 20 units of complexity metric for task 1 using our approach. This represents a 46 per cent of complexity metric reduction for task 1 in contrast with the complexity metric for the DBLP tool executing the same task. The comparison of complexity metric values obtained after to have carried out 4 search tasks using both DBLP tool and the proposed approach is shown in Table 7.

Table 7. Complexity metric values for each performed task using both DBLP and the approach

	Complexity metric values	
Tasks	**DBLP**	**Approach**
T1	43	23
T2	17	10
T3	14	10
T4	23	13

Taking previous results into account it can be concluded that the proposed approach enables a reduction in complexity in search task execution on bibliographic data performed by non-tech-users in contrast with the complexity that these kinds of users face when they use the DBLP tool.

Therefore the complexity reduction implies an improvement in software usability, which allows users to use it, even those who are non-tech-users. This way the technical barrier that has, until now, prevented certain users from consuming bibliographic data published as Linked Data, has been overcome.

7 Conclusions and Future Work

The literature review has allowed us to see that Linked Data consumption applications are divided in two great groups: generic applications and specific domain applications. Specific domain applications existing at present have limitations such as absence of a general characterization of dataset and usability issues (intuitiveness, ease of learning and memory in time). Unlike other specific domain applications for Linked Data consumption, the proposed approach developed allows a general description of datasets. It also includes IA components and search paradigms to improve the overall experience of non-tech-users in Linked Data consumption. Moreover, it has been demonstrated that complexity is reduced by using this approach; it indicates an improvement in usability. Therefore the user is subject to a lower cognitive load, encouraging ease of learning and memory in time. In addition, the approach allows users to reach their goals in search processes in a faster and easier way. However there are challenges which we need keeping working on, especially in the case of the facets generated by the solution proposed. These facets depend on Ontologies used for describing the bibliographic data of dataset. Therefore if new Ontologies need to be added in order to describe new classes in data, it also becomes necessary to redefine the SPARQL queries that retrieve the properties and resources from the RDF graph stored in the server. Consequently we strongly recommended an implementation of a mechanism that will allow automatic updating of the SPARQL query patterns from Ontologies that describe the resources inside the dataset.

References

1. Hidalgo-Delgado, Y., Rodríguez-Puentes, R.: La web semántica: Una breve revisión. Revista Cubana de Ciencias Informáticas **7**(1) (2013) 75–84
2. Dadzie, A.S., Rowe, M.: Approaches to visualising linked data: A survey. Semantic Web **2**(2) (2011) 89–124
3. Sudeepthi, G., Anuradha, G., Babu, M.P.: A survey on semantic web search engine. IJCSI International Journal of Computer Science Issues **9**(2) (2012) 1–5
4. Heath, T.: How will we interact with the web of data? Journal IEEE Internet Computing **12**(5) (2008) 81–85
5. Heath, T., Bizer, C.: Linked Data Evolving the Web into a Global Data Space. 1st edition edn. Morgan & Claypool Publishers (2011)
6. Staab, S., Studer, R.: Handbook on Ontologies. 2nd edition edn. Springer-Verlang, Berlin Heidelberg (2009)
7. Klyne, G., Carroll, J.J.: Resource description framework (RDF): Concepts and abstract syntax - w3c recommendation (2004)
8. PrudHommeaux, E.: SPARQL query language for RDF (2008)
9. Humfrey, N.: EasyRDF: A PHP library designed to make it easy to consume and produce RDF. In: Proceedings of Web Science Conference 2012. (2012)
10. Mika, P., Tran, T.: Semantic search - systems, concepts, methods and the communities behind it. In: Proceedings of Web Science Conference 2012, Evanston, USA (2012) 121
11. Cheng, G., Qu, Y.: Searching linked objects with falcons: Approach, implementation and evaluation. International Journal on Semantic Web and Information Systems (IJSWIS) **5**(3) (2009) 49–70
12. Catasta, M., Tummarello, G.: Sig.ma: Live views on the web of data. Journal of Web Semantics: Science, Services and Agents on the World Wide Web **8**(4) (2010) 355–364
13. Maali, F., Cyganiak, R., Peristeras, V.: Re-using cool URIs: Entity reconciliation against LOD hubs. In: Proceedings of the Linked Data on the Web Workshop. (2011)
14. Nowack, B.: Paggr: Linked data widgets and dashboards. Journal of Web Semantics: Science, Services and Agents on the World Wide Web **7**(4) (2009) 272–277
15. Campell, D.J.: Task complexity: A review and analysis. Academy of Management Review **13**(1) (1988) 40–52
16. Ranganathan, S.R.: Elements of library classification. 3rd edition edn. Asia Publishing House, Bombay (1962)

17. Oren, E., Delbru, R., Decker, S.: Extending faceted navigation for RDF data. In: The Semantic Web - ISWC 2006. Volume 4273 of Lecture Notes in Computer Science. 1st edn. Springer Berlin Heidelberg, Athens (2006) 559–572
18. Pervin, L.A.: Personality: Theory and research. John Wiley & Sons (1993)
19. Rauterberg, M.: How to measure cognitive complexity in human-computer interaction. Cybernetics and Systems Research (1996) 815–820
20. Sobiesiak, R., Diao, Y.: Quantifying software usability through complexity analysis. IBM Design: Papers and Presentations (2010)