A Conceptual Framework for Linked Data Exploration

Alessandro Bozzon, Marco Brambilla, Emanuele Della Valle, Piero Fraternali, and Chiara Pasini

Politecnico di Milano, Dipartimento di Elettronica e Informazione
P.za L. Da Vinci, 32. I-20133 Milano - Italy
{name.surname}@polimi.it

Abstract. An increasing number of open data sets is becoming available on the Web as Linked Data (LD), many efforts has been devoted to show the potential of LD applications from the technical point of view. However, less attention has been paid to the analysis of the information seeking requirements from the user point of view. In this paper we examine the Information Seeking Process and we propose a general framework that address all its requirements in the context of LD-based applications. We support seamless integration of both Linked and non-Linked data sources and we allow designers to define complex, rank-aware result construction and exploration rules based on rank aggregation and multiple many-to-many data navigation.

1 Introduction

An increasing number of data sets is becoming available on the Web. In this trend, Linked Data (LD) plays a central role thanks to initiatives such as the W3C Linked Open Data (LOD) community project[1] that are fostering LD best practice adoption.

With the growth of the available corpus of Web data, the need arises for effective mechanisms targeted to human users for searching, exploring, and consuming such data. The Semantic Web Community has largely investigated this need from a technical point of view, but limited effort was devoted to considering the full set of requirements of an *Information Seeking Process* (ISP)[15,14], which classifies the activities performed by search users into a well defined set of information seeking stages, i.e. initialization, selection, exploration, formulation, collection, and presentation. Indeed, whilst LD is intrinsically well shaped for coping with information exploration and navigation, no existing works try to apply the full extent of the ISP requirements to the LD domain.

In this paper, we propose a conceptual framework that covers all the stages and requirements of the ISP in the LD setting. The approach is general enough to cover exploration of any kind of source, including deep web sources, search engines, LD sources, and proprietary repositories.

[1] `http://esw.w3.org/SweoIG/TaskForces/CommunityProjects/LinkingOpenData/`

Our approach introduces three additional innovative contributions with respect to the existing solutions: 1) Seamless integration of *both Linked and non-Linked data sources*. Exploration paths upon data sources can be defined using both exact matching on IRIs within LD boundaries, and exact or approximate matching on literals within LD boundaries and on generic data types outside the LD cloud; 2) *Multiple many-to-many navigation* of data. Results can be built in a structured way by combining several concepts and allowing navigation of multiple many-to-many associations at a time; 3) Support of complex *rank-aware result construction and exploration*. Composite results can take into account ranking and/or ordering of result parts coming from different sources, by supporting aggregated ranking functions.

The rest of the paper is organized as follows: Section 2 compares our work to other proposals; Section 3 presents a running example; Section 4 describes our general-purpose approach; Section 4.3 discusses the two application scenarios of free data exploration and vertical application in the context of the running example; Section 5 describes our implementation experience; and Section 6 concludes.

2 Related Work

In the past decades, several works proposed model for the characterization of the information seeking process [15,14]. In [14] the ISP model is characterized by 6 information seeking stages: 1) *Initialization* recognizes the information need and marks introduction of a problem; 2) *Selection* aims at identifying the general area for investigation; 3) *Exploration* is about extending the user understanding of the field and to relate it with what is already known; 4) *Formulation* makes a focused perspective on the topic emerge; 5) *Collection* gathers information and allows one to select interesting findings and to export them; and 6) *Presentation* appropriately renders the collected information.

In the context of Linked Data, we can identify three classes of applications: *data exploration*, *concept exploration* and *vertical applications*. Applications for *Data Exploration* typically leverage the graph representation of data to allow schema-free navigation of the existing information, and they typically apply link-traversing strategies, with no support for search, filtering, data re-shaping or alternative visualizations. Known example of *Data Exploration* are Linked Data browsers like Tabulator [3] and Marble [2].

Concept Exploration applications aim at allowing users to explore a dataset through concepts, their properties, and their relationships, by means of SPARQL endpoints and rich visual interfaces. They also provide data analytics, aggregation functionalities, basic visualization, filtering, faceted search, and pivoting. Parallax [12] and gFacet [11] are excellent solutions that cover formulation, collection and presentation processes of ISP. Parallax also provides support for expansion with related topics, where the available relationships are the ones pre-defined in the underlying collection. The selection and formulation stages are well covered by tools like Sig.ma [17], which provides end-users with search tools for Linked Data. A key distinguish feature of Sig.ma is the incremental display of data while relevant sources are discovered, which enhances the user experience by highlight data provenance. Explorator [8] provides extensive functionalities for the visually-aided composition of queries, and result filtering. Other noteworthy examples of visual exploration of linked data are DERI Pipes [16], RKB Explorer [9] and VisiNav [10].

Other solutions, like Microsoft Pivot [1], have been adapted to Linked Data browsing too [18], thus letting the user explore results by zooming, panning, or pivoting. Our proposal is located in the same application space as the above mentioned tools, sharing several key features such as native support for incremental exploration, data visualization, data relationships highlighting and navigation. However, only RKB Explorer provides support for the initialization and selection steps, whereby only Explorator, Pivot, VisiNav support pivoting.

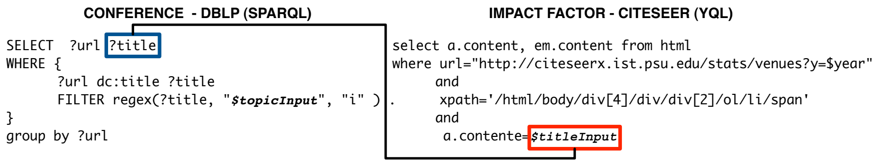
A richer support to ISP is available in *vertical* LD-based applications, which exploit a well-defined and constrained set of classes and relationships in a schema to provide a predefined data exploration and rich interaction experience that usually comprises: customized interaction features, integration with non LD, and simultaneous visualization of result sets produced by several queries, along with their relationships. The BBC's Semantic Music Project [13][2] is an example of *domain specific* LD based application. Although this kind of LD-based applications are targeted to a specific domain of interest, their development would have benefited from a general purpose approach like the one proposed in this paper.

3 Running Example

To provide a better understanding of our approach, the rest of the paper will be discussed upon a running example targeted to the exploration of scientific publications. This is a non trivial domain where multiple class of applications can be built on the same data — e.g., recruiting of researchers, assessment of the quality level of a scientific venue, search for bibliographic references on a topic, etc. The required information is not available in a single repository, thus requiring the integration of different data sources. Some of these sources expose LD, while others expose semi-structured information. A sub-set of authors, papers and conferences are available, for instance, on DBLP (and thus amenable for queries through SPARQL end-points). Papers' citations and conferences' impact factors can be retrieved through page scraping[3] from a Web application like CiteSeer. Data about European research projects are available in Cordis database which is exposed as a SPARQL end-point.

[2] `http://www.bbc.co.uk/music/artists`

[3] We exploit YQL (Yahoo Query Language - `http://developer.yahoo.com/yql/`) as a middleware infrastructure for page scraping.



```
SELECT  ?url ?title
WHERE {
      ?url dc:title ?title
      FILTER regex(?title, "$topicInput", "i" ) .
}
group by ?url
```

```
select a.content, em.content from html
where url="http://citeseerx.ist.psu.edu/stats/venues?y=$year"
     and
      xpath='/html/body/div[4]/div/div[2]/ol/li/span'
     and
      a.contente=$titleInput
```

Fig. 1. An example of SPARQL and YQL queries involved in a *pipe* join

4 Our Approach

Our framework consists of a two-phase application life-cycle, comprising *configuration*, which is oriented to the application specification, and *consumption*, which is oriented toward the actual exploration of data sources within the boundaries of the existing data relationships and/or within the navigation paths defined at configuration time, possibly using the defined data visualization paradigms and interaction mechanisms.

4.1 Application Configuration

In the configuration phase, a domain expert (or a technical stakeholder of the application), knowledgeable about the information need and the relevant data sources, defines some aspects of the application behavior.

In the most general case, the configuration phase requires the definition of a comprehensive *exploration template*, an abstract application model that defines the set of *data sources* consumed by the application, the set of *exploration targets*, each one representing a distinct concept in the data sources (either defined intensionally —in terms of constraints on properties — or extensionally, through enumeration of concepts), a set of *input, output* and *ranking properties* for each *exploration target* (where *rankings* denote the function that impose an order on the retrieved data), and the set of *relationships* on pairs of exploration targets, based on predicates upon *input and output properties*; such predicates can traverse existing relationships among concepts, or be applied on value-based property matching (e.g., exact matching, string similarity, spatial approximate matching, temporal approximate matching, and so on). Custom relationships are calculated at run-time by means of orchestration of queries over distributed (Linked and not-Linked) data sources. The actual sequence of queries depends on the exploration pattern, on the access restrictions imposed by the data sources, on the input-output parameters dependencies existing between the involved exploration targets (some output parameters of the second exploration target are matched to some input parameters of the first one), and on the join selectivity statistics that can be extracted from the actual joined data. The orchestration may comprise two join operators over the results of queries: *parallel* join and *pipe* join.

Figure 1 shows an example of *pipe* join that involves a Linked and a not-Linked data source, to associate all the *conferences* about topic given as input with their impact factor: the *?title* output value of each instance result from the leftmost query is provided as input to the rightmost query (*$titleInput*).

To enable advanced visualization and interaction features, the template can provide the *domain and range* values for the output properties to visualize, and the list of advanced *interaction mechanisms* allowed for the application; for instance *grouping* and *clustering* criteria that can be applied on the extracted result set to help users exploring the retrieved data.

We stress that all the above configurations can be manually or automatically defined. Our approach makes no assumption on the actual origin of the application configurations. In case of manual configuration, the configuration phase requires higher domain and technical expertise than the consumption phase (because the user must be aware of the data sources, the meaning of the exploration options, and the presentation options). However, we point out that the configuration step can be performed in a declarative, visually-aided way.

In general, orchestration and query generation exploit parametric query templates in the native query language supported by the data source. In case of non linked data, the query template must be specified by the registrar when declaring the data source in the repository.

4.2 Application Consumption

The *application consumption phase* is based on the approach presented in [5], which proposes a conceptualization of exploration primitives for addressing "search as a process".

Upon query submission, the system invokes the involved data sources, producing a result set of joined results (ranked according to a given function) which defines the initial user concepts space, shown to the user according to her data visualization choices (e.g., tabular representation, geographic map, charts, etc.). The result set conforms to an evolving result schema, which specifies projected attributes, ranking attributes, and allowed relationships. A set of interaction primitives enable manipulation, exploration and expansion of search results, thus allowing for continuous evolution of the query and of the result set itself. For instance, the system presents a first batch of results, and users browse them; if users are not satisfied, a *more* operation calculates additional results (within the currently defined concepts) by fetching new data and combining them, also with the previous results. The *re-rank* operation re-orders the result set according to a different ranking function; *filtering*, *grouping* and *clustering* allow the user to re-shape the current information space to better suite its view point on the retrieved data. At this point, the user can select the most relevant result instances and continue with the exploration by executing an *Expand* operation that traverse one of the available *relationships* of the selected results

After that, the user submits another object query, possibly by providing additional selection criteria; the system will then retrieve connected object instances and form a "combination" with the objects retrieved at the preceding steps. At any stage, users can "move forward" in the exploration, by adding a new class to the query, or "move backward" (backtrack), by excluding one of the classes

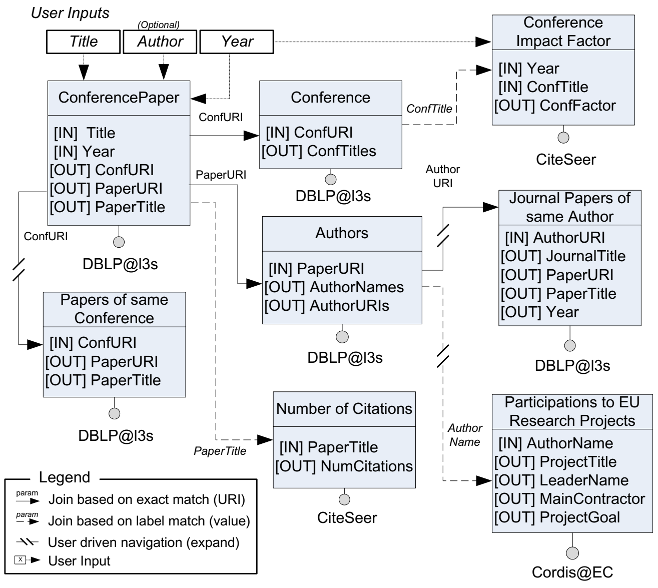


Fig. 2. Visual representation of the exploration template for a *vertical* running example application

from the query, or by "unchecking" some of their previous manual selections of relevant object instances.

4.3 Our Approach at Work on the Running Example

To make our approach more concrete, we describe its application in the context of the running example presented in Section 3.

We design a *vertical* application, where the user may submit a (part of a) conference paper title, the year of publication, and possibly one or more authors; as a response, the systems extracts for him the list of matching papers, the corresponding authors and conferences, the number of citations of each paper, and the impact factor of the conference, ordered according to the importance of the paper, expressed as function of the number of citations and of the conference relevance. To get a better overlook on the scientific relevance of the retrieved publication, the user may then decide to extend the results by navigating toward other papers published in the same conference, the journal papers of each author, and the European research projects the authors have been responsible for. Some of the information required by the application, i.e. the conference impact factor, is not available on the original LD sources, and it therefore needs to be extracted directly from the Web.

Figure 2 depicts an explanatory visual representation of the exploration template for such an application. In the example we assume that the initial results comprise information about *combinations* of: *Conference Paper*, *Conference*, *Authors*, *Conference Impact Factor*, and *Number Of Citations*. Custom

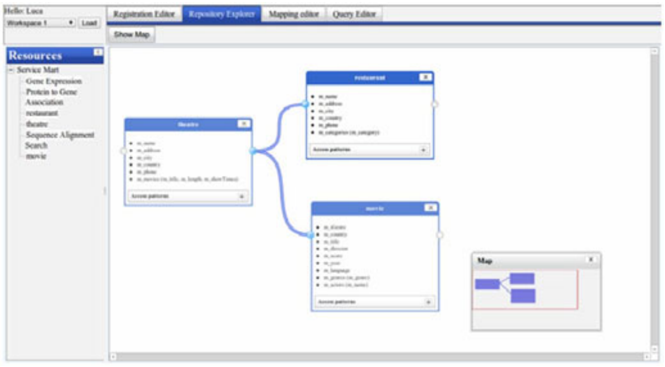


Fig. 3. Overview of the toolsuite for application configuration

relationships include *Participation To EU Projects*, *Papers Of Same Conference*, and *Journal Papers Of Same Author*. Some of these concepts are joined based on the navigation of IRI-based relationships, while others are joined based on literal comparison according to a predicate (e.g., string similarity, spatial distance, or others, based on the attribute type). The result is a set of concept combinations, ordered by the combination ranking function, which, in this case, rewards highly cited papers published in relevant conferences.

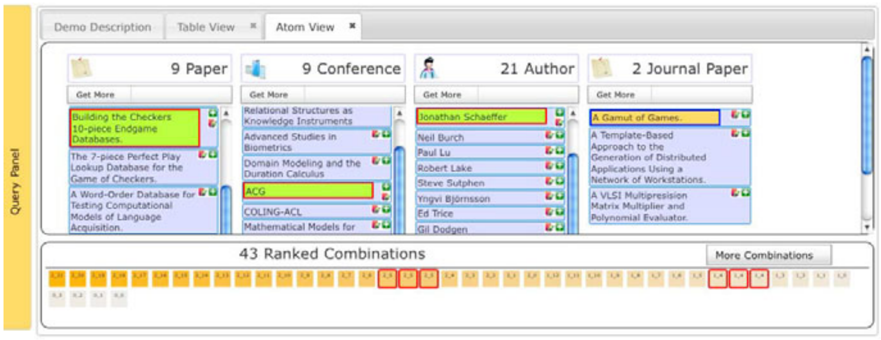
Indeed, the *Initialization* and *Selection* stages of the ISP process are extremely important for such an application, as the initial problem requires the definition of an exploration starting point that comprises a ranked combinations of concepts. The selection can be performed by means of appropriate visual aid tools, like the ones depicted in Figure 3.

The *Exploration* stage is also stressed, as the application includes several navigation steps to expand the retrieved information space. Likewise, the *Formulation* stage is enabled by the availability of filtering and clustering conditions.

Finally, the *Presentation* stage can be enabled by alternative visualizations of the same result set, which can stress the visual properties of the rendered outputs according to the targeted purposes. objects that constitute it are highlighted.

5 Implementation Experience

This section describes the software prototype that implements our framework for LD exploration applications. That showcases our approach by enabling the execution of (1) queries over selected data sources, (2) joins between queries results, and (3) data manipulation primitives for content ranking, visualization and exploration. The prototype is built upon a three-tier, distributed Web architecture (Figure 5), and it features rich and fluid user interactions (as described in Section 4.2) by means of asynchronous server communications and client-side data storage, processing and manipulation. Beside improving the user experience, a rich-client architecture provides, as additional benefit, the implementation of a



Building the Checkers 10-piece Endgame Databases.
The 7-piece Perfect Play Lookup Database for the Game of Checkers.
A Word-Order Database for Testing Computational Models of Language Acquisition.

Relational Structures as Knowledge Instruments
Advanced Studies in Biometrics
Domain Modeling and the Duration Calculus
ACG
COLING-ACL
Mathematical Models for

Jonathan Schaeffer
Neil Burch
Paul Lu
Robert Lake
Steve Sutphen
Yngvi Björnsson
Ed Trice
Gil Dodgen

A Gamut of Games.
A Template-Based Approach to the Generation of Distributed Applications Using a Network of Workstations.
A VLSI Multipresision Matrix Multiplier and Polynomial Evaluator.

Fig. 4. User Interface prototype: set of concept tables (Paper, Conference, and Author tables) and ranked list of combinations (numbered list at the bottom of the screenshot)

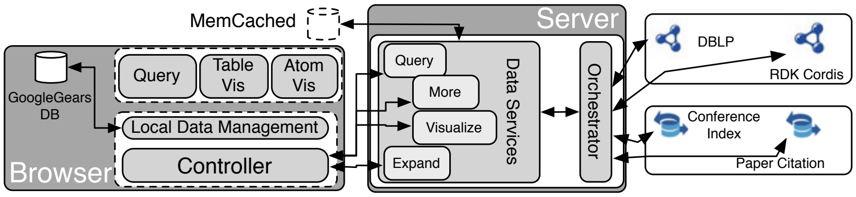


Fig. 5. System architecture overview

light-weight scalable REST server architecture, which naturally supports node distribution and replication.

The user interface (Figure 4) is an HTML5 application written according to the Model-View-Controller design pattern and exploiting the libraries *jQuery* and *JavascriptMVC*. The UI has been designed to be dynamically instantiated at run-time, based on the application configuration files.

The server-side is designed as a pluggable orchestration system featuring four kinds of executable nodes: *query nodes*, for query execution, *split* nodes, to trigger the parallel execution of queries, *join* nodes, to perform join operations over query results, and *data transformation* nodes, devoted to the manipulation of (joined) query results. Query and data transformation nodes provide standard interfaces for configuration management, invocation and result manipulation; the definition of the business logic required to interact with custom data sources is therefore left to the application configuration, while the framework is responsible to grant its correct execution. To reduce latency time, we adopted a distributed memory object caching system (*MemCached*) to store information about user interactions and to hold the results of query and join executions.

The server-side supports query formulation, evolution and storage. Each query is uniquely identified in the orchestrator and can be re-executed at any moment. Each result set produced by a query execution is also identifiable and retrievable. A prototype and a video based on the running example are published at: http://www.search-computing.org/demo/ui. The system is now undergoing a major implementation effort, and alternative application scenario has been recently demonstrated [6,4].

6 Conclusions

In this paper we presented a conceptual framework for the exploration of Linked (and non-Linked) Data that cover all the phases of the ISP process. The proposed join-based approach for the creation of custom relationships saves the user several exploratory link navigations between concepts and our tunable global ranking function provides a customizable ranking of combinations of objects. Furthermore, in our work exploration is not confined to data aggregated in one repository, but, thanks to value-based joins, can span linked data and arbitrary data sources wrapped as Web services.

Acknowledgements. This research is part of the Search Computing (SeCo) project[7], funded by the ERC IDEAS program (www.search-computing.org).

References

1. Microsoft pivot: `http://getpivot.com/`
2. Becker, C., Bizer, C.: Workshop about Linked Data on the Web (LDOW2008). A location-enabled linked data browser. In: Procedings of the 1st Workshop about Linked Data on the Web, LDOW 2008 (2008)
3. Berners-lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: Exploring and analyzing linked data on the semantic web. In: 3rd Int.l Semantic Web User Interaction Ws, SWUI 2006 (2006)
4. Bozzon, A., Braga, D., Brambilla, M., Ceri, S., Corcoglioniti, F., Fraternali, P., Vadacca, S.: Search computing: multi-domain search on ranked data. In: SIGMOD Conference, pp. 1267–1270 (2011)
5. Bozzon, A., Brambilla, M., Ceri, S., Fraternali, P.: Liquid Query: Multi-domain Exploratory Search on the Web. In: WWW 2010: 19th International Conference on World Wide Web, pp. 161–170. ACM Press, New York (2010)
6. Bozzon, A., Brambilla, M., Ceri, S., Fraternali, P., Vadacca, S.: Exploratory search in multi-domain information spaces with liquid query. In: WWW (Companion Volume), pp. 189–192 (2011)
7. Ceri, S., Brambilla, M. (eds.): Search Computing. LNCS, vol. 5950. Springer, Heidelberg (2010)
8. de Araújo, S.F.C., Schwabe, D.: Explorator: a tool for exploring rdf data through direct manipulation. In: LDOW (2009)
9. Glaser, H., Millard, I., Jaffri, A.: RKBExplorer.com: A Knowledge Driven Infrastructure for Linked Data Providers. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 797–801. Springer, Heidelberg (2008)
10. Harth, A.: VisiNav: Visual Web Data Search and Navigation. In: Bhowmick, S.S., Küng, J., Wagner, R. (eds.) DEXA 2009. LNCS, vol. 5690, pp. 214–228. Springer, Heidelberg (2009)

11. Heim, P., Ertl, T., Ziegler, J.: Facet Graphs: Complex Semantic Querying Made Easy. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 288–302. Springer, Heidelberg (2010)
12. Huynh, D.F., Karger, D.R.: Parallax and companion: Set-based browsing for the data web. Technical report, Metaweb Technologies Inc. (2009)
13. Kobilarov, G., Scott, T., Raimond, Y., Oliver, S., Sizemore, C., Smethurst, M., Bizer, C., Lee, R.: Media Meets Semantic Web – How the BBC Uses DBpedia and Linked Data to Make Connections. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) ESWC 2009. LNCS, vol. 5554, pp. 723–737. Springer, Heidelberg (2009)
14. Kuhlthau, C.C.: Inside the search process: Information seeking from the user's perspective. Journal of the American Society for Information Science 42(5)(5), 361–371 (1991)
15. Marchionini, G.: Exploratory search: from finding to understanding. Commun. ACM 49(4), 41–46 (2006)
16. Phuoc, D.L., Polleres, A., Hauswirth, M., Tummarello, G., Morbidoni, C.: Rapid prototyping of semantic mash-ups through semantic web pipes. In: WWW, pp. 581–590 (2009)
17. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig.ma: live views on the web of data. In: WWW 2010: 19th International Conference on World Wide Web, pp. 1301–1304. ACM, New York (2010)
18. Workbench, I.: `http://iwb.fluidops.com/pivot`