FaRBIE: A Faceted Reactive Browsing Interface for Multi RDF Knowledge Graph Exploration

Luis Fuenmayor[1], Diego Collarana[2,3], Steffen Lohmman[3], Sören Auer[4]

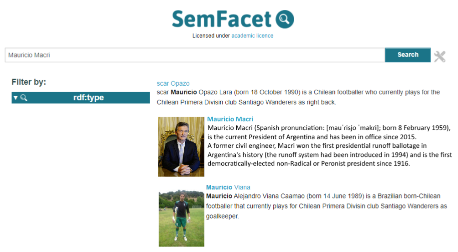
[1] RWTH Aachen University, Aachen, Germany
[2] University of Bonn, Bonn, Germany
[3] Fraunhofer IAIS, St. Augustin, Germany
[4] Technische Informationsbibliothek, Hannover, Germany
luis.fuenmayor@rwth-aachen.de
collaran@cs.uni-bonn.de
steffen.lohmann@iais.fraunhofer.de
soeren.auer@tib.eu

Abstract Linked Data brings numerous RDF Knowledge Graphs to the Web, and as a result, the Linking Open Data (LOD) cloud comprises several independent – but linked – graph datasets. Faceted Browsing is a popular type of User Interface (UI) to explore the knowledge in these RDF graphs. Due to the distributed and linked nature of RDF graphs, exploring more than one graph at a time is a common interaction scenario in this context. However, most state-of-the-art user interfaces allow users to browse only one RDF graph at a time. Additionally, exploring multiple RDF graphs adds a degree of uncertainty to the user interface, such as connectivity problems and longer query response times as well as variations in the size and semantic complexity of the retrieved data. In this paper, we present a faceted browsing approach named *FaRBIE*, which enables users to explore multiple RDF knowledge graphs simultaneously. The reactive UI approach implemented in *FaRBIE* aims to 'react' to the uncertainty imposed by querying multiple RDF graphs in order to enhance the user experience. *FaRBIE* is composed of reactive UI components, refined from common faceted browsing UIs and improved by means of reactive design patterns. A component-based UI approach supports the convenient generation and reuse of the reactive elements.

1 Introduction

In the current age of Internet, data is being created at an explosive rate. However, beyond simple data growth, it becomes more and more important to understand *how* such data correlates. Consequently, knowledge graphs gain importance, as does the field of Linked Data, which brings a increasing number of RDF Graphs to the Web. As a result, the Linked Open Data (LOD) cloud is comprised of several independent – but linked – graph datasets. User Interfaces (UIs) oriented towards the exploration of RDF graphs become equally significant, as a means of bridging the gap between the technology and non-technical users. Faceted Browsing is a popular User Interface (UI) paradigm to explore and navigate the knowledge contained in RDF graphs.

Due to the distributed nature of Linked Data, exploring more than one RDF graph at a time is a common interaction scenario. However, this common scenario lacks tailored

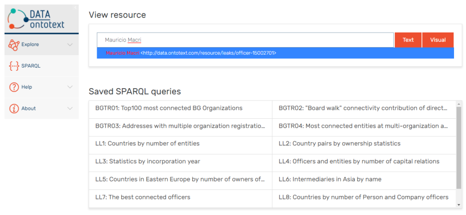


scar Opazo
scar **Mauricio** Opazo Lara (born 18 October 1990) is a Chilean footballer who currently plays for the Chilean Primera Divisin club Santiago Wanderers as right back.

Mauricio Macri
Mauricio Macri (Spanish pronunciation: [mauˈrisjo ˈmakri]; born 8 February 1959), is the current President of Argentina and has been in office since 2015.
A former civil engineer, Macri won the first presidential runoff ballotage in Argentina's history (the runoff system had been introduced in 1994) and is the first democratically-elected non-Radical or Peronist president since 1916.

Mauricio Viana
Mauricio Alejandro Viana Caamao (born 14 June 1989) is a Brazilian born-Chilean footballer that currently plays for Chilean Primera Divisin club Santiago Wanderers as goalkeeper.

(a) Exploring DBpedia using SemFacet



Mauricio Macri <http://data.ontotext.com/resource/leaks/officer-15002701>

BGTR01: Top100 most connected BG Organizations	BGTR02: "Board walk" connectivity contribution of direct...
BGTR03: Addresses with multiple organization registratio...	BGTR04: Most connected entities at multi-organization a...
LL1: Countries by number of entities	LL2: Country pairs by ownership statistics
LL3: Statistics by incorporation year	LL4: Officers and entities by number of capital relations
LL5: Countries in Eastern Europe by number of owners of...	LL6: Intermediaries in Asia by name
LL7: The best connected officers	LL8: Countries by number of Person and Company officers

(b) Exploring Linked Leaks using OntoText's UI

Figure 1: **Motivating Example.** A user typically requires two different UIs to explore the RDF graphs of DBpedia and Linked Leaks.

and efficient UI support at the moment. Let us assume the following *multi RDF* and *distributed* browsing scenario in the context of a crime investigation: during an ongoing investigation for corruption, browsing and analyzing information coming from various sources is one of the key steps performed by investigators. In the case of a "politically exposed person", such as a politician, an investigator wants to explore whether or not this politician is in any form involved or related to the Panama Papers scandal, and at the same time retrieve additional general information about the politician. While the Linked Leaks Dataset[5] contains an RDF representation of the Panama Papers, DBpedia[6] contains general information about politicians. Figure 1 illustrates how a user typically requires two different UIs for exploring information about *Mauricio Macri* in two RDF graphs. In this case, SemFacet (Figure 1a) is used to browse the RDF graph of DBpedia, while the OntoText browser (Figure 1b) is used to explore the RDF graph of Linked Leaks.

Additionally, the exploration of multiple RDF graphs brings new challenges at the UI level. The UI has to deal with a higher degree of uncertainty, such as connectivity problems and longer query response times. These are issues which are only aggravated by the variations in the size of the retrieved data, and the complexity in the semantics of the data, all factors which potentially have a negative effect on the usability of the interface, ultimately progressing into a decrease of the overall user experience. State-of-the-art approaches [2,4,6,9] are mainly designed to explore one RDF graph at a time. Consequently, they do not address these challenges sufficiently.

In this paper, we focus on tackling the problem of exploring multiple RDF graphs and its challenges, hence conceiving *FaRBIE*, a Faceted Reactive Browsing Interface. *FaRBIE* implements a reactive user interface style with solid User Experience (UX) design principles. In the context of this paper, a *Reactive User Interface* should be understood as an interface that is not only capable of providing an interactive and real-time feedback to the users, but also of adapting the interface in order to cope with the challenges that are manifested in both retrieval times and results complexity from the set of independent and multiple RDF graphs being queried. The main contributions of

[5] http://data.ontotext.com/linkedleaks
[6] http://wiki.dbpedia.org

this paper are as follows: *i)* Definition of the main challenges at the UI level when exploring multiple RDF graphs at the same time. *ii)* A reactive component-based UI approach that handles the uncertainty imposed by the intrinsic nature of RDF graphs. *iii)* A proof of concept and an initial open source implementation using modern Web UI development technologies.

2 Related Work

Faceted browsing on RDF data has been the focus of several research works. However, we believe the problem of easy-to-use interfaces for the exploration of multiple RDF graphs is still underexplored. Additionally, most of the available tools are targeted towards advanced users instead of non-technical lay users. Arenas et al. [2,4] introduced *SemFacet* as a faceted search approach enhanced with Semantic Web technologies. *SemFacet* is capable to automatically generate the facet names and values from metadata provided in RDF and OWL. Additionally, it uses the implicit knowledge of the RDF graph to improve the interface by showing new facets or deriving new annotations. *SemFacet* is based on a theoretical framework that accounts for both RDF data and OWL 2 axioms [1]. However, the scope of *SemFacet* is limited to faceted search on a single dataset.

Stadler et al. [9] presented *Facete*, a spatial data exploration application that can be applied to SPARQL endpoints. The facets are automatically generated and a map view of the spatial data is displayed to the user. The focus of *Facete* is on the exploration of spatial data instead of general entity data, as in our approach. Additionally, *Facete* also considers only one RDF dataset for exploration.

Stolz and Hepp [10] conducted an evaluation of the appropriateness of an adaptive faceted search UI as a search paradigm for e-commerce on the Web of Data. In their work, they present preliminary evidence of the applicability of an adaptive UI for faceted search. However, in contrast to our reactive approach that changes the user interface elements according to the semantics of the new data received as search results, the UI adapts according to a user profile, temporal context, and diversification.

Ferré [5] presented *Sparklis* with the goal to combine the expressivity of formal languages and the usability of faceted search. *Sparklis* enables non-technical users to explore a SPARQL endpoint and answer complex questions based on facets suggested by the application. The facets are complemented by SPARQL queries to retrieve answers to complex questions.

Finally, Khalili et al. [6] presented *LD-R*, a framework to develop reusable Semantic Web driven UIs. Although the focus of the work is to provide a development framework, inside its configuration it incorporates Faceted Browsing. Thus, the approach goes in the same direction that is envisioned for *FaRBIE*.

3 UI Problems when Browsing Multiple RDF graphs

In the following, we will briefly describe what we identified as some of the main UI problems when browsing multiple RDF graphs.

Reactiveness towards the semantics of the data (P1) The quality and amount of the semantics of data in a *multiple knowledge graph exploration* scenario vary in a significant way. We may find general concepts (e.g., Organization) to more specialized concepts (e.g, Terrorist Organization). The UI needs to deal and react to the variety of the semantics in the data, in the sense of preserving the semantics of the information being retrieved. Most of the current implementations [3,9] are designed with single RDF knowledge graph exploration scenarios in mind, where flows of information in the form of entities and attributes that originate from them are static in structure. In such cases, designing interfaces that implement *reactivity* in the semantic context have received more attention of researchers recently. However, this is unfortunately not the case when it comes to the exploration of multiple knowledge graphs. This is due to the fact that the flow of information in such cases is no longer static in structure – it is inherently different, constantly growing and evolving, thus posing a challenge when it comes to designing standardized but scalable interfaces that cope with such change. Hence, we identified a lack of reactive approaches that tackle semantic contexts when it comes to multiple RDF graph exploration. We attempt to tackle this problem by developing reactive UI components, which allow for semantic, multiple RDF knowledge graph exploration while preserving the semantic contextualization of the results being obtained.

Error visibility and feedback (P2) Another potential pitfall that can be observed when *exploring multiple knowledge graphs* is that of error visibility and feedback. Errors during the browsing process need to be made clearly visible so as to provide better feedback to the users, in accordance to the *visibility of the system status* and the *help users recognize, diagnose, and recover from errors* heuristics referred to by Nielsen [8]. At the same time, the attention span time frames mentioned by Nielsen [7] should also be preserved. These usability heuristics help safeguard that interfaces are sufficiently comfortable and functional when the user performs the intended tasks of a system, ensuring further adoption of the technology. However, these principles are difficult to preserve in the context of multiple knowledge graph exploration, due to the nature of how federated searches operate – different queries for different knowledge graph sources, performed in parallel, typically imply different response times for the queries as well. This contributes to longer waiting times, where the user could be left without any feedback for time spans that surpass a few seconds, causing confusion and triggering a sense of lack of control in the users, which ultimately breaks the feedback flow between the interface and the user. In the long run, this has the potential of becoming a fatal flaw that undermines the development of the technology. We attempt to tackle this problem by the implementation of a reactive UI component, which allows for improved visibility and feedback of errors that may happen during the federated search process, where the user can have full feedback as to the current system status, as well as obtaining information about the nature of the errors that have occurred.

Minimalist design (P3) A UI for exploring RDF graphs needs to manage well the topic of *minimalism* and avoid violating the *aesthetics and minimalist design* rules that also form part of Nielsen's heuristics [8]. To address these rules, we took three main requirements into consideration when designing our approach: *maximum screen space* *usability*, *maximum visibility* (to also contribute in solving **P2**), and *minimal cluttering*. However, providing *minimal cluttering* in a screen space where there is both abundant information to display and considerable functionality to support becomes a challenge, especially in systems which have multi-layer navigation bars, as they either consume an important block of screen space or become too confusing or difficult to follow for the user. Thus, these design requirements were also carefully considered when designing our UI components. Moreover, the implementation of reactive features in our design further aims to contribute to minimizing the required screen space, as well as improving the ergonomics of the interface by providing familiarity in the interface by means of semantically enabling or disabling views or functions according to the results being obtained during the search.

4 The FaRBIE Approach

To tackle the challenges of browsing multiple RDF graphs, and to keep up with providing non-technical users with a more enjoyable and usable experience, we propose a *reactive user interface* design style. In contrast to imperative approaches (e.g., libraries such as jQuery), a Reactive UI updates itself by *reacting* to changes in the data and rendering the right components whenever such data changes occur. Reactive UIs are component-oriented, where each component may evolve independently, facilitating reusability in the interface. A reactive UI component may contain not only the *view* but also pieces of *logic* to react appropriately to the semantics of the data. Hence, we argue that this style of UI fits well for RDF-based applications. ReactJS[7] has become one of the most popular libraries to implement this style of user interface.

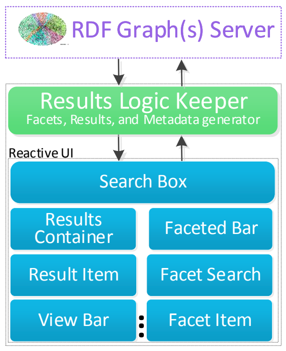
4.1 Architecture

By making the user interface components reactive, we can provide developers with the possibility of making such components interact with the users in real time, while the queries for the multiple datasets are still running. In general, a common faceted browsing UI can be divided into three main UI sections: *i)* the search box or input section and source selection, where the user enters a keyword to start the search process and select data sources and entity types of interest. *ii)* the facets section contains all the UI elements to filter and narrow down the results; the facets are automatically generated based on the results collected from the RDF graphs. *iii)* the results section shows the entities found in the RDF graphs that match the keyword. Additionally, it provides users with feedback concerning the current state of a multiple dataset query (whether the search has succeeded, has failed, or is in progress). Figure 2 illustrates the *FarBIE* architecture.

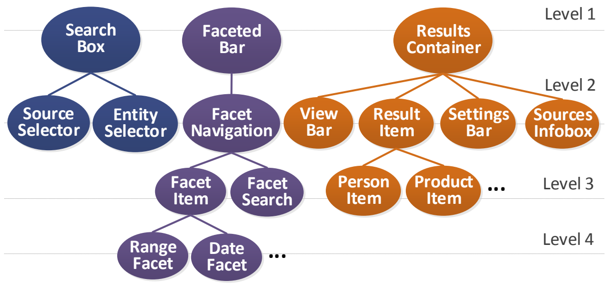
4.2 Logic Keeper

The Logic Keeper is not a user interface component but a component responsible for handling the communication with the RDF graph servers (e.g., using SPARQL HTTP

[7] `https://facebook.github.io/react/`



(a) *FaRBIE* architecture



(b) UI components

Figure 2: **UI Architecture.** (a) *FaRBIE* architecture containing the Results Logic Keeper and Reactive UI Components; (b) Reactive UI Components organized in levels from generic to specific. The UI components can be extended and specialized in providing a better UX according to the semantics of data.

requests). It uses the keyword search input and prepares the queries to the RDF graph servers. In this paper, we assume that the RDF graph server supports Keyword Search[8]. The Logic Keeper manages all the logic applied to the results coming from the RDF graph servers, including 1) facets generation, 2) entity results preparation, and 3) metadata creation from the search process.

To *generate the facets*, we based our implementation on the work of Arenas et al. [3]. In brief, generic SPARQL queries are applied on the resulting entities to generate the facets. Additionally, the list of facets and the number of facet values are computed and maintained by the Logic Keeper. The second responsibility of the Logic Keeper is that of *preparing the entity results*, where a snippet is composed per result, and values such as dates are standardized for the reactive UI components. Finally, *meta-data about the search process is created*, including provenance of results, server request success or error information, the number of results, the number of results by type and the type of entities. All this metadata is computed and managed by the Logic Keeper. To trigger the reaction of the UI components, the Logic Keeper communicates whenever there is a change in the search results data, under the following circumstances: 1) more results arrive from the RDF graphs, resulting in new facets or new result items, or 2) a user interaction in the UI components demands more data.

4.3 Reactive UI components

The following reactive components were designed to tackle the UI challenges mentioned in Section 3. For the sake of terminology, we will be naming entities as *categories* and attributes as *elements* in the following.

[8] Common Triple Stores, such as Virtuoso or Fuseki, usually provide a keyword search functionality `https://jena.apache.org/documentation/query/text-query.html`

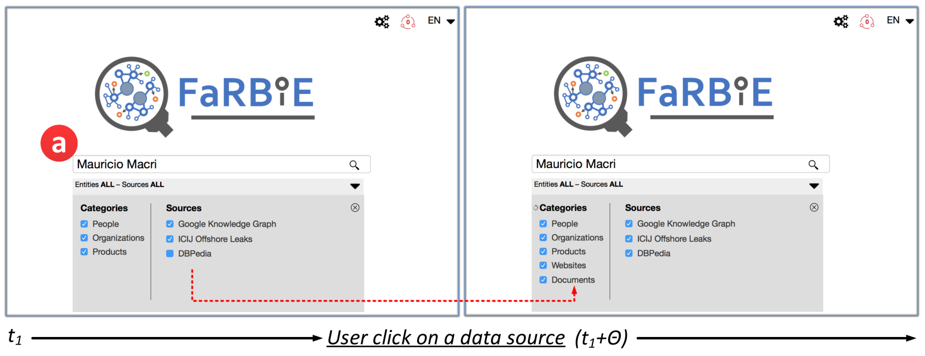


Figure 3: (a) **Search Box** reactive component. A new data source selection produces a reaction in the categories list with the new entities available in the graph

Search Box: The purpose of this UI component is to provide the user with an interface for the input of the query parameters, and thus the creation of the federated query. No particular reactivity needs to be incorporated in this component apart from the common auto-suggestion and auto-completion features well-know for this type of UI component.

Source Selector & Entity Selector: Allows the user to focus on specific RDF graphs or entities. The *reactivity* designed on these components is triggered by a user interaction. When the user filters a data source, the categories list is updated with the searchable entities of that RDF graph.

Faceted Bar: Through the use of a real-time-populated Faceted Bar, we attempt to address the *reactivity* and *visibility* issues in terms of views navigation and results (i.e. data) filtering for a set of RDF knowledge graph datasets. We attempt to achieve this by combining the *Accordion*[9] user interface element and the *List* menu patterns into one unified reactive component, which we call the Faceted Bar. The *Faceted Bar* provides users with a layered menu-styled navigational experience, in order to provide the ability of browsing through entities and thus filtering results as well, in a single component, which could potentially save time and improve the ergonomics.

For the purposes of our design, we support the implementation of this component as a left-side-bar or as a top-bar, according to the developer's preference. The *Faceted Bar* also allows to minimize the impact of the *screen space trade-off*, for the benefit of saving additional screen space without sacrificing visibility in neither the navigation area (i.e., the categories and elements filtering) nor the actual screen space required for the data presentation (i.e., results presentation).

Facet Navigation Menu: The Facet Navigation Menu is a child UI component of the *Faceted Bar* UI component. It allows users to semantically navigate within the elements being obtained as results from the federated search query, through entity categorization. The idea is to provide menus and sub-menus which allow users to narrow

[9] Accordion menu pattern in CSS3.
`https://designmodo.com/css3-accordion-menu/`

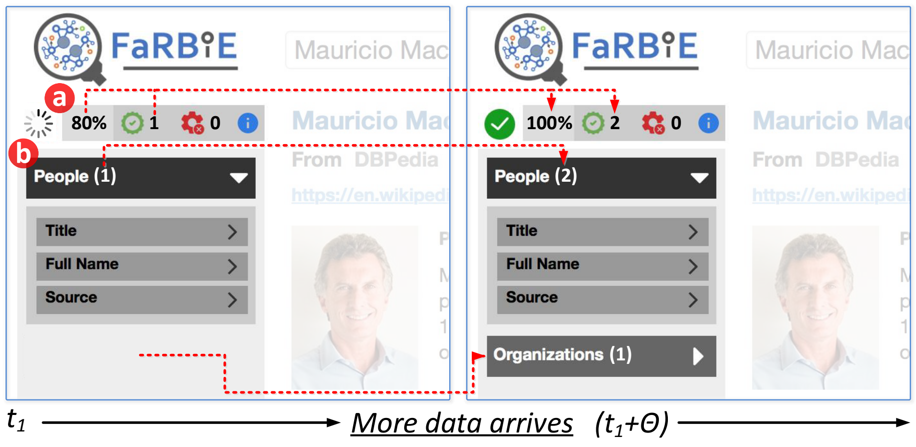


Figure 4: **(a) Source Box** component reacts when a data source query is retrieved successfully or with errors. **(b) Faceted Bar** component reacts to new facet items or categories.

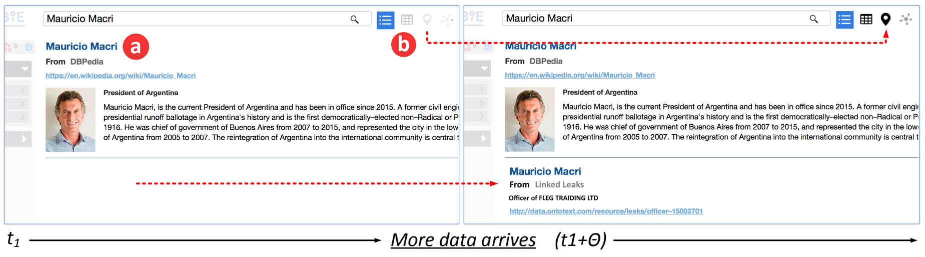
down the list of results, in order to improve usability by means of offering better grouping and increased visibility, while providing a design backbone for future developers at the same time. Our approach supports default entity categorizations using the types *Persons, Organizations, Products*, and *Documents*. For instance, it maps all the elements obtained as a result of the query which belong to the *person* category, thus creating a "menu" container for such attributes. The Facet Navigation Menu UI component holds two child UI components: *Facet Item* and *Facet Search*.

Facet Items: The purpose of the Facet Item UI Component is to serve as the final display and selection place for each category. Thus, elements such as *gender* for a category of type *person*, for example, would be placed as checkbox UI components, which the user could use as selection to further narrow the results view of the query.

Facet Search: The purpose of the Facet Search UI Component is to serve as a means of a refined search among the obtained elements throughout each category. By using such component, we provide users also with the ability of refining the field of options available for navigation of the elements.

Sources Infobox: The purpose of this UI component is to provide a toolbox interface where users can have an overview of the status corresponding to the sources being queried. This component has three elements: 1) successfully retrieved sources, 2) failed sources, and 3) the *information button*, which pop ups a dialog with detailed information regarding the status of the sources, response times, as well as error codes and messages, which clearly indicate the nature of a failure in a queried source.

Results Container: The purpose of this UI component is to encapsulate the main UI components related to the results of the query. This includes the screen area allocated for displaying the list or map of results belonging to a query, in which each result of the query is then displayed in a child *ResultsItem* UI component. At the same time, the children *ResultsItem*, *Source Box*, *Views Bar* and *Settings Bar* can also be found under


Mauricio Macri

Mauricio Macri

From DBPedia

https://en.wikipedia.org/wiki/Mauricio_Macri

President of Argentina

Mauricio Macri, is the current President of Argentina and has been in office since 2015. A former civil engi
presidential runoff ballotage in Argentina's history and is the first democratically–elected non–Radical or P
1916. He was chief of government of Buenos Aires from 2007 to 2015, and represented the city in the low
of Argentina from 2005 to 2007. The reintegration of Argentina into the international community is central t

Mauricio Macri

Mauricio Macri

From DBPedia

https://en.wikipedia.org/wiki/Mauricio_Macri

President of Argentina

Mauricio Macri, is the current President of Argentina and has been in office since 2015. A former civil engi
presidential runoff ballotage in Argentina's history and is the first democratically–elected non–Radical or P
1916. He was chief of government of Buenos Aires from 2007 to 2015, and represented the city in the low
of Argentina from 2005 to 2007. The reintegration of Argentina into the international community is central t

Mauricio Macri

From Linked Leaks

Officer of FLEG TRAIDING LTD

http://data.ontotext.com/resource/leaks/officer-15002701

Figure 5: **(a) Results Container** component reacts when more results are found in the RDF graphs; additionally, it selects the best view according to the semantics of the results. **(b) View Bar** component reacts after analyzing the results, e.g., the map view is enabled when geo-data is found in the search results.

this UI component. The *reactivity* designed on this component is triggered whenever more data is coming from the RDF graphs, e.g., new results are appended to the list.

Results Item: The purpose of this UI component is to provide a unique container each of the results being obtained from the query will be mapped to, in order to be later displayed in the *Results Container* component. Thus, each *Results Item* UI component will be shown or hidden from the *Results Container*, depending on the navigational input obtained from the *Faceted Bar* component. The *reactivity* designed on this component is triggered by the semantics of the entities contained in the data, e.g., for a person, it may be more relevant to show demographic information, but for an organization, its location information might be more relevant. This is achieved by specialized views in the UI component structure hierarchy.

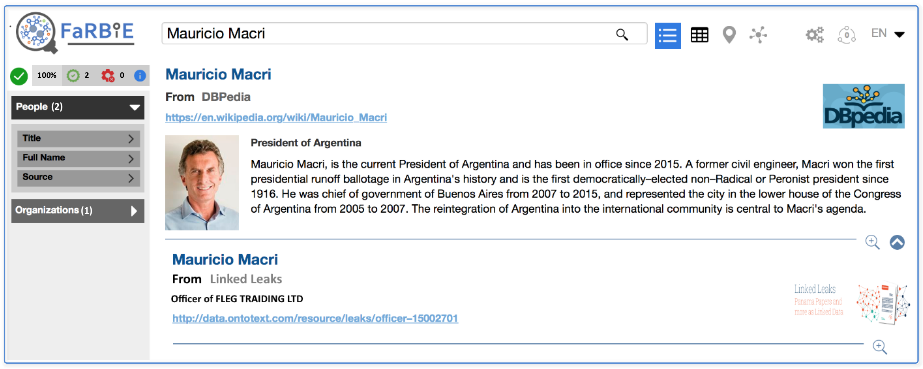
Views Bar: The purpose of this UI component is to provide the users with the possibility of switching the display of the results through different view modes. The *reactivity* designed on this component is triggered by more data coming from the RDF graphs. After analyzing the results, an appropriate view is automatically enabled. *FaRBIE* supports the following common *view modes*:

– *List mode*: This is the default view mode. It displays the results in a list.
– *Table mode*: It displays the results in a table if a comparison of the entity attributes is of interest.
– *Map mode*: It displays the results with location information on a map if geodata is provided.
– *Graph mode*: It uses a graph visualization to display the relationships between the results if a sufficient number of links is found.

Settings Bar: The purpose of this UI component is to provide the user with the possibility of accessing additional options, such as entering credentials to obtain log-in tokens, importing a previously saved configuration file to use as input parameter for the federated search, or selecting the displayed language. Other options could be implemented and supported by future researchers and developers in order to adapt to their needs. No particular reactivity has been incorporated into this component.

4.4 Proof-of-concept

To validate our approach, we have developed a proof-of-concept, based on the introduced use case scenario in the criminal investigation domain. We have configured *FaRBIE* to explore two datasets, namely *DBPedia* and *Linked Leaks*. Figure 6 shows the results using *Mauricio Macri* as keyword. Two people and one organization were found matching the keyword, already providing insights to the possible activities and relation between *Mauricio Macri* and the Panama Papers scandal, with only a single search.



Mauricio Macri
From DBPedia
https://en.wikipedia.org/wiki/Mauricio_Macri

President of Argentina

Mauricio Macri, is the current President of Argentina and has been in office since 2015. A former civil engineer, Macri won the first presidential runoff ballotage in Argentina's history and is the first democratically–elected non–Radical or Peronist president since 1916. He was chief of government of Buenos Aires from 2007 to 2015, and represented the city in the lower house of the Congress of Argentina from 2005 to 2007. The reintegration of Argentina into the international community is central to Macri's agenda.

Mauricio Macri
From Linked Leaks
Officer of FLEG TRAIDING LTD
http://data.ontotext.com/resource/leaks/officer–15002701

Figure 6: **Proof of Concept.** Joined insights from DBpedia and Linked Leaks.

In order to implement *FarBIE*, we evaluated different frameworks for web user interfaces as well as web development platforms resulting in the following selection:

- **ReactJS**[10]**:** A modern javascript library for building web user interfaces. It is a component-oriented library, and the features of virtual DOM it provides fit perfectly the requirements of modifying the user interface dynamically when new data is sent from the server to the user interface. In the work of Khalili et al. [6], it is used as the core technology to provide a reusable set of user interface elements to build Linked Data applications.
- **Web Socket:** A protocol providing full-duplex communication channels over a single TCP connection. It is an ideal protocol to realize the communication of the reactive user interface with the backend system. In our case, a web socket is opened between the LogicKeeper component and the federated search engine. The data is continuously pushed from the server to the client.
- **Play Framework**[11]**:** Is a high velocity web framework for Java and Scala. Many web frameworks, such as Grails, Tomcat, Spring, PHP, or Rails, use threaded servers. A threaded server assigns one thread per request and uses blocking I/O. The play framework is based on an event server (Netty). It assigns one thread/process

[10] `https://facebook.github.io/react/`
[11] `https://www.playframework.com`

per CPU core and uses non-blocking I/O. Threaded vs. event matters in a reactive user interface, as the engine spends most of the time waiting for query results.

The proof-of-concept interface and the intial source code is available as an open source project.[12] *FaRBIE* is empowered with flexible user interface components that react to new data coming from the server. The user is able to explore the portion of search results as soon as they are retrieved from the datasets in the RDF graph federation. Instant filtering is possible without waiting for the complete set of results.

5 Conclusions

We presented *FaRBIE*, a reactive faceted browsing UI to explore multiple RDF graphs from the LOD cloud at a time. *FaRBIE* follows a reactive user interface approach that handles the uncertainty imposed by the intrinsic nature of RDF graphs, with the goal to provide a better user experience. *FaRBIE* is composed of several reactive UI components which react to changes of the semantics, variations in the size of the data, and the disparities in the response times coming from the different sources, allowing for a real-time user experience. We argue that the reactive user interface style used in RDF user interfaces, such as faceted browsing reactivity, is a path for improved overall user experience and becoming a key factor in bringing Linked Data technologies closer to non-technical users. For future work, we plan to formalize and generalize our approach, and perform an empirical evaluation including a user study that compares *FaRBIE* with state-of-the-art interfaces. Finally, we will investigate the feasibility of integrating *FaRBIE* with the aforementioned LD-R framework.

Acknowledgments: The presented work has been funded by the German Ministry of Education and Research (BMBF) in the context of the project LiDaKrA ("Linked-Data-basierte Kriminalanalyse", grant no. 13N13627).

References

1. Arenas, M., Grau, B.C., Kharlamov, E., Marciuska, S., Zheleznyakov, D.: Enabling faceted search over OWL 2 with semfacet. In: Proceedings of the 11th International Workshop on OWL: Experiences and Directions (OWLED 2014) co-located with 13th International Semantic Web Conference on (ISWC 2014), Riva del Garda, Italy, October 17-18. pp. 121–132. CEUR-WS (2014)
2. Arenas, M., Grau, B.C., Kharlamov, E., Marciuska, S., Zheleznyakov, D.: Towards semantic faceted search. In: 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, Companion Volume. pp. 219–220. ACM (2014)
3. Arenas, M., Grau, B.C., Kharlamov, E., Marciuska, S., Zheleznyakov, D.: Faceted search over rdf-based knowledge graphs. J. Web Sem. 37-38, 55–74 (2016)
4. Arenas, M., Grau, B.C., Kharlamov, E., Marciuska, S., Zheleznyakov, D., Jiménez-Ruiz, E.: Semfacet: semantic faceted search over yago. In: 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, Companion Volume. pp. 123–126. ACM (2014)

[12] `https://github.com/LiDaKrA/FaRBIE`

5. Ferré, S.: Expressive and scalable query-based faceted search over SPARQL endpoints. In: The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, Part II. Lecture Notes in Computer Science, vol. 8797, pp. 438–453. Springer (2014)
6. Khalili, A., Loizou, A., van Harmelen, F.: Adaptive linked data-driven web components: Building flexible and reusable semantic web interfaces. In: 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, Proceedings. Lecture Notes in Computer Science, vol. 9678, pp. 677–692. Springer (2016)
7. Nielsen, J.: Usability Engineering. Academic Press, Cambridge, MA, USA (1993)
8. Nielsen, J., Molich, R.: Heuristic evaluation of user interfaces. In: Conference on Human Factors in Computing Systems, CHI 1990, Seattle, WA, USA, April 1-5. pp. 249–256. ACM (1990)
9. Stadler, C., Martin, M., Auer, S.: Exploring the web of spatial data with facete. In: 23rd International World Wide Web Conference, WWW ’14, Seoul, Republic of Korea, April 7-11, Companion Volume. pp. 175–178. ACM (2014)
10. Stolz, A., Hepp, M.: Adaptive faceted search for product comparison on the web of data. In: Engineering the Web in the Big Data Era - 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26. Lecture Notes in Computer Science, vol. 9114, pp. 420–429. Springer (2015)