User Experience Benchmarking and Evaluation to Guide the Development of a Semantic Knowledge Graph Exploration Tool

Roberto García\textsuperscript{a,*}, Juan-Miguel López-Gil\textsuperscript{b}, Jordi Virgili-Gomà\textsuperscript{a} and Rosa Gil\textsuperscript{a}

\textsuperscript{a} Universitat de Lleida, Spain
E-mails: roberto.garcia@udl.cat, jordi.virgili@udl.cat, rosamaria.gil@udl.cat

\textsuperscript{b} University of the Basque Country, Spain
E-mail: juanmiguel.lopez@ehu.eus

Abstract. Despite the increasing amount of semantic data available, there is still a lack of adoption of user-facing applications based on semantic technologies, especially those geared toward the exploration of disparate semantic datasets. Benchmarks have already shown their usefulness leading to advances in different domains and until recently there was not a benchmark of semantic data exploration tools. Using the Benchmark for End-User Structured Data User Interfaces (BESDUI), we explore now how it can guide the development of a new tool for semantic knowledge graphs exploration, RhizomerEye. The results at the current stage of development show better results than those for the RhizomerEye predecessor. However, there is the risk of overfitting the tool to the benchmark, overloading the user interface to produce the best benchmark results but producing an unusable UI. To avoid this, an evaluation with real users has been also conducted, using the same dataset and tasks provided by the benchmark, but involving real users to measure the User Experience (UX) instead of deriving the UX metrics analytically. Moreover, the evaluation has been complemented with the user satisfaction dimension, unmeasurable by the benchmark. Overall the results are promising, showing comparable results to those of the benchmark, especially for users with knowledge about semantic technologies. On the other hand, the evaluation with real users has made it possible to identify potential improvements to RhizomerEye, also taking into account user satisfaction and ways to better suit BESDUI to be used in evaluations with real users.

Keywords: Semantic data, Benchmark, Exploration, Usability

1. Introduction

Through open data efforts such as the Linked Open Data Cloud [1], web pages annotated using schema.org [2] motivated by Search Engine Optimisation benefits or the increasingly popular Knowledge Graphs [3], the amount of semantic data is increasing. However, this has not translated into increased adoption of user-facing applications based on semantic technologies [4], and, usually, this is intended. The benefits of semantic technologies should be evident on the server side, while client applications should isolate end-users from the underlying complexities. This is the case for task-specific applications with a user interface tailored to the task at hand.

However, there are cases where no custom client applications have been built to hide the underlying semantic data from end users, in many cases because the application niche is too narrow or recent. This is especially true for users exploring semantic data they want to reuse, especially if it is the combination of multiple datasets [5], or
for semantic data they are generating and want to inspect to assert its quality. Tech-savvy users can use standards like SPARQL to explore semantic datasets. However, most users are unable to do this, and even developers who are conversant with SPARQL struggle when querying unfamiliar datasets [6].

Consequently, there are multiple attempts to make semantic data exploration more interactive and intuitive [7]. Nevertheless, due to this heterogeneity, these efforts are difficult to compare, particularly from the user’s point of view. Therefore, a reference benchmark was required [8] and our proposal was the Benchmark for End-User Structured Data User Interfaces (BESDUI) [9]. BESDUI showed its usefulness as a benchmark for existing structured data exploration tools, including both semantic and non-semantic data.

Continuing this line of work, we focus now on other areas where BESDUI can make a contribution. First of all, as a way to guide the development of RhizomerEye, a new version of an existing semantic data exploration tool called Rhizomer [10]. In this regard, BESDUI can be easily integrated with the development process because it does not require the involvement of real users. The benchmark is based on a set of predefined tasks and data plus the analysis of the interaction steps required to complete the tasks. This has helped during RhizomerEye development, providing an unbiased and very generic testing framework that also facilitates repeatability. Moreover, it encourages the development of tools that better fill the niche of generic semantic data exploration tools supporting a wide range of tasks. To this end, and as detailed in Section 3.4, BESDUI includes metrics that take into account not just efficiency but also effectiveness regarding the set of predefined data exploration tasks.

Additionally, it is important to note that BESDUI does not replace tests with real users, who are still required to measure subjective aspects of the User Experience (UX) such as learnability or user satisfaction [11]. To this end, and also to rule out that basing RhizomerEye development on the benchmark does not result in overfitting or user interface overloading, this paper also contributes an evaluation involving real users using the same data and tasks as the benchmark. The aim of this evaluation is to check that RhizomerEye is still usable and that the completion times with real users converge with those obtained analytically with BESDUI for experienced users. This is the expected behavior because BESDUI measures the ideal set of interaction steps followed to accomplish each task using a particular tool.

From the analysis of the interaction logs provided by all participants in the study, it has been possible to assess all interaction steps that BESDUI uses to compute its metrics and also the actual time users spend to complete each of the tasks using RhizomerEye, when they were able to complete them. Even though the time required to fulfill the proposed tasks is almost four times longer than the analytical results for the average of all participants, the important insight is that the results with real users converge with the analytical results of BESDUI when we limit the comparison to those participants that report previous experience with RhizomerEye or Semantic Web technologies in general.

The rest of this paper is organized as follows. First, we present the related work in Section 2, focusing on that related to UX benchmarking and evaluation in the context of the Semantic Web. Then, in Section 3, we report on our experience using BESDUI to drive the development of a new semantic data exploration tool, which we call RhizomerEye. In Section 4, we detail the study with real users we have carried out using BESDUI data and tasks, which included the evaluation of RhizomerEye’s effectiveness, efficiency, and satisfaction in use. Finally, Sections 5 and 6 present the discussion and the conclusions and future work respectively.

2. Related Work

Given the user interaction challenges when dealing with large amounts of semantic data, which is continually growing, there are multiple attempts to develop user-facing applications in a wide range of domains, such as healthcare [12], energy [13] or digital humanities [14]. More generic tools are also being developed [15], ranging from Linked Data browsers [16] to Controlled Natural Language query engines [17] or faceted browsers [10].

However, they are difficult to compare and evaluate due to a lack of well-established practices for UX evaluation tailored to the Semantic Web domain [18]. There are some proposals based on evaluation with real users, ranging from the exploration and visualization of the ontologies underlying semantic data [19] to semantic data exploration tools [20]. However, comparability is difficult when users need to be involved and it is impossible to guarantee reproducibility at scale.
The most promising tool in this kind of situation is benchmarking, with a long tradition of helping to organize and strengthen research efforts in a particular research area [21]. An example is the Text REtrieval Conference (TREC) benchmarks [22], which have become the de facto standard for evaluating any text document retrieval system. Also, there are success stories in areas related to the Semantic Web like the Ontology Alignment Evaluation Initiative [23].

However, in connection with semantic web and user interaction, there have been just a few efforts based on quite informal criteria, such as the Intelligent Exploration of Semantic Data Challenge\(^1\). The Benchmark for End-User Structured Data User Interfaces (BESDUI) [9] is the first one focusing on user interaction with structured data, thus including semantic data. BESDUI is the starting point for the contributions made in this paper. First, regarding the use of this benchmark to guide the development of a new semantic data exploration tool. Second, about evaluating the suitability of the benchmark in comparison to an evaluation with real users, thus not based on the analytical approach that BESDUI is based on.

Regarding an appropriate sample size for user testing, it is crucial to obtain reliable and meaningful results while keeping under control this costly activity. Jakob Nielsen claims that with a sample size of only 5 participants, it is possible to identify 80% of the existing usability problems [24]. In this line, Steve Krug, in his book "Don’t Make Me Think" [25] and Erika Hall, in "Just Enough Research" [26], also concur with the recommendation of having at least 5 participants as sufficient to detect important usability issues. This statement is supported by both theoretical and practical arguments in each case.

To go beyond qualitative results and also obtain reliable quantitative ones, we have followed the recommendations by Nielsen and Landauer in [27], where it is estimated the optimal number of participants in a user test for different project sizes. The authors propose 15 participants in medium to large projects like RhizomerEye. Following this recommendation, and as detailed in Section 4, our objective will be to involve at least 15 participants in the evaluations with real users.

3. Developing a Semantic Data Exploration Tool

This section reports on the development of a new version of Rhizomer [10], an existing semantic data exploration and visualization tool. Rhizomer was developed before BESDUI was available, though it is among the tools that have been evaluated using this benchmark. Rhizomer already showed very good results for the benchmark, but its analysis highlighted many improvement opportunities.

Despite the good results, and taking into account the potential improvements already identified in that previous study using BESDUI [9], Rhizomer was based on an aged technological stack and its architecture made it very difficult for others to use it without going through a complicated deployment process. This triggered the development of a new version, called RhizomerEye [28], which was developed from scratch but guided by the same approach that had already shown its usefulness with Rhizomer.

Rhizomer’s approach is motivated by the aim of exploring a semantic dataset without any prior knowledge about its structure or the underlying semantic data and query languages. And it is addressed through the three classical data analysis tasks proposed by Shneiderman [29]: (1) getting an overview of the data, (2) zooming and filtering, and (3) viewing details on demand. Each of these tasks is further detailed in the following subsections, including details about how the BESDUI benchmark influenced the development of the RhizomerEye’s features supporting each of these tasks.

The added value of driving the development using the BESDUI benchmark is that the tasks under consideration are just based on the typical information needs for data exploration considered in the Berlin SPARQL Benchmark (BSBM) Explore Use Case\(^2\). They were conceived without considering the user interaction dimension, just considering typical information needs and the SPARQL queries required to satisfy them. And that is our objective when


\(^2\)Berlin SPARQL Benchmark (BSBM) Explore Use Case, http://wbsg.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/spec/ExploreUseCase/
developing RhizomerEye, to develop a tool capable of satisfying these fundamental information needs when exploring a dataset, and make them possible through a user interface that requires the minimum amount of interaction steps to complete them. This approach is useful during the development process, as detailed in the next subsections. However, this analytical approach ignores important aspects like user interface complexity resulting from overloading it or user satisfaction. To address these shortcomings, we have also conducted an evaluation with real users, as reported in Section 4.

3.1. Overview

This is the first of the data analysis task supported by RhizomerEye. It allows users to get the full picture of the dataset. Rhizomer automatically generates a word cloud to provide an overview of the kinds of things in the dataset. This is the default overview mechanism because it works even for really big datasets like DBpedia, with more than 100 million statements, because the 300 most common classes are displayed.

For a more informative overview that also includes how the main classes are related, there is also the option of a network representation. In this case, the 30 most instantiated classes are shown as nodes together with the most frequent properties connecting them as labeled edges. Figure 1 shows an example of the network overview of the BESDUI benchmark dataset based on the BSBM benchmark. In this case, it is important to note that this network overview is derived from the data and not a predefined ontology, like done by ontology visualization tools like VOWL [30].

Both overview features are derived from querying the underlying data with SPARQL queries [31]. This approach makes it possible to even explore ontology-less data, such as that generated by directly transforming existing data to RDF, or in any case to check that the explored data is properly connected and follows the intended ontologies.
It is also possible to point RhizomerEye to the ontologies that structure the data to be explored. In this case, Rhizomer will retrieve the labels for the classes and properties from the ontologies to render more user-friendly presentations. The overview task is supported by the set of features implemented by RhizomerEye:

- **Word Cloud**: overview the classes in a dataset through a word cloud with the names of the classes and where their size is proportional to the number of instances of each class.
- **Network**: an overview of the main classes, and relationships among them, using a network representation that includes classes and relationship names, as shown in the center of Figure 1.
- **Classes Autocomplete**: this feature is implemented as an input field, which autocompletes the text typed by the user to the labels or local names of the classes instantiated in the dataset and allows choosing the class to focus on. This autocompleted input field appears at the top of Figure 1.

The development of the previous features, combined with the use of the labels from the underlying ontologies, have been ultimately driven by the BESDUI benchmark. These features make it possible to initiate all the tasks based on locating a specific class (product, product subclass, or review based on the benchmark tasks) directly from the overview while requiring the minimum number of interaction steps, just pointing and clicking. Even in the case that the class to focus on is not among the most instantiated ones, the classes autocomplete will minimize the number of interaction steps and thus produce a good result based on the benchmark.

### 3.2. Zoom and Filter

For the second data analysis task, usually after selecting a class from the overview or from the results of a text search, RhizomerEye generates a faceted view. This way the user zooms into that particular class and can filter its instances using facets based on their properties, as shown in Figure 2. This view is also generated automatically, driven by the underlying data. This approach also allows exploring data that does not fully comply with an existing schema and highlights these inconsistencies during exploration to help users spot them.

As in the Overview case, the Zoom and Filter tasks supported by RhizomerEye make use of the ontologies that the data is based on if they are available. In this case, the ontologies are just used to retrieve properties, ranges, and values labels. The zoom-and-filter task is supported by the following set of features, which have been also driven by the BESDUI benchmark as detailed next:

- **Facet Values Filter**: this is the basic component of the faceted view that, for each facet, shows the 10 most common values for the corresponding property when used by the instances of the class under the focus. Clicking any value filters the displayed instances to those featuring that value for the property. Clicking again would instead exclude that value from the results, as shown in Figure 2 for facet "productFeature" where the value highlighted in red is excluded. Moreover, if multiple values are selected for a facet, it is possible to switch between requiring that all of them are present or just some of them. This combination of facet value filtering options has been implemented because in combination they make it possible to complete all the BESDUI tasks that involve filtering just one class. This includes if multiple values of the same facet should be combined and all match together or only at least one of them, or if some of them must be excluded. As mentioned in the introduction, these decisions guided by the benchmark might reduce RhizomerEye’s usability. To avoid that, an evaluation with real users has been carried out, as reported in Section 4. For instance, Figure 2 shows a combination of a required and an excluded facet value required by one of the BESDUI’s tasks. Moreover, all facet filters are summarised at the top of the faceted view to help users keep track of the restrictions being placed.

- **Facet Values Autocomplete**: in addition to the facet values filter, each facet features an input field where users can start typing to filter based on less common facet values, i.e., those not among the top ten. The input field features autocompletion to the potential values based on the filter applied so far. In terms of the BESDUI benchmark, this feature reduces the number of interaction steps required to complete those tasks that involve filtering based on less common facet values. In any case, the feasibility of this decision from the UX point of view will be validated through the study with real users.
Fig. 2. Faceted view (click)

- **Numeric Range Facet**: for facets with numeric ranges (such as integer, decimal, or year), the minimum and maximum values are shown together with a slider to filter instances based on a user-defined range of values. Many benchmark tasks involve range restrictions on numeric properties and a range display with sliders for the upper and lower limits has been implemented to reduce the required interaction steps. In any case, in order to avoid overloading the user interface, like for the facet values filter, all facets appear collapsed and need to be expanded before they can be used for filtering.

- **Class Text Search**: this feature is implemented as an input field displayed on top of all facets. It implements a text search among all facet values for the current class. For some BESDUI tasks, this is more efficient than expanding a particular facet and using the autocomplete feature for that particular facet.

- **Global Text Search**: like the previous feature, this input field performs a text search based on the user input though, in this case, the search is across the whole dataset, not just a particular class that has been previously focused in. Consequently, it is available from the page showing the overview, as shown at the top of Figure 1. However, it is classified as a filtering feature because, ultimately, it is implementing a class text search, though on all classes in the dataset. The result of a global text search is also a faceted view but with just one facet, the type facet. Some sample instances related to literals containing the typed text, or resources whose label contains that text, are shown in that type-based faceted view. However, if the user wants to dig deeper, one of the types should be selected from
the type facet. Then, the faceted view for that class is displayed with the class text search filter already applied based on the text typed by the user for the global search.

This feature supports the BESDUI tasks requiring reaching a particular instance, e.g. a known product, with a minimal amount of interaction steps because it is not needed to go through the corresponding class faceted view first, or if the class corresponding to the instance is not known or unclear from the task description.

3.3. Details-on-demand

After zooming and filtering, the user reaches the instances of interest. The paginated list of instances in the faceted view just shows some of the resource properties, like label, caption, or comment. After clicking one of those instances, the details view displays all its properties and values. Users can also browse resources linked directly or through reverse facets, as shown in Figure 3.

3.4. Benchmark Results

During the RhizomerEye development process, the tool was repeatedly evaluated using the BESDUI benchmark in order to improve the resulting metric values. Based on the Keystroke Level Model (KLM) [32], BESDUI takes into account the following fundamental user actions: K for keystrokes or button presses, P for mouse pointing to a target, and H for homing the hands on the keyboard or other device, which includes movement between any two devices as well as the fine positioning of the hand.
The KLM counts of the user interactions needed to complete a task, if possible with the analyzed tool, are then used to calculate the efficiency metrics. Time is the conversion of the operators into a time measure, where each keystroke or mouse click K accounts for 0.2 seconds, mouse pointing actions P to 1.1 seconds, and homing the hands H equals 0.4 seconds. Operator Count simply refers to the total number of operators required. In addition, BESDUI includes three quality-in-use metrics, one for effectiveness and two for efficiency:

- **Capability (C) (effectiveness):** what proportion of one task is completed (0% if not possible to complete or 100% otherwise) or, for the whole benchmark, the percentage of all 12 tasks completed.

- **Operator Count (OC) (efficiency):** how many KLM Operators are required to complete a task or the average count just for completed tasks.

- **Time (T) (efficiency):** each KLM Operator has a corresponding average time to complete it as detailed previously. For a task, this metric is computed by multiplying, for each operator type, the time for each operator by the operator count. Then, summing them all together. For the whole benchmark, it is the average time considering just the completed tasks.

Additionally, BESDUI also proposed a combined effectiveness/efficiency metric that has been also computed for RhizomerEye:

- **Task Efficiency (TE) (effectiveness/efficiency):** measured as the ratio of Capability to Time over one minute, "goals per minute". For the whole benchmark, it is computed using the percentage of all 12 tasks completed divided by the average time for the completed tasks, then multiplied by 60 to compute the goals per minute.

Based on the following metrics, the results for RhizomerEye at its current development stage are compared to those already provided for different structured data exploration tools as shown in Table 1. As can be observed, RhizomerEye already shows the same Capability that its predecessor Rhizomer, the highest among the semantic data exploration tools. The best Capability is that of Sieuferd [33], which is a relational data exploration tool that can be also benchmarked using BESDUI because the benchmark also provides the same data in relation format and the corresponding tasks as SQL queries.

However, this high Capability is possible due to a very complex user interface, which penalizes Sieuferd on the other metrics, because completing each task requires a lot of interaction steps. On the other hand, PepeSearch [34] is the tool with the lowest Operator Count and Time but due to a really simple user interface for semantic data exploration, which just allows completing the three simplest tasks. The Task Efficiency metric is the most relevant one because it balances Capability and Time. In this case, RhizomerEye is now the best-performing tool, with 2.8 tasks per minute, surpassing the previous leader, Rhizomer with 2.2 tasks per minute. Based on these results, RhizomerEye has benefited from using the BESDUI to guide its development. In the next section, we check if this also translates to a fruitful and satisfactory UX involving real users.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Capability</th>
<th>K (0.2s)</th>
<th>P (1.1s)</th>
<th>H (0.4s)</th>
<th>Operator Count</th>
<th>Time</th>
<th>Task Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rhizomer</td>
<td>58%</td>
<td>16.4</td>
<td>11</td>
<td>2</td>
<td>29.4</td>
<td>16.2</td>
<td>2.2</td>
</tr>
<tr>
<td>RhizomerEye</td>
<td>58%</td>
<td>13</td>
<td>8.3</td>
<td>1.7</td>
<td>23</td>
<td>12.4</td>
<td>2.8</td>
</tr>
<tr>
<td>Virtuoso FCT</td>
<td>46%</td>
<td>23.5</td>
<td>14.5</td>
<td>3.5</td>
<td>41.5</td>
<td>22.1</td>
<td>1.2</td>
</tr>
<tr>
<td>Sieuferd</td>
<td>96%</td>
<td>48.7</td>
<td>19.8</td>
<td>2.9</td>
<td>71.3</td>
<td>32.63</td>
<td>1.8</td>
</tr>
<tr>
<td>PepeSearch</td>
<td>17%</td>
<td>7</td>
<td>2.5</td>
<td>1.5</td>
<td>11</td>
<td>4.8</td>
<td>2.1</td>
</tr>
</tbody>
</table>

4. Study

In this section, we describe the study conducted to validate the analytic results produced by BESDUI with real users. Once the material is presented, the instruments used to gather information throughout the session are reported.
and the collected measures are detailed. The participants that took part in the study, together with the definition of
the tasks they had to perform are then described. Finally, the procedure followed in conducting the study is detailed
and the obtained results are shown.

4.1. Material and Instruments

The BESDUI benchmark was chosen because it defines a suite of benchmarks for comparing the performance of
these systems across architectures [9]. It is based on the Berlin SPARQL Benchmark (BSBM) which, to enable users
to discuss the queries in real-world terms, instantiates a single arbitrary e-commerce schema and a testing dataset
available both as relational and semantic data. For the study, the RhizomerEye tool was used, whose technical details
are detailed in [28]. This tool helps explore any knowledge graph available as Semantic Web data as it is not tailored
to a specific data schema or ontology, like those used in the BESDUI dataset.

The user logs in the sessions were captured using rrweb.io, an open-source web session replay library that provides easy-to-use APIs to record user’s interactions and replay them remotely. The library allows collecting the
different interactions that participants perform in a specific tab on their browser. A series of questionnaires were also used, specifically the After-Scenario Questionnaire (ASQ) [35], the UMUX-Lite [36], and the TAM [37]. Feedback participants provided during and after performing the tasks was also recorded.

4.2. Measures

To measure usability, the metrics proposed in the ISO/IEC 25010 standard [38] were used. This standard views
usability as a component of quality in use with three elements: effectiveness in use, efficiency in use, and satisfaction in use. For effectiveness in use, we used the proposed metric "percentage of accomplished tasks". Regarding efficiency in use, "times per task" was the chosen metric from those proposed by ISO/IEC 25010.

In addition to that, user interaction logs were also gathered as they define the steps of the interaction. As rrweb.io logs all events generated through the navigation in a replayable HTML file, these logs were extracted to a JSON file. Then, we developed a script to extract the events that tackle specific user interactions. These events included inputs, mouse movements, mouse interactions (such as mouse up, down, click, or double-click), and scrolls. They were subsequently translated to the KLM model (see Section 3.4): K for keystrokes or button presses, P for mouse pointing to a target, and H for homing the hands on the keyboard or other device. Apart from the events in KLM, scrolling was also considered, defined as the amount of scrolling users performed during the task at hand.

Listing 1 shows a fragment of a rrweb.io JSON log and the comments in the right column describe the corresponding events and KLM actions that can be derived. For instance, there are mouse clicks when the log entry specifies that the source is 2, or key inputs when it is 5. From the stream of identified events, it is possible to derive the KLM interaction actions. For instance, to count a K action for each mouse click or key press. Additionally, when the user switches from the mouse to the keyboard, or vice-versa, a homing H action is also derived. On the other hand, mouse movements are also logged so, when there is a click preceded by mouse movements it is possible to derive a pointing P action.

```
Listing 1: Example of rrweb.io JSON log illustrating the identified KLM events
```

```
[ ...,
  {
    "data": {
      "source": 1, // Mouse movement
      "positions": [{
        "id": 2208, // Overed element
        "x": 431, "y": 826, // Coordinates
        "timeOffset": -401
      }, ...]
  }
```

3https://github.com/rrweb-io/rrweb
As for satisfaction in use, we used the ASQ questionnaire to measure it, using a 7-point Likert scale ranging from 1 (strongly disagree) to 7 (strongly agree). For measuring overall usability, another two questionnaires were used. First, the Technology Acceptance Model (TAM) questionnaire, which [37] is composed of twelve questions designed to measure user acceptance of a given technology, using a 7-point Likert scale ranging from 1 (extremely disagree) to 7 (extremely agree). Second, the UMUX-Lite questionnaire [36], which is designed to obtain a measurement of perceived usability that includes two questions and uses a 7-point Likert scale ranging from 1 (strongly agree) to 7 (strongly disagree).

4.3. Participants

Sixteen (16) participants took part in the study, following the recommendations by Nielsen and Landau [27] about the number of participants in a user test detailed in Section 2. The participants were between 25 and 53 years old (40 on average). There were twelve (12) men and four (4) women. All participants were experienced in data exploration and processing tasks in different domains (computer science, agronomy, or bioinformatics) and reported an average confidence in these skills of 4.125, on a scale between 1 and 5.

Regarding their knowledge of the Semantic Web and related technologies, on the same 1 to 5 scale, they reported an average knowledge of the Semantic Web of 2.25, 2.13 in the case of graph-oriented databases, and 1.75 in the SPARQL query language. Finally, 8 of them reported previous experience with Semantic Web tools, including ontology editors like Protege, semantic datasets like DBPedia, semantic data explorers like Rhizomer, or graph databases like Virtuoso. However, just 4 of them in the last year and self-reporting a knowledge of the Semantic Web higher than 3 in the previous scale. This subset of 4 users is considered the "experienced real users" in Table4.

4.4. Tasks

The tasks to be performed were the ones defined in BESDUI [9]. Except for one, tasks are all straight adaptations of the Berlin SPARQL Benchmark (BSBM). To cover a gap in the initial benchmark, Task 2 has been added as a variation of Task 1 (OR versus AND operations for combining subqueries). The tasks are:
Task 1. Find products for a given set of combined features:
A client seeks a product that presents a specific set of features that should all be satisfied.

Task 2. Find products for a given set of alternative features:
Contrary to Task 1, the client is seeking a product satisfying at least one of some alternative features. Task 2 has been added beyond those provided by BSBM. It makes Task 1 less specific by considering feature alternatives. This benchmarks how exploration tools let users define OR operations.

Task 3. Retrieve basic information about a specific product for display purposes:
The client wants to view all available information about a specific product.

Task 4. Find products having some specific features and not having one feature:
The client has a more specific idea about what she wants, i.e. features that the products should have and others that the product should not.

Task 6. Find products that are similar to a given product:
After finding a product that fulfills the client’s expectations, she wants to find products with similar features.

Task 7. Find products having a name that contains some text:
The client remembers parts of a product name and wants to find the product again using those parts of the name.

Task 10. Get information about a reviewer:
In order to decide whether to trust a review, the client asks for any kind of information that is available about the reviewer.

4.5. Procedure
First and before the evaluation session, the participants filled out a demographic questionnaire to check their previous knowledge about information search and the semantic web. One evaluator guided the evaluation session for all participants. The evaluator presented each participant with a sequence of tasks to be performed. For each participant, the order in which the tasks were performed was randomly selected to avoid any bias and nullify the learning effect. Next, the participants used RhizomerEye in a browser to carry out each task without a time limit. RhizomerEye was opened in a Chrome web driver, which allowed every session to be recorded with the tracking script by rrweb.io. After finishing or abandoning each task, participants filled out the ASQ questionnaire about the corresponding task. In addition, in order to obtain additional feedback, participants were instructed to briefly discuss with the evaluator the performance and problems found with each task. After completing all tasks, participants completed the TAM and the UMUX-Lite questionnaires. The whole session lasted approximately 30-40 minutes, depending on the user.

4.6. Results
For efficacy in use, Table 2 displays the efficacy in terms of the amount and percentage of tasks accomplished by the participants.

<table>
<thead>
<tr>
<th>Task</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 6</th>
<th>Task 7</th>
<th>Task 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passed</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>13</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Not passed</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Percentage</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>81.25%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Regarding efficiency in use, Figure 4 displays the interaction time results for all tasks, except for Task 6, which are shown in Figure 5. Results are displayed as line charts for each task, with the X axis showing the participants and the Y axis showing the time in seconds for the given participant in the given task.
For satisfaction in use, Figure 6 displays the results of the ASQ, showing one radar plot for each task. The radar plot for each task shows the results of the three questions in the ASQ questionnaire for each participant, with an axis representing the current participant (values from U1 to U16), values of the answers ranging from 1 to 7 (1 in the center of the radar), and grids with different color for each question linking the results of each participant (see legend in the bottom right part of the figure for the specific color of each question).

Figure 7 displays the results of the TAM questionnaire for each score (X-axis) and TAM question (Y-axis) in a scatter plot. The scores range from 1 to 7, with 1 being the smallest possible value and 7 being the best possible score. Each dot represents the answer of a specific participant for the corresponding question, with the color of the dots determining the participant (see legend in the upper left part of the figure for the color of each participant). The maximum, minimum, mean, and 25 and 75 percentiles for each TAM question are displayed. The first 6 questions describe perceived usefulness, which achieved a mean of 5.18. As for the last 6 questions, they describe the perceived ease of use, reaching a mean of 4.31.

The results for the UMUX-Lite questionnaire are displayed in Table 3. The table shows the minimum, mean, maximum, and standard deviation for the marks responded by the participants for each question. Scores range from 1 to 7, with 1 representing the highest attainable score. Additionally, the last row in the table shows the computed UMUX-Lite score, ranging from 1 to 100, the higher the better.

5. Discussion

First, the benchmark results show how effective and efficient the analyzed user interfaces are in the scope of the different metrics and tasks defined in BESDUI [9]. The tool with the highest effectiveness is Sieuferd, which can finish all tasks with the exception of Task 12 (it is only partially completed). However, these powerful capabilities are accompanied by a more complex user interface, which offers the lowest efficiency: on average, these activities require over 71 operators and 32 seconds to execute. On the other hand, PepeSearch only allows performing three
Fig. 5. Interaction time results for Task 6. Participants 2, 3, and 6 did not complete the task. The reported time is till the moment they gave up trying to complete the task. Interactive version (click)

Table 3

<table>
<thead>
<tr>
<th>UMUX-Lite question</th>
<th>Min</th>
<th>Mean</th>
<th>Max</th>
<th>Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>This system’s capabilities meet my requirements</td>
<td>2.00</td>
<td>3.88</td>
<td>7.00</td>
<td>1.96</td>
</tr>
<tr>
<td>This system is easy to use</td>
<td>1.00</td>
<td>3.06</td>
<td>5.00</td>
<td>1.29</td>
</tr>
<tr>
<td>Mean</td>
<td>1.50</td>
<td>3.47</td>
<td>6.00</td>
<td>1.45</td>
</tr>
<tr>
<td>Dev.</td>
<td>0.71</td>
<td>0.57</td>
<td>1.41</td>
<td>0.73</td>
</tr>
<tr>
<td>UMUX-Lite score</td>
<td>16.67</td>
<td>58.85</td>
<td>91.67</td>
<td>10.87</td>
</tr>
</tbody>
</table>

Table 3 minimum, mean, maximum and standard deviation for UMUX-Lite results, where (1) corresponds to "strongly agree" and (7) to "strongly disagree", and the UMUX-Lite score, from 1 to 100, the higher the better.

tasks because of its much simpler user interface, but despite its simplicity, it is quite efficient, taking 11 operators and only 4.8 seconds on average to finish these three tasks. Around half of the tasks are supported by Rhizomer, RhizomerEye, and Virtuoso, which are more efficient than Sieuferd and less efficient than PepeSearch. As for Task Efficiency, it enables obtaining a score that is more evenly distributed between effectiveness and efficiency. RhizomerEye receives the highest score because it offers a 58% Capability with less than 40% of the time that Sieuferd requires, translating to a Task Efficiency of 2.8 goals per minute. Rhizomer comes next with 2.2 goals per minute. This final statistic seeks to strike a balance between effectiveness and efficiency by penalizing technologies that are overly focused on a small number of tasks. Therefore, despite being the most efficient tool in terms of Operator Count and Time, PepeSearch receives the third-best Task Efficiency of 2.1 goals per minute. Sieuferd comes next, and then Virtuoso.

The results of the comparison have been supported by the evaluation with real users. The participants in the study performed all but one of the tasks in an efficient and effective way, and are overall quite satisfied with RhizomerEye. The efficacy in use results indicate that all tasks were completed by all participants, except for Task 6, which was correctly finished by 13 out of 16 total participants (see Table 2). As for efficiency in use, the total interaction time
Fig. 6. ASQ questionnaire results per task and user (U1-U16), where (1) corresponds to "strongly disagree" in the center of the radar and (7) to "strongly agree". Interactive version (click)

for performing each task (see Figure 4) shows the amount of time each participant needed to perform each task. Even though the order of the tasks was randomized, people who had to perform a relatively difficult task first did not have previous knowledge of the platform, which led to visible differences in the total interaction time amongst participants for the same task.

Besides, there were some cases in which a given user was not aware of the link between some aspect mentioned
in the task definition and its corresponding text in the user interface, such as participant 6 in Task 10, who did not realize that the author of review and reviewer referred to the same concept until much later than other participants. In Task 6, as shown in Figure 5, data from participants 2, 3, and 6 were recorded, but these participants did not finally complete the task. The difference in time between the participants was caused by the manner in which the users performed the search, as only one user identified label as the name of the instance and was capable of fast-filtering product features. The other participants had to check them one by one, therefore requiring more time.

Regarding satisfaction in use, Figure 6 shows the answers of the participants after each task. Apart from the randomization of tasks for each user, variability in responses depended heavily on the task, with substantial differences between them. The shape of the grids for each question and task displays such differences. Overall, tasks that were more simple to perform, such as Tasks 1 and 2, received better scores than other more complicated ones. Task 6 is noticeably the task that participants valued the worst, scoring less than 4 out of 7 (with 7 being the best) in almost all cases, which indicates dissatisfaction with the completion of the task.

With regard to overall usability, results for TAM (see Figure 7) are generally quite good, with most questions scoring 5 on a scale from 1 to 7, being 7 the best score. There are differences between results for TAM questions, but the mean of all of them is above 4 out of 7, indicating good acceptance of RhizomerEye by the participants. According to TAM, good ratings of usefulness and ease of use (perceived usability) influence the intention to use, which influences the actual likelihood of use [37].

Furthermore, results of UMUX-Lite (see Table 3) show that the mean result for UMUX-Lite responses for all experts was 3.47 out of 7 (with 1 being the best possible score). It also shows the UMUX-Lite score [39] for each participant, which achieved a mean of 58.85 in a ranking from 0 to 100. To contextualize the meaning of the achieved score, the System Usability Scale (SUS) [40], one of the most commonly used questionnaires to evaluate the perceived usability of a given system, consists of ten questions on a five-point Likert scale that provides a score from 0 to 100 range. As [36] demonstrated, there is a correspondence between SUS and UMUX-Lite scores, which indicates that the usability of RhizomerEye is appreciated as good by the participants.

As reported by the participants themselves or derived from user session replays, several aspects of the RhizomerEye user interface noticeably helped the participants fulfilling the tasks:
– **Breadcrumbs**: participants valued having visual feedback on the selections they had made, which allowed them to check mistakes and correct them (see the upper part of Figure 2, breadcrumbs are in yellow)

– **Exclude option**: all participants recognize the exclude option as the NEGATION they required to fulfill some of the tasks (see left of Figure 2, exclude appears in red as a clickable option on the selected *stroboscopes* product feature)

– **OR**: all participants adequately interpreted that the OR option had to appear when clicking on the AND (see left of Figure 2, the AND appears on the right side of *ProductFeature*)

Regarding the issues detected during user evaluation, they have arisen due to different factors. First, a series of issues arose from the task definitions in the benchmark that proved to be ambiguous for the participants:

– **similar**: Even though Task 6 defined product similarity as *with shared features*, except for two participants, the others did not have a clear idea of what *similar* meant, and three decided to leave the task undone. Two participants explicitly asked the facilitator what similarity was, and from there on they were able to complete the task at hand.

– **author of the review**: Some participants spent more time than the one they needed because they did not associate the *author of the review* in the definition of the task with the *reviewer* field in the database, while one of them was lost until it was figured out after many turnarounds through the user interface, as it can be seen Figure 4. It happened in Task 10.

– **name**: the field *label* in the database was not detected as the *name* of an instance by almost none of the users, leading to most users not being able to detect the most efficient way to solve some of the tasks. It happened mainly in Tasks 3 and 6.

From the analysis of the interaction logs produced by all evaluated participants, it has been possible to measure all interaction steps based on the KLM model and the real-time needed to complete each of the tasks using RhizomerEye, when they were able to complete it. The results are presented in Table 4 together with the analytical results for RhizomerEye already presented in Table 1.

<table>
<thead>
<tr>
<th>RhizomerEye</th>
<th>Capability</th>
<th>K (0.2s)</th>
<th>P (1.1s)</th>
<th>H (0.4s)</th>
<th>Operator Count</th>
<th>Time</th>
<th>Task Efficiency</th>
<th>Time  (real)</th>
<th>Task Efficiency (real)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytical</td>
<td>58%</td>
<td>13.0</td>
<td>8.3</td>
<td>1.7</td>
<td>23.0</td>
<td>12.4</td>
<td>2.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Experienced real users</td>
<td>58%</td>
<td>18.6</td>
<td>13.5</td>
<td>4.6</td>
<td>36.7</td>
<td>20.4</td>
<td>1.7</td>
<td>86.9</td>
<td>0.4</td>
</tr>
<tr>
<td>Avg. real users</td>
<td>57%</td>
<td>29.0</td>
<td>16.7</td>
<td>9.2</td>
<td>54.9</td>
<td>27.8</td>
<td>1.2</td>
<td>108.6</td>
<td>0.3</td>
</tr>
<tr>
<td>Avg. real users w/o Task 6</td>
<td>50%</td>
<td>21.9</td>
<td>11.8</td>
<td>6.4</td>
<td>40.1</td>
<td>19.9</td>
<td>1.5</td>
<td>73.4</td>
<td>0.4</td>
</tr>
</tbody>
</table>

As it can be observed in Table 4, though the results for the average of all participants more than double the analytical results, when we restrict the comparison to just the participants with experience in Semantic Web technologies, as defined in Section 4.3, the results are more similar. Considering that the participants showing the best results are among those who reported they had knowledge about SPARQL and semantic web platforms, we can say that the evaluation with real users does confirm that the analytical results approximate thus of an experienced real user as expected.

The results for the average of all participants improve significantly when Task 6 is excluded; getting results similar to those for just the experienced users. Consequently, it seems that there has been some overfitting as a consequence of applying BESDUI during the development of RhizomerEye, but just in the case of Task 6. This supports the impressions gathered by the evaluator during the sessions about being a task too complex or confusing for many of the participants, even if they finally completed it. Consequently, it is worth considering a new approach to supporting Tasks 6 as detailed next.
The biggest deviation observed between analytical and results with real users is between the time derived analyti-
cally and the time really required. This also translates to task efficiency, which is based on both capability and time.
The results show that the time is three times worst for the best participant and about ten times worst for the average
of participants.
The existence of a deviation in the required time to fulfill the tasks is logical, as participants need time to process
the information displayed on the screen each time an action changes the content before moving to the next action
or sequence of actions. User expertise and previous knowledge also play a role in the time users may require to
fulfilling the tasks. The issue also seems related to an unrealistic choice of times associated with the KLM interaction
operations: 0.2 seconds for clicks and key presses (K), 1.1 s. for pointing (P), and 0.4 s when switching between the
keyboard and the mouse (H). Additional tests are required to experimentally determine more adjusted values in the
context of semantic web data exploration, as the current values are the default ones for KLM and are mostly based
on accumulated experience in the context of desktop applications.

In any case, the results for the experienced participants are really good when considering just operator count
and the kinds of operations. This shows that despite the potential risks arising from using BESDUI to guide Rhi-
zomer development, the resulting user interfaces have an acceptable complexity for real users, especially for users
with experience with semantic web technologies. The only exception is Task 6, which is about looking for similar
instances.

Consequently, we can make the following proposals for user interface improvement, which are based on the
experience gained through the user tests and based on the qualitative input provided by participants based on their
experience performing the benchmark tasks:

- **Specific search for similarity**: Searching for similar instances has proved to be troublesome for almost all
  participants and a consequence of overfitting to BESDUI. In order to make it easier, some of them proposed
  including a link from an instance with a specific facet-based similarity search, in which the possible fields
  would be filled with the features of the given instance and the users should deselect features if they intend to
  include more instances as the result of the search.

- **All button**: Some of the participants proposed including a new button for selecting all possible elements of a
  given field. This way, they would not need to select them one on one.

- **Slider bar**: The slider bar in the faceted search (see left part of Figure 2) caused problems for most participants
  when adjusting the numbers in the slider bar to the ones in the corresponding task definition. On the one hand,
  it was hard to adjust them completely because the bar was too sensitive to adjust the exact number. On the
  other hand, the numbers selected by the participants sometimes automatically jumped to the number closest
  to them, as it was the number corresponding to the element closest to it in the instances that met the criteria.
  It was confusing for users as they see how a number specifically set by them suddenly changed. A possible
  improvement would be to make a small histogram to show the distribution of values within the range; with that
  information, it would be easier to maintain the original limits of the range. Another choice could be to allow
  specifying specific numbers for the lower and upper values in the selection using text boxes.

### 6. Conclusions and Future Work

The BESDUI benchmark has shown its usefulness, especially being the only benchmark for the evaluation of
structured data exploration tools. First of all, its analytical nature, which does not require the involvement of real
users, makes it feasible to apply the benchmark along the development of semantic data exploration tools while
guiding it.

Secondly, the proposed tasks to be evaluated have been defined independently, without the involvement of the
development team, and they focus on the typical information needs in a data exploration scenario. They have mo-
tivated the development of many of the RhizomerEye features that make it possible to satisfy those generic infor-
mation needs using its user interface, as detailed in Section 3. The result is a generic data exploration tool that is
applicable to other usage scenarios beyond the dataset and set of tasks defined by BESDUI because it is completely
data-driven.
However, there was the risk of overfitting RhizomerEye to BESDUI, that is, overloading the user interface to be able to complete the maximum amount of tasks with the minimal number of interaction steps based on the KLM model. To this end, we also conducted the study with real users reported in Section 4. It was possible to verify that real users are still able to perform the same tasks, using the same tool and data.

User interaction logs have been analyzed, as detailed in Section 4, to identify all interaction steps performed by the participants in the study while trying to complete the tasks, using the same approach that is used in BESDUI. This has facilitated comparing the results with real users to BESDUI’s analytical results. They show that, at least for users with experience with semantic web technologies, the number of interaction steps performed to complete the tasks is very similar to those determined analytically.

This rules out the overfitting risk, though it has shown that the values used to translate interaction steps to the time required to complete tasks are unrealistic as currently defined in BESDUI. They are inherited from the underlying Keystroke Level Model (KLM) [32], which is based on previous experiences with desktop applications that might not be valid in a Web application context and a data exploration scenario. It remains future work to further experiment with real users to better determine a mapping from KLM to time measures in this particular context.

As future work, it is foreseen to adapt the benchmark to remove detected ambiguity from the definition of the tasks and implement improvements in the user interface of RhizomerEye to solve detected problems, mainly the ones related to Task 6. It is also intended to extend the evaluation of RhizomerEye involving real users with additional sets of data and tasks, for instance, those used in the evaluation study based on the Quality in Use Model for Semantic Web Exploration Tools [20]. In that case, the dataset was not a synthetic one.

Finally, as it has been shown, RhizomerEye currently shows 58% Capability, which means there are still 5 BESDUI tasks that cannot be completed using this tool. Thus, continuing with the so far proven very fruitful approach of using BESDUI to guide RhizomerEye development, our plan is to add additional features that make it possible to complete more tasks. The focus is placed on making it possible to combine filters on more than one class, something that is possible with tools like Virtuoso Facets or the old version of Rhizomer using a feature called pivoting [10].

The most promising approach is to enable switching from one class faceted view to that for a related class through facets while keeping all the filters already applied to the source class.

TODO: alternative mechanism to support Task 6.

Acknowledgements

This work was partially supported by project "ANGRU: Applying kNowledge Graphs to research data ReUsability" with reference PID2020-117912RB-C22 and funded by MCIN/AEI/10.13039/501100011033. This research also benefits from funding from the Research Group program of the University of the Basque Country under contract GIU21/037.

References


Semantic Web

A.-S. Dadzie and E. Pietriga, Visualisation of Linked Data – Reprise,

E. Ikkala, E. Hyvönen, H. Rantala and M. Koho, Sampo-UI: A full stack JavaScript framework for developing semantic portal user interfaces,


R. García et al. / User Experience Benchmarking and Evaluation of a Semantic Knowledge Graph Exploration Tool


