

# USER-CENTERED DESIGN AND EVALUATION OF OVERVIEW COMPONENTS FOR SEMANTIC DATA EXPLORATION

*JOSEP MARIA BRUNETTI, ROBERTO GARCÍA  
UNIVERSITAT DE LLEIDA, LLEIDA, SPAIN  
{ JOSEPMBRUNETTI, RGARCIA }@DIEI.UDL.CAT*

## Structure Abstract

**Purpose.** The growing volumes of semantic data available in the Web result in the need for handling the Information Overload phenomenon. The potential of this amount of data is enormous but in most cases it is very difficult for users to visualize, explore and use this data, especially for lay-users without experience with Semantic Web technologies.

**Design/methodology/approach.** The Visual Information-Seeking Mantra “Overview first, zoom and filter, then details-on-demand” proposed by Shneiderman describes how data should be presented in different stages to achieve an effective exploration. The overview is the first user task when dealing with a dataset. The objective is that the user is capable of getting an idea about the overall structure of the dataset. Different Information Architecture (IA) components supporting the overview tasks have been developed, so they are automatically generated from semantic data, and evaluated with end-users.

**Findings.** The chosen IA components are well known to Web users, as they are present in most web pages: navigation bars, site maps and site indexes. We complement them with Treemaps, a visualization technique for displaying hierarchical data. These components have been developed following an iterative User-Centered Design methodology. Evaluations with end-users have shown that they get easily used to them despite the fact that they are generated automatically from structured data, without requiring knowledge about the underlying semantic technologies, and that the different overview components complement each other as they focus on different information search needs.

**Originality/value.** Obtaining semantic datasets overviews cannot be easily done with the current semantic web browsers. Overviews become difficult to achieve with large heterogeneous datasets, which is typical in the Semantic Web, because traditional Information Architecture techniques do not easily scale to large datasets. This can be a serious limitation when exploring a dataset for the first time, especially for lay-users. Our proposal is to reuse and adapt existing Information Architecture (IA) components to provide this overview to users and show that they can be generated automatically from the thesaurus and ontologies that structure semantic data while providing a comparable user experience to traditional web sites.

**Keywords:** Semantic Web, Linked Data, Search, Exploration, Human-Computer Interaction.

**Article classification:** Research Paper.

# 1 INTRODUCTION

The growing volumes of semantic data available in the Web result in the need for handling the Information Overload phenomenon (Thomas 2007). The potential of this vast amount of data is enormous but in most cases it is very difficult and cumbersome for users to visualize, explore and use this data, especially for lay-users (Dadzie & Rowe 2011) without experience with Semantic Web technologies. From the end-user perspective, the available datasets are monolithic and opaque files, which usually can just be explored using complex semantic queries or complex user interfaces. Improving the cognition of large datasets remains a standing problem in the Semantic Web.

To interact with large datasets, the literature proposes to divide the task of exploring a dataset into different stages. Shneiderman (Shneiderman 1996) presents the Visual Information-Seeking Mantra: “Overview first, zoom and filter, then details-on-demand”. The Mantra describes the fundamental set of tasks for data analysis and how data should be presented to users to achieve an effective exploration. It is presented as a guideline for how to design visual interfaces, where the first stage they present should be to gain an overview of the entire collection. Obtaining an overview is an important first step in the exploration of data.

Greene et al. (Greene et al. 1999) argued that a good overview “provides users with an immediate appreciation for the size and extent of the collection of objects the overview represents, how objects in the collection relate to each other, and importantly, what kind of objects are not in the collection”. Overviews provide a general context for understanding the available information and can be used as starting point for navigation. Despite visualizing and interacting with Linked Data is an issue that has been recognized from the beginning of the Semantic Web, cf. e.g. (Geroimenko & Chen 2002), users have not embraced it yet (Shadbolt et al. 2006). This is due in part to the fact that users find it difficult to use, even for advanced users with experience in Semantic Web technologies (Heath et al. 2006).

Semantic Web browsers should provide mechanisms that allow users to navigate the information effectively and easily. As datasets grow bigger and more complex, it becomes essential to have a compact representation of the information space. However, obtaining this overview cannot be easily done with the current semantic web browsers. Most of existing browsers assume the end user will start browsing from a specific URI, but most of end users do not even know what a URI means. They need an exploration starting point. Moreover, these browsers only provide access to single (but detailed) resources sequentially, which is very slow and easily saturates the user’s working memory. There is little or no support to obtain overview information quickly and easily at the beginning of the exploration of a new dataset. This can be a serious limitation when exploring a dataset for the first time, especially for lay-users.

Our proposal is to draw from the experience accumulated in the Information Architecture (IA) domain (Morville & Rosenfeld 2006) as well as to reuse and adapt existing IA components to provide this overview to users. Such IA components are well known to Web users, as they are present in most web pages: navigation bars, site maps and site indexes. We complement these components with treemaps (Shneiderman 1992), a visualization technique for displaying hierarchical data.

This approach was applied in Rhizomer<sup>1</sup>, a tool capable of publishing Semantic Web datasets while facilitating user awareness of the published content. These components have been evaluated with lay-users as part of a User-Centered Design development process (González et al. 2013). Our findings can be used as guidelines for designers of semantic web tools. They can decide which components to use depending on the concrete tasks their users need to solve and the characteristics of their datasets.

The remainder of this paper is organized as follows: Section 2 discusses related work. Section 3 introduces our approach, based on the Information Architecture domain. It also presents the development methodology and defines our target users. Section 4 describes the notion of overview in the context of Semantic Web datasets. Section 5 introduces the overview components implemented. Section 6 presents their evaluation with users performing real tasks and proposals to improve these components. Finally, Section 7 contains conclusions and future work.

## 2 RELATED WORK

Related work can be roughly classified into tools supporting Linked Data visualization and exploration as well as approaches for creating overviews of large datasets.

### 2.1 LINKED DATA VISUALIZATION AND EXPLORATION

Exploring and visualizing Linked Data is a problem that has been addressed by several projects. Dadzie et al. (Dadzie & Rowe 2011) analyzed some of those projects concluding that most of them do not provide overviews on the data. Moreover, most of them are designed only for tech-users, which makes even more complicated to understand what the data is about.

Semantic Web browsers such as Disco (Bojars et al. 2007), Tabulator (Berners-Lee et al. 2006) or Explorator (Araújo et al. 2009) allow users to navigate the graph structures and usually display property-value pairs in tables. They provide a view of a subject, or a set of subjects and their properties, but not any additional support getting a broader view of the dataset being explored.

Graph-based tools such as Fenfire (Hastrup et al. 2008), RDF-Gravity<sup>2</sup>, IsaViz<sup>3</sup> provide node-link visualizations of the datasets and the relationships between them. Although this visual approach can help obtaining a better understanding of the data structure in some cases graph visualization does not scale well to large datasets (Frasincar et al. 2005). Sometimes the result is a complex graph difficult to manage and understand (Karger & Schraefel 2006).

Other tools provide visual overviews by pointing resources on a map with different icons depending on their type, e.g. DBpedia Mobile (Becker & Bizer 2009) or LinkedGeoData browser (Stadler et al. 2011). However, they are restricted to visualizing and browsing concrete domains, in this case spatial data.

To summarize, none of the existing tools provide generic and text-based overviews for RDF data. Moreover, most of them make it difficult for non-technical users to explore linked data or they are restricted to concrete domains. Consequently, it is still difficult for end users to obtain an overview of

---

<sup>1</sup> <http://rhizomik.net/rhizomer/>

<sup>2</sup> <http://semweb.salzburgresearch.at/apps/rdf-gravity>

<sup>3</sup> <http://www.w3.org/2001/11/IsaViz>

datasets, comprehend what kind of structures and resources are available, what properties resources typically have and how they are mostly related with each other.

## 2.2 TECHNIQUES FOR VISUALIZING AND INTERACTING WITH LARGE-SCALE DATASETS

In the Information Visualization domain we can find several useful techniques for visualization of hierarchically structured datasets (Card et al. 1999). The techniques developed for this challenge can be mainly classified in node-link diagrams and space-filling visualizations. The traditional node-link representations such as SpaceTree (Plaisant et al. 2002) or Cone Trees (Robertson et al. 1991) combine the conventional layout of trees with zooming interaction. However, they make poor use of the available display space and leave the root side of the tree completely empty. They do not give an effective overview of large hierarchies and nodes need to be expanded manually. When data gets larger the screen quickly becomes overcrowded with nodes and their children.

On the other hand, space-filling techniques such as Treemaps (Shneiderman 1992) make a better use of the space by partitioning it into a collection of geometric shapes (i.e. rectangles) representing the tree structure. Some evaluations and comparisons have been performed on hierarchy visualization systems. Kobsa evaluated six visualization techniques with hierarchical data to compare their efficacy and user satisfaction with them (Kobsa 2004). Barlow et al. compared four different visualizations to evaluate the users' ability to understand the topology of the hierarchy and comparisons of node size (Barlow & Neville 2001). Their results show that each technique has its strengths and weaknesses depending on the exploration tasks and dataset characteristics.

However, these studies have mainly focused on users performing tasks related with understanding the topology and navigating through the hierarchy. None of them has performed an analysis of whether these visualization systems can be used to perform real end-users tasks and whether they provide good overviews of datasets.

## 3 APPROACH

Our starting point is the fundamental set of tasks for data analysis proposed by Shneiderman (1996). In the following, we present each task associated with the chosen interaction pattern for web design (Kalbach 2007):

- **Overview:** to get a full view of the entire collection. We propose to apply the Global Navigation interaction pattern<sup>4</sup> or the Directory Navigation pattern<sup>5</sup>.
- **Zoom & Filter:** to select items of interest and filter out uninteresting items through exploratory search. The proposal is the Faceted Navigation pattern<sup>6</sup>.
- **Details:** after filtering resources, the user can view detailed information about them. The proposal is the Details on Demand pattern<sup>7</sup>.
- **History:** to keep a history of actions and support undo and replay actions. We propose to apply the Breadcrumb Navigation pattern<sup>8</sup>.

---

<sup>4</sup> <http://www.welie.com/patterns/showPattern.php?patternID=main-navigation>

<sup>5</sup> <http://www.welie.com/patterns/showPattern.php?patternID=directory>

<sup>6</sup> <http://www.welie.com/patterns/showPattern.php?patternID=faceted-navigation>

<sup>7</sup> <http://www.welie.com/patterns/showPattern.php?patternID=details-on-demand>

In this paper we focus on the Overview task, which should be the first stage for data analysis. Our proposal is to elaborate the selected interaction patterns to perform this task in the context of semantic data. We have chosen these patterns because they are simple and very common so users are very comfortable using them. They are part of the “culture” about how information is presented in the Web so they can be easily learned. Although they look like the common ones, these ones should be capable of giving access to the richer semantic data they are built on top of.

The Global Navigation pattern proposes to design a persistent set of links that enables users to get to the key areas or functions of a website. These links should be visible to users from every page, so they can move directly from one section to another. On the other hand, the Directory Navigation pattern proposes to organize the items or pages into several groups. This pattern should be implemented when there are a large number of items, so users can select and focus on an item out of a large set.

Once users have obtained an overview of the data they can select those kinds of items they are interested in. Then, exploratory search can be used to navigate through different instances and their properties. However, the other tasks are out of the scope of this paper and the reader can find more information about how we implemented them in (Brunetti et al. 2013).

### 3.1 INFORMATION ARCHITECTURE

To implement the Global Navigation and Directory Navigation patterns, our proposal is to draw from the experience accumulated in the Information Architecture (IA) domain (Morville & Rosenfeld 2006) and adapt existing IA components to the context of semantic data.

Information Architecture (IA) is the art and science of organizing information. In the context of the World Wide Web, Information Architecture is the discipline that organizes and labels the information on websites. This discipline includes analyzing the contents, organizing web pages and designing the navigation systems. We have explored the most appropriate Information Architecture components to implement the Global Navigation and Directory patterns. The objective is to adapt existing IA components to provide this overview to users.

The challenge of structuring and presenting semantic data to end-users can be addressed with the experience accumulated in the Information Architecture domain. Information Architecture focuses its efforts in this problem, especially in complex systems and situations with large amounts of information. A good IA can improve the quality of a website and users can find easier the information they are looking for.

Traditionally, User-Centered Design techniques like Card Sorting (Spencer 2009) are used to develop the Information Architecture of websites. However, this technique requires much time from developers and most of this effort is wasted as soon as the content structure is established. Then, the Information Architecture becomes something static. If new kinds of items are introduced or a part of the content becomes more relevant, the Card Sorting should be repeated, at least in part.

In the case of web sites build on top of semantic data we have the opportunity to automate part of the process of generation and maintenance of the information architecture. The RDF data model enables the use of automatic tools to process it in a more sophisticated way. This is possible because semantic data is structured by thesauri and ontologies, which hierarchically organize the kinds of things described in the

---

<sup>8</sup> <http://www.welie.com/patterns/showPattern.php?patternID=crumbs>

dataset. They specify all the classes or concepts but also which entities belong to a certain class or are related to a certain concept.

### 3.2 USER PROFILES

Our aim is to make it possible that lay users, not just Semantic Web technologies experts, can reach the Semantic Web. Using the Personas approach (Garrett 2010) (Pruitt & Adlin 2006), we can illustrate the target audience as shown in Table 1. Personas are also useful to elaborate on the associated scenarios and derive contextualized user tasks. In our case, we will use Christina Warren and Michael Harper personas to motivate the set of user tasks related with DBPedia valuated during user tests in Section 6.



<p><b>Christina Warren</b> is a 23 years old <b>journalist</b> who has recently finished her studies and is currently in charge of the <b>Films section of an online journal</b>. She likes to write about <b>curious facts</b> like “who appears most in films where Woody Allen is both the director and an actor”. However, these kinds of things are really <b>difficult to find out using</b> resources like <b>Wikipedia or IMDb</b>.</p>	
	<p><b>Michael Harper</b> is a 30 years old <b>freelance developer</b> that creates and commercialize mobile applications using online application stores. He works mainly in solo projects and without any financial support. He is currently developing a <b>mobile application</b> that supports <b>bird watching</b> and as a way to <b>reduce development costs</b> to a minimum he is trying to <b>reuse</b> as much as possible available <b>data about bird species, habitats, etc.</b></p>

Table 1. Personas illustrating the intended users

### 3.3 METHODOLOGY

Rhizomer’s development methodology is based on the MPIu+a development process (Granollers 2003) and the RITE Method (Medlock et al. 2002) (Rapid Iterative Testing and Evaluation). MPIu+a is a development framework for interactive systems that integrates the discipline of Software Engineering with the basis of Human-Computer Interaction, Usability and Accessibility. MPIu+a has the user as the most important part of the process. User-Centered Design (UCD) is a type of design process for interactive systems focused on the users who will use the system. In UCD the user participates in all the stages of the design process. User requirements are considered from the beginning and included into the whole development cycle.

The RITE Method is an iterative development process based on short iterations. Each iteration, or reduced set of iterations, includes an evaluation of the Information Architecture components with a small group of users. This makes the evaluations differ from traditional ones mainly in the sense that much smaller groups of users are recruited for the tests, though tests are performed much more frequently. Consequently, results for individual test iterations are less significant from a statistical and quantitative point of view. The main results from just a testing session are basically qualitative and are used to guide the next development iteration. However, as many evaluation iterations are accumulated along the development process, it is possible to perform a quantitative analysis of the quality in use properties measured during each iteration and validate the evolution of the quality in use from a quantitative point of view. Moreover, the overall costs of the evaluation are significantly reduced.

## 4 OVERVIEW IN SEMANTIC WEB DATASETS

Overview is the first user task when dealing with a dataset. The objective is that the user is capable of getting an idea about the overall structure of the dataset. However, overviews become difficult to achieve with large heterogeneous datasets, which is typical in the Semantic Web. As datasets grow bigger and more complex, it becomes more difficult to understand the overall structure and to browse them efficiently (Hirsch et al. 2009). A common approach to obtain an overview and support the exploration of large datasets is to structure them hierarchically (Elmqvist & Fekete 2010). Hierarchies allow users to visualize different abstractions of the underlying data at different levels of detail. They can improve the navigation as they allow users to create a mental model of the content structure (van Ham & van Wijk 2002).

In the case of Semantic Web and Linked Data, this overview is usually about which are the main kinds of things in the dataset and how they are structured, i.e. the more instantiated classes and their hierarchical structure. It is also possible to obtain an overview from the point of view of the more common subjects the data is about and how they are structured, for instance as thesaurus.

### 4.1 OVERVIEW GENERATION

The overview is generated from the hierarchy structure of the dataset, i.e. its schema. For each class, we store information about the number of instances of the class, its URI, its labels and a list of its subclasses. All this information is retrieved using a set of SPARQL queries. Listing 1 shows the SPARQL query to obtain the root classes, i.e. those non-blank without a superclass different from owl:Thing or rdfs:Resource.

```
SELECT DISTINCT ?root
WHERE {
  ?root rdf:type ?class.
  FILTER (?class=owl:Class || ?class=rdfs:Class)
  OPTIONAL {
    ?root rdfs:subClassOf ?super .
    FILTER (?root!=?super && ?super!=owl:Thing && ?super!=rdfs:Resource && !isBlank(?super))
  }
  FILTER (!bound(?super) && isURI(?root) &&
    !isBlank(?root) && ?root!=owl:Thing )
}
```

Listing 1: SPARQL query to obtain the root classes

Listing 2 presents the SPARQL query to get direct subclasses for a given class marked with the placeholder “%classURI%”, i.e. there is not an intermediary class between them in the hierarchy.

```
SELECT DISTINCT ?sub
WHERE {
  ?sub rdfs:subClassOf %classURI% .
  OPTIONAL {
    ?sub rdfs:subClassOf ?sub2 .
    ?sub2 rdfs:subClassOf %classURI% .
    FILTER (?sub!=?sub2 && ?sub2!=<%1$s>
      && !isBlank(?sub2))
  }
  FILTER (!bound(?sub2))
}
```

Listing 2: SPARQL query to obtain direct subclasses for a given class

Finally, Listing 3 shows the SPARQL query to obtain the number the number of instances of each instantiated class.

```
SELECT ?class COUNT(?x)
WHERE {
  ?x a ?class
} GROUP BY ?class
```

Listing 3: SPARQL query to count the number of instances of each instantiated class

Once the class hierarchy is generated, we store it in a local file so it can be reused by the overview components. If the dataset is modified, i.e. some resources are added or removed, this file is updated. We use the vocabulary (Alexander & Hausenblas 2009) to describe the dataset, its class hierarchy and the number of instances of each class. The property is used to provide descriptions of parts of the dataset, i.e. subsets. The property expresses the number of distinct subjects (number of instances) that occur in the dataset or subset. By using these properties, each class-based partition describes a subset of instances that belong to a particular class. The file also provides metadata about the dataset that can be reused by other applications and users. Listing 4 shows a part of the VoID description for the DBPedia dataset.

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix void: <http://rdfs.org/ns/void#> .

...

<http://rhizomik.net/dataset> void:classPartition <http://dbpedia.org/ontology/Species> .

<http://dbpedia.org/ontology/Species> rdfs:label "Species" ;
  void:classPartition <http://dbpedia.org/ontology/Archaea>,
    <http://dbpedia.org/ontology/Bacteria>,
    <http://dbpedia.org/ontology/Eukaryote> ;
  void:distinctSubjects 202339 .

<http://dbpedia.org/ontology/Archaea> rdfs:label "Archaea" ;
  void:distinctSubjects 171 .

<http://dbpedia.org/ontology/Bacteria> rdfs:label "Bacteria" ;
  void:distinctSubjects 328 .

<http://dbpedia.org/ontology/Eukaryote> rdfs:label "Eukaryote" ;
  void:distinctSubjects 199085 .

...
```

Listing 4: Part of the VoID description for the DBPedia dataset

## 5 OVERVIEW COMPONENTS

The overview components are created from the hierarchy structure generated. For datasets without a schema it is also possible to use these components but their structure will be flat. Next, we describe the overview components that have been implemented.

### 5.1 NAVIGATION MENUS

Navigation menus, in the case of websites and as shown in Fig. 1, let users navigate through different sections and pages of the site. They tend to be the only consistent navigation element, being present on every page of the site. The objective is to generate a global navigation menu that takes into account all



the classes considered in a dataset but also how they are instantiated. Consequently, if there are few instances of some classes or they are not instantiated at all, they should be less relevant in the menu bar. On the contrary, classes that do have a lot of instances should be shown prominently in the menu bar. This approach makes it possible to show the user the navigation bar that best fits the data in the dataset at that particular moment.

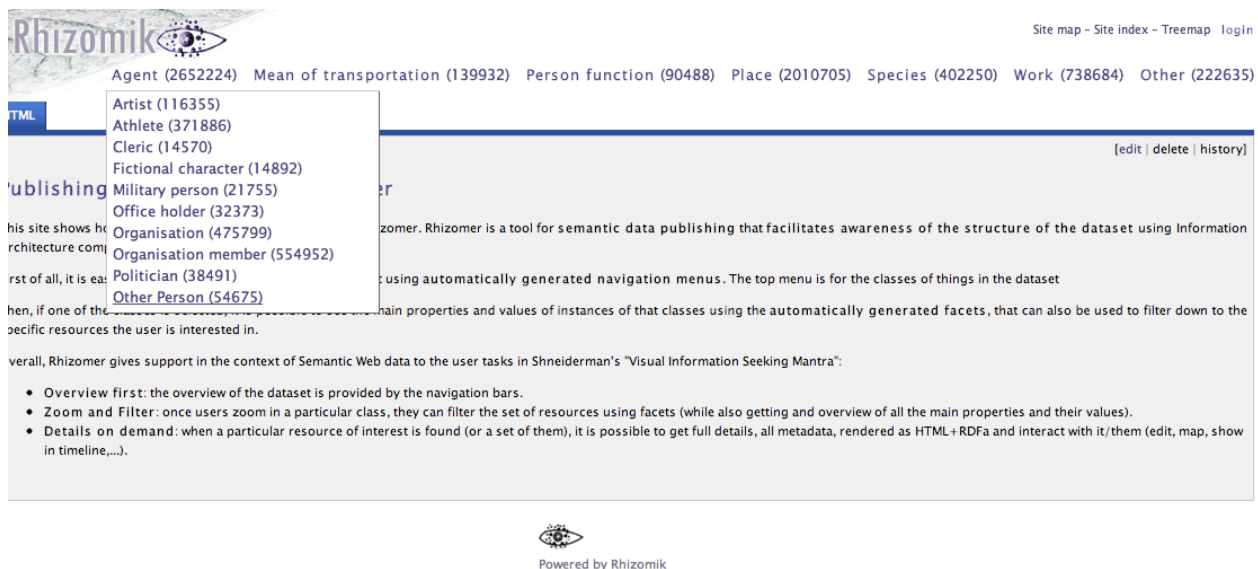


Fig. 1. Navigation menus

This component can generate both global and local menus, i.e. a menu for the whole dataset or for a subset of it. The site administrator can also configure some parameters such as the number of levels in the menu, the number of items of each level, the order of items and a list of classes or namespaces to omit. According to these parameters, this component generates the menu applying a recursive algorithm that mainly performs two operations:

- Split those classes with a large amount of instances in subclasses.
- Group those classes with few instances in a superclass.

These operations are recursively performed until the menu is completed. Figure 2 illustrates the process of generating the navigation menu for a subset of DBpedia, with 7 elements in the first level. In the original hierarchy there are only 3 classes in the first level. Therefore, there are 4 free spots in the menu. To cover these free spots, the algorithm identifies which classes are appropriate to divide, taking into account their number of instances and their number of subclasses.

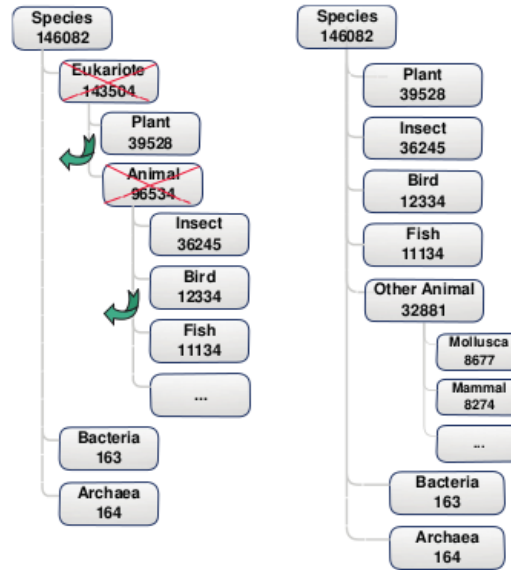


Fig. 2. Generating a navigation submenu for DBpedia Species with 7 slots (left original, right result)

First, the *Eukariote* class is removed and its subclasses, *Plant* and *Animal*, are moved up to a higher level in the hierarchy. After this step, the navigation menu contains 4 elements: *Plant*, *Animal*, *Bacteria* and *Archaea*. From here, the algorithm is applied recursively until the menu is completely generated. In the next step, the *Animal* class is chosen and divided. However, in this case, there is not space for all its subclasses in the first level of the menu. For this reason, the subclasses with a higher number of instances are moved up to the main level of the menu while the rest of subclasses are grouped inside *Other Animal*. This algorithm is applied recursively until the menu is completely generated.

## 5.2 HTML SITE MAPS

Navigation menus are quite effective because lay-users are comfortable with them, most website feature them and they are used to interacting with them. However, they just provide an overview of the most frequent classes, those more instantiated. In order to gain a more detailed overview, web sites usually apply the Directory Navigation pattern through different sorts of sitemaps. Site maps act as a navigation aid by providing an overview of the site's content at a first glance. A site map is a web page that lists all the pages of a website, normally organized hierarchically.

In the case of large sites, instead of containing links to all the pages, they can list the main pages (e.g. categories) of the site. When the site contains many levels in the structure and many elements on each level the site map functions as a navigation alternative to navigation menus.

We have implemented two different versions of the sitemap and users can switch between them. The first one is related with the structure of navigation menus, which has been generated from the original hierarchy. It complements the main site navigation and users can find there options that were are not directly available from navigation bars. It can be seen as a summarized version of the complete hierarchy structure. The second one reflects the original hierarchy of the dataset. It is showed as a tree with multiple levels and the users can expand it. Figure 3 shows the two site map versions.



Fig. 3. Site map versions

### 5.3 TREEMAPS

One of the most popular methods for displaying hierarchical data is Treemaps (Shneiderman 1992). They produce space-efficient overviews of hierarchically structured datasets. They visualize hierarchical relationships of the data elements but also display quantitative information regarding the distribution of elements by their size. Treemaps use a rectangle to show the tree root and its children. Each child has a size proportional to the cumulative size of its descendants. They are a good method to display the size of each node in a hierarchy, i.e. its number of instances.

We have implemented the Treemap visualization following recommended design issues (Turo & Johnson 1992) and using the *JavaScript Infovis Toolkit*<sup>9</sup>. The generated Treemaps, like the one shown in Figur 4, show two hierarchy levels but allow users to zoom in. We used squarified Treemaps as the tilling algorithm. It produces rectangles with low aspect ratio, which makes it easier to compare their sizes. Different colors are used for each class and its subclasses. Label size is proportional to the number of instances of each node.

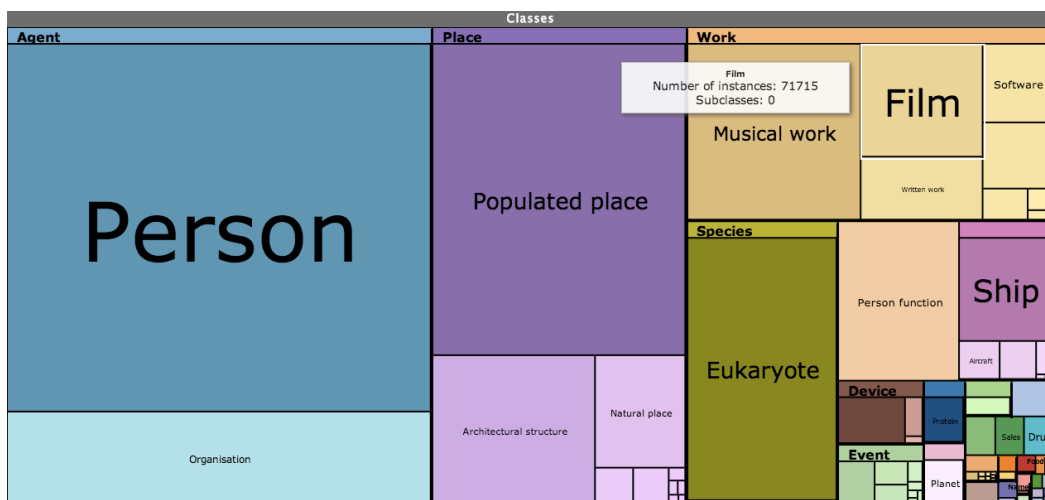


Fig. 4. Treemap

<sup>9</sup> <http://thejit.org/>

## 5.4 SITE INDEX

A site index is a navigational and informational tool that lists all the pages or categories alphabetically. Users can spend a lot of time looking for a concrete page through the site map. While a site map provides a general view of the overall site contents, an A-Z index provides access to particular content. An alphabetical list, like the one shown in Figure 5, can better suit users' mental model when they are searching for a specific page.

However, while site maps can give users context, site indexes provide no context. Non-related categories appear in the site index without giving users any additional information. Therefore, we have implemented the site index so that it provides also context information of each class. When the user moves the pointer over an element an overlay appears showing its parent and its subclasses.



Fig. 5. Site index

## 6 EVALUATION

The goal of the evaluation was to determine whether solving tasks with the 4 different components differs with respect to task completion times, response accuracy and user satisfaction. Comparisons between the different components can provide valuable information about the effectiveness and usability of these systems depending on the use case. We do not aim to face the systems against each other but to provide a basis for design recommendations and guidelines.

### 6.1 EXPERIMENTAL DESIGN

The tests were conducted at the UsabiliLAB<sup>10</sup> where sessions were registered using Morae Recorder together with Morae Observer<sup>11</sup> to analyze test data. The software collected the task time and the effectiveness (completed, not completed and partially completed). Users were presented Rhizomer with the DBPedia dataset and the four components to use. They were encouraged to use the think aloud protocol (Lewis 1982) to express their opinion and thoughts on the system while performing tasks.

<sup>10</sup> <http://griho.udl.cat/about/infrastructure.html>

<sup>11</sup> <http://www.techsmith.com/morae.html>

We asked 10 participants to participate in the study (6 males, 4 females), having a unique profile characterized by good knowledge of information technology; they work regularly with computers and use the Internet. None of them had expertise in Semantic Web technologies or had previously used the DBPedia. The chosen tasks comprise the common information needs when users navigate and want to find something in a web site (Morville & Rosenfeld 2006): known-item seeking, exhaustive research, re-finding. The items to find slightly vary for each task and component to reduce the effect of learning<sup>12</sup>:

1. **Known-item seeking:** find one node for different levels of the hierarchy using navigation menus (1), site map (2), Treemap (3) and site index (4).
  - **Task 1:** find 1st level item - Work (1), Place (2), Event (3), Species (4).
  - **Task 2:** find 2nd level item - Monument (1), Insect (2), Aircraft (3), Film festival (4).
  - **Task 3:** find 3rd level item - Reptile (1), Sports team (2), Hospital (3), Golf Player (4).
2. **Exhaustive research:**
  - **Task 4:** arrange from bigger to smaller in number of resources using navigation menus (1), site map (2) and Treemap (3).
    - Place, Agent and Species (1).
    - Work, Place and Mean of transportation (2).
    - Eukaryote, Person and Musical Work (3).
  - **Task 5:** topology understanding. Using navigation menus (1), site map (2) and Treemap (3) find subtypes of:
    - Mean of Transportation (1), Species (2), Work (3).
3. **Re-finding:** find previously visited items using navigation menus (1), site map (2), Treemap (3) and site index (4).
  - **Task 6:** find Monument (1), Insect (2), Aircraft (3), Film festival (4).

In total, users had to perform 6 tasks using navigation menus, 6 with the site map, 6 with the Treemap and 4 using the site index. 12 tasks belong to known-item seeking, 6 to exhaustive research and 4 to re-finding. We ran the evaluation as a within-subject design, where each participant performed all tasks using all four components, except for tasks 4 and 5, which was not performed using the site index because it does not provide the context information needed to perform these kind of tasks. Users were given a maximum of 2 minutes to perform each task.

## 6.2 RESULTS AND DISCUSSION

Figure 6 shows the average task completion times. The results show that user's performance depended on a combination of the task and the component used. Overall, all tasks were correctly completed except task 3, which is explained later. The results are discussed in the next subsections.

### 6.2.1 KNOWN-ITEM FINDING

Navigation menus were the fastest component to find large known-items located in the first level of the hierarchy (e.g. Work). Users could see the most important items at a single glance in a horizontal line. However, their efficiency decreases when users have to find items located deeper in the hierarchy (e.g. Monument or Reptile).

With the site map, users could easily find nodes located in the first or second level of the hierarchy (e.g. Place or Insect). However, the *summarized* site map only shows two levels of the hierarchy and users had

---

<sup>12</sup> The numbers between parentheses refer to the component to be used to perform the corresponding task.

to switch to the *full* site map to find items located deeper (e.g. Sports team). Only 2 users were able to find this option and were able to complete the task.

Something similar occurs with the Treemap. Users could easily find large nodes located in the first or second level of the hierarchy (e.g. Event) but it was more difficult for them to find smaller nodes (e.g. Aircraft) because their label was not visible until they zoomed-in.

It was difficult for users to find nodes located deep in the hierarchy (e.g. Aircraft, Hospital) using the Treemap, site map or navigation menus. They needed to understand how the dataset was hierarchically structured and guess where the item could be located. In these cases, the site index was the fastest component because users did not need to understand the site structure. They only needed to locate the item in the alphabetical index.

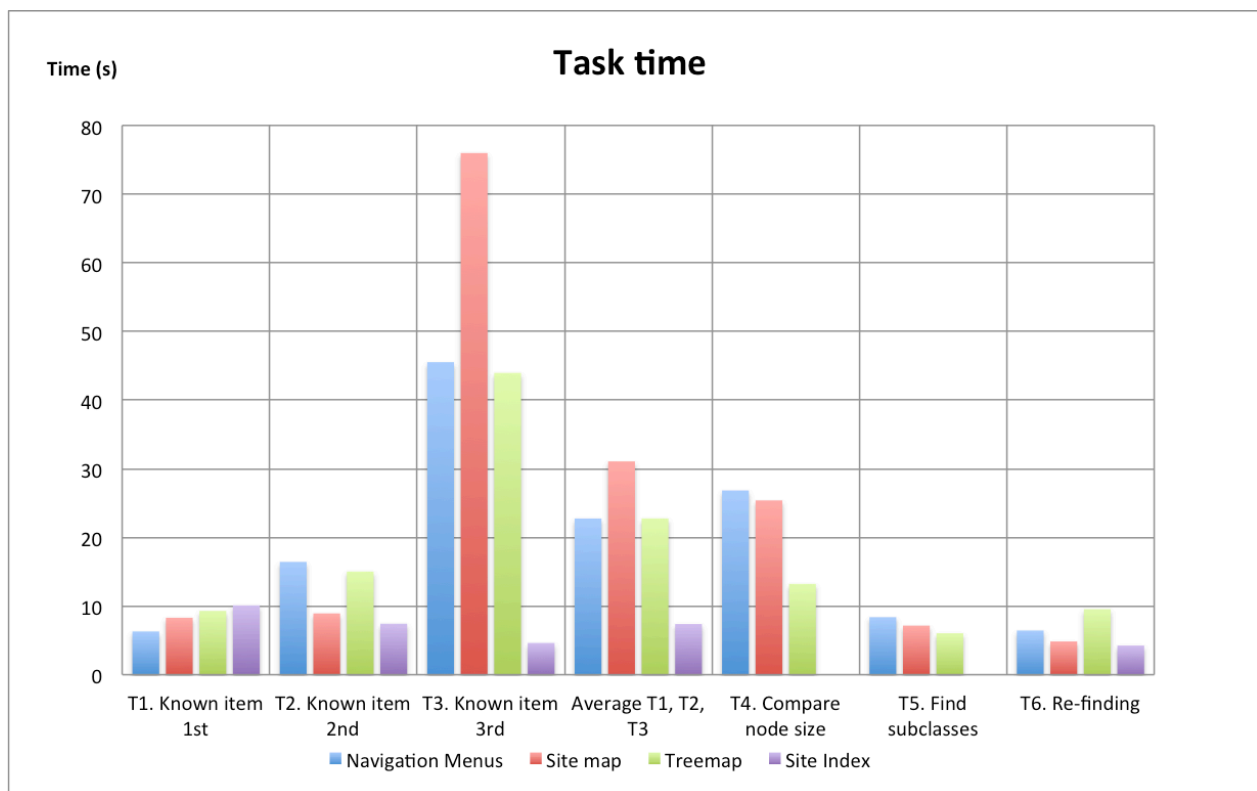


Fig. 6. Average task completion time

### 6.2.2 EXHAUSTIVE RESEARCH

The Treemap was the fastest component to compare node sizes. Rectangle areas allowed users to easily compare different nodes. On the other hand, using navigation menus or the site map users had to manually compare the number of instances. Several users pointed out that it was difficult because thousands were not separated by dots.

The Treemap was also the best component to understand the topology and to find a class and its sub-classes. Different colors allow easily identifying groups of related nodes. However, in this case there is not a big difference compared to navigation menus and the site map.

### 6.2.3 RE-FINDING

Users found previously visited nodes (T6) faster than the first time (T2) with all components. The site map was significant faster than navigation menus. The Treemap was the worst component to perform this task. The visual representation of nodes did not help much users to remember where was located the item to find.

### 6.3 USER SATISFACTION

After finishing the tasks with each component, we gave users a written questionnaire to evaluate their confidence, satisfaction or frustration with it. Users were asked to rate the following statements from 1 to 7, being 1 the most negative and 7 the most positive. Table 2 shows the questions and user ratings.

Question	NM	SM	TR	SI
Q1. The information showed on this component is helpful	6.2	6.2	5.7	6.4
Q2. This component is easy to use	6.4	6.2	5.7	6.6
Q3. Using this component, it was easy to find the information I was looking for	6.3	5.3	5.5	7
Q4. From this component, I understand the structure of the website	6.3	6.6	5.4	4.6
Q5. From this component, I understand what content is available on the website	6.2	6.3	4.9	5.8
Average	6.28	6.12	5.44	6.08

Table 2. User satisfaction questionnaire for navigation menus (NM), site map (SM), Treemap (TR) and site index (SI)

Overall, users had a good impression of the overview components. Users prefer navigation menus, the site map and the site index instead of the Treemap. They are used to common web components more than to interactive visualization methods. Most of them had never seen a Treemap before and required some time to learn how to use it. There were no significant differences between traditional web components in terms of use preference. It is important to notice that the site index was considered the best component to find concrete information (Q3) but the worst to understand the site structure (Q4).

### 6.4 PROPOSAL

From test results and their analysis, we have elaborated the following proposals to improve the existing overview components.

#### 6.4.1 NAVIGATION MENUS

The proposal for navigation menus is to improve the algorithm that generates them. Right now, the main factor considered to generate the menus is the number of instances of each class. As a result, it is possible that some options in the first level of the generated menu do not have subclasses. For example, in our evaluation with DBpedia, the class appeared in the first level but did not have any subclasses. Then, we cannot use a submenu for this class and we waste space in the navigation menu that could be used to show more options. Meanwhile, other classes with fewer instances but with more subclasses do not appear in the first level. The proposal is to study how to generate the navigation menus considering both factors: the number of instances and the number of subclasses for each class.

#### 6.4.2 SITE MAP

As the results show, users did not understand the difference between the two site map versions and did not know how to switch between them. The first one reflected the original hierarchy of the dataset, displayed as a tree with multiple levels. The second one, related with the structure of navigation menus, was a summarized version with the main classes. Only two users were able to find how to switch between both

versions. In other studies regarding site map usability (Tedesco 2008), participants successfully used site maps that were not organized to reflect the site’s main navigation.

Moreover, these studies suggest that multi-column site maps work better because users need less scrolling to get an overview of the site’s structure. Therefore, our proposal is to keep only the full site map and display it using two columns. The full version allows reaching all the classes in the original hierarchy while the summarized version only shows the main classes.

### 6.4.3 TREEMAP

The main issue with Treemap was to find small nodes. In some cases, there was not enough space to show their labels inside small rectangles. Users could only see these labels when they moved the mouse pointer over the rectangles. Our proposal is to study how to group small nodes into a supernode. Then, when zooming into this supernode, it would be easier to locate small nodes. We also propose to complement the Treemap with a keyword search that could allow finding these nodes.

## 7 CONCLUSIONS AND FUTURE WORK

We presented four components that support the overview task proposed by Shneiderman in his visual seeking mantra. These components are useful for obtaining a broad view of the datasets, their main types and the relationships between them. The RDF data model being prevalent on the Data Web enables us to create these overviews in a generic and automatic way. A good overview provides users an appreciation of the collection of objects and can help achieving their information seeking goals. Rhizomer, with these overview components, allows users to explore datasets even if the publisher of the data does not provide any exploration or visualization means.

Our evaluation shows that each component has its strengths and weaknesses depending on the use case. Navigation menus are useful to find the main important classes but they can only show a limited number of options. Site maps provide a general top-down view of the overall site contents but users need to understand the site structure to find concrete items. Treemaps are useful to get an overview of the data, compare node sizes and identify groups of related items through different colors. However, it is very difficult to locate those nodes that have a small area. A site index provides easy access to particular content but does not give information about how a website is structured.

The previous findings are aligned with current knowledge about how these user interface components contribute to the users experience while seeking information in “classical” web sites. Consequently, from these similarities and the quantitative and qualitative results obtained during user testing, it has been shown that it is possible to obtain a comparable user experience, even for very large semantic datasets like DBPedia, using components automatically generated from the underlying data, ontologies and vocabularies.

Our future work focuses on improving these components and their usability, as described in Section 6.4. We will further study these components with other datasets to evaluate how they adapt to different datasets depending on their size and structure, and also how users perform similar tasks using them. This will also provide insights about the particularities of the user experience when exploring big datasets using automatically generated user interface components like the ones generated. This will also better motivate alternative mechanism for data exploration.



For instance, another possibility to create overviews is to apply clustering algorithms. Clustering is the process of classifying similar resources into groups called clusters. This could allow us to create navigation menus with a limited number of options that represent the whole dataset. Clustering could be also used to create overviews of datasets without a schema. Finally, we also plan to develop a search system. Some users asked for it and they even used the search option available in the web browser. A search system would allow users to easily find known-items without needing to know where it is located in the hierarchy.

## REFERENCES

- Alexander, K. & Hausenblas, M. (2009), Describing linked datasets - on the design and usage of void, the vocabulary of interlinked datasets, in 'Linked Data on the Web Workshop (LDOW'09)', in conjunction with 18th International World Wide Web Conference (WWW'09).
- Araújo, S. F. C., Schwabe, D. & Barbosa, S. D. J. (2009), Experimenting with explorer: a direct manipulation generic RDF browser and querying tool, in 'Workshop on Visual Interfaces to the Social and the Semantic Web (VISSW2009)'.
- Barlow, S. T. & Neville, P. (2001), A comparison of 2D visualizations of hierarchies, in 'INFOVIS', pp. 131–138.
- Becker, C. & Bizer, C. (2009), 'Exploring the geospatial semantic Web with DBpedia mobile', *Journal of Web Semantics: Science, Services and Agents on the World Wide Web* 7(4), 278–286.
- Berners-lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A. & Sheets, D. (2006), Tabulator: Exploring and analyzing linked data on the semantic web, in 'In Proceedings of the 3rd International Semantic Web User Interaction Workshop'.
- Bojars, U., Passant, A., Giasson, F. & Breslin, J. (2007), An architecture to discover and query decentralized RDF data, Vol. 248 of *CEUR Workshop Proceedings ISSN 1613-0073*.
- Brunetti, J., Garcia, R. & Auer, S. (2013), 'From Overview to Facets and Pivoting for Interactive Exploration of Semantic Web Data', *International Journal of Semantic Web and Information Systems* 9(1), 1–20.
- Card, S. K., Mackinlay, J. D. & Shneiderman, B. (1999), *Readings in Information Visualization: Using Vision to Think*, Academic Press, London.
- Dadzie, A.-S. & Rowe, M. (2011), 'Approaches to visualising linked data: A survey', *Semantic Web* 2(2), 89–124.
- Elmqvist, N. & Fekete, J.-D. (2010), 'Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines', *IEEE Trans. Vis. Comput. Graph.* 16(3), 439–454.
- Frasincar, F., Telea, R. & Jan Houben, G. (2005), Adapting graph visualization techniques for the visualization of RDF data, in 'Visualizing the Semantic Web, 2006', pp. 154–171.
- Garrett, J.J. (2010), *The Elements of User Experience: User-Centered Design for the Web and Beyond*, 2 ed., New Riders, Indianapolis, IN, United States.
- Geroimenko, V. & Chen, C., eds (2002), *Visualizing the Semantic Web*, Springer.
- Granollers, T. (2003), User centered design process model, integration of usability engineering and software engineering, in 'Proceedings of INTERACT 2003', pp. 673–675.
- González, J.L., García, R., Brunetti, J.M., Gil, R. & Gimeno, J.M. (2013), 'Using SWET-QUM to Compare the Quality in Use of Semantic Web Exploration Tools', *Journal of Universal Computer Science* 19(8), 1025–1046.
- Greene, S., Marchionini, G., Plaisant, C., Shneiderman, B. & computer Interaction Laboratory, H. (1999), 'Previews and overviews in digital libraries: Designing surrogates to support visual information seeking'.
- Hastrup, T., Cyganiak, R. & Bojars, U. (2008), 'Browsing linked data with Fenfire'.

- Heath, T., Domingue, J. & Shabajee, P. (2006), User interaction and uptake challenges to successfully deploying semantic web technologies, *in* 'Proc. 3rd International Semantic Web User Interaction Workshop'.
- Hirsch, C., Hosking, J. & Grundy, J. (2009), Interactive visualization tools for exploring the semantic graph of large knowledge spaces, *in* '1st Int'l Workshop on Visual Interfaces to the Social and the Semantic Web (VISSW 2009)', Sanibel Island, Florida.
- Kalbach, J. (2007), *Designing Web Navigation. Optimizing the User Experience*, O'Reilly & Associates, Inc., Sebastopol, CA, USA.
- Karger, D. & Schraefel, M. (2006), The pathetic fallacy of RDF. Position Paper for SWUI'06.
- Kobsa, A. (2004), 'User experiments with tree visualization systems'.
- Lewis, C. H. (1982), Using the "Thinking Aloud" Method In Cognitive Interface Design, Technical Report RC-9265, IBM.
- Medlock, M. C., Wixon, D., Terrano, M., Romero, R. L. & Fulton, B. (2002), Using the RITE method to improve products; a definition and a case study, *in* 'Usability Professionals Association'.
- Morville, P. & Rosenfeld, L. (2006), *Information Architecture for the World Wide Web*, 3rd ed., O'Reilly & Associates, Inc., Sebastopol, CA, USA.
- Plaisant, C., Grosjean, J. & Bederson, B. B. (2002), Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation, *in* 'Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)', INFOVIS'02, IEEE Computer Society, Washington, DC, USA, pp. 57-64.
- Pruitt, J. & Adlin, T. (2006) *The Persona Lifecycle: Keeping People in Mind Throughout Product Design*, Morgan Kaufmann, San Francisco, CA, USA.
- Robertson, G. G., Mackinlay, J. D. & Card, S. K. (1991), Cone Trees: animated 3D visualizations of hierarchical information, *in* 'Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology', CHI '91, ACM, New York, NY, USA, pp. 189-194.
- Shadbolt, N., Berners-Lee, T. & Hall, W. (2006), 'The semantic web revisited', *IEEE Intelligent Systems* 21(3), 96-101.
- Shneiderman, B. (1992), 'Tree visualization with tree-maps: 2-d space-filling approach', *ACM Trans. Graph.* 11(1), 92-99.
- Shneiderman, B. (1996), The eyes have it: A task by data type taxonomy for information visualizations, *in* 'IEEE Visual Languages', UMCP-CSD CS-TR-3665, College Park, Maryland 20742, U.S.A., pp. 336-343.
- Spencer, D. (2009), *Card Sorting: Designing Usable Categories*, Rosenfeld Media, LLC.
- Stadler, C., Lehmann, J., Höffner, K. & Auer, S. (2011), 'Linked-geodata: A core for a web of spatial open data', *Semantic Web Journal*.
- Tedesco, D. (2008), *Site Map Usability: 47 Design Guidelines Based on Usability Studies with People Using Site Maps*, Nielsen Norman Group.
- Thomas, J. (2007), 'Visual analytics: a grand challenge in science: turning information overload into the opportunity of the decade', *J. Comput. Sci. Coll.* 23(2), 5-6.
- Turo, D. & Johnson, B. (1992), 'Improving the visualization of hierarchies with Treemaps: Design issues and experimentation'.
- van Ham, F. & van Wijk, J. J. (2002), Beamtrees: Compact Visualization of Large Hierarchies, *in* 'Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)', Vol. 00, IEEE Computer Society, Los Alamitos, CA, USA.