Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea Triennale in Informatica per il Management

On the path of Exploratory Search Users
A Semantic Web tool

Tesi di Laurea in Ingegneria del Software

Relatore:
Chiar.mo Prof.
Paolo Ciancarini

Correlatore:
Dott.
Andrea Giovanni Nuzzolese

Presentata da:
Caroline Dos

Sessione I
Anno Accademico 2012-2013

Abstract

L'Exploratory Search, paradigma di ricerca basato sulle attività di scoperta e d'apprendimento, è stato per diverso tempo ignorato dai motori di ricerca tradizionali. Invece, è spesso dalle ricerche esplorative che nascono le idee più innovative. Le recenti tecnologie del Semantic Web forniscono le soluzioni che permettono d'implementare dei motori di ricerca capaci di accompagnare gli utenti impegnati in tale tipo di ricerca. Aemoo, motore di ricerca sul quale s'appoggia questa tesi ne è un esempio efficace. A partire da quest'ultimo e sempre con l'aiuto delle tecnologie del Web of Data, questo lavoro si propone di fornire una metodologia che permette di prendere in considerazione la singolarità del profilo di ciascun utente al fine di guidarlo nella sua ricerca esplorativa in modo personalizzato. Il criterio di personalizzazione che abbiamo scelto è comportamentale, ovvero basato sulle decisioni che l'utente prende ad ogni tappa che ritma il processo di ricerca. Implementando un prototipo, abbiamo potuto testare la validità di quest'approccio permettendo quindi all'utente di non essere più solo nel lungo e tortuoso cammino che porta alla conoscenza.

Contents

List of Figures

Chapter 1

Introduction

Also known as Digital Age or Computer Age, the Information Age is the period which begins in the last quarter of the 20th century and goes on until our days. This term does not mean obviously that information did not exist before, but insists in how the new technologies have allowed a certain democratization of the information access and how this has revolutionized the world economy. By means of Internet, one of the most famous and important discoveries of the 20th century, information reach us and nowadays have a central place in our life. Today, more than two billions of people are connected to the Internet and daily navigate on the World Wide Web. [1]. Each year the number of users grows exponentially.

The information access has created new customs and needs. As a case in point, a study conducted by ComScore[Com13] puts in evidence how 80% of Europeans Internet users regularly consult news or information sites. It is not necessary to report the success encountered by social networks like Facebook and LinkedIn or collaborative encyclopedias like Wikipedia, all instruments which create and exploit network effects.

A normal and predictable consequence of this exponential growth is the proliferation of contents and services available on the Web, and the users meet more and more difficulties to reach the right contents at a right time.

[1] Report available at http://www.internetworldstats.com/stats.htm

To avoid users to be hindered in their search task by this information overload (which has to remain a richness and not a failure), collections of tools and technologies based on non-traditional paradigms are needed. These instruments, to counteract the negative effect of the Web expansion, have to consider essential points like [Tva11]:

- the diversity of seeking activities, e. g. look up, learning, investigation;
- the diversity of users and their needs; [2]
- the navigation problem, i.e. , the lack of orientation of the user in the "hyperspace";
- the information overload.

As regards the first two points, traditional search engines have focused for years on only one sub-task of information seeking, namely the information retrieval based on the keyword paradigm, i.e. on the lookup activity. However a search often involves other activities with a more complex process, like the learning and the discovery of information, two recurring activities when an exploratory search is performed. A straight consequence of this traditional focusing on only one of the searching tasks is the lack of consideration of the variety of profiles of the users. Indeed, the researchers or the decision-makers will certainly use the data on the Web in a very particular way. The innovation which is at the beginning of their activities compels them to progress on knowledge spaces which have not been explored yet, where the risk to get lost in a mass of new data and then in a not fully mastered set of information is high.

[2] The current referencing strategies of the search engines, by being based on popularity criteria, the selection and the classification of the results which are returned to the users after they have made a keyword query do not take into account the diversity of the users (for the same query, the response is the same for all). On the contrary, the classification is based on the number of click for each page. Don't we run the risk in the end, considering that users usually visit only the first pages of result and the overload of information, to bring us toward a single-minded approach?

Taking into account this heterogeneity of needs, a new generation of search engines, based on the Semantic Web technologies and on the paradigm of Exploratory search has thus been created. By focusing on the satisfaction of a single class of users who mainly have recourse to discovery and learning activities, they allow, thanks to the use of the Semantic web technologies, to bypass the problem of overload information by presenting only the data corresponding the most to the searched topic, and to give the user the means to employ it and to orientate him in a knowledge space he does not know yet.

How the technologies of the Semantic Web promote a browsing based on discovery and learning? How can they be employed in order to guide the user throughout his search? These are the questions we will try to answer to in this work.

In chapter 2, we will introduce the paradigm of the Exploratory Search, which is the background context of our study. In Chapter 3, we will briefly approach the technologies of the Semantic Web allowing the effective implementation of these search engines. In Chapter 4, we will present Aemoo, a search engine which shows the application of these technologies in the domain of the Exploratory Search. Finally, in chapter 5, we will deal with the more specific question concerning the orientation of the user inside the space of information, trying to consider the singularity of each user by means of a methodology that we will describe and that we have had the chance to implement and test on Aemoo.

Chapter 2

Exploratory search

The Exploratory Search is not a new paradigm of search, but it has not been taken in consideration by traditional search engines. In this chapter we present this concept, beginning in section 1 to place the Exploratory Search task in the context of the more general Information Seeking process or on the contrary in the context of the more specific search task which is the Information retrieval. In section 2 we expose the classical motivations of the users who perform an exploratory search. Finally, in section 3 we will study the behavior of these users during a search.

2.1 Information seeking, Information retrieval and Exploratory Search

Information Search is a field which had benefited from an amazing scientist's interest since the web creation. Indeed, by the informational nature of the resources published on the Internet, a resource really exists on the Web only from the moment it can be found.

It is the Information Retrieval, which leans on what we call the query-response paradigm or even the Lookup model, adopted since the beginnings of the web, and which still prevails nowadays in the most popular search engines, which used to concentrate, at the begin, all the research community's

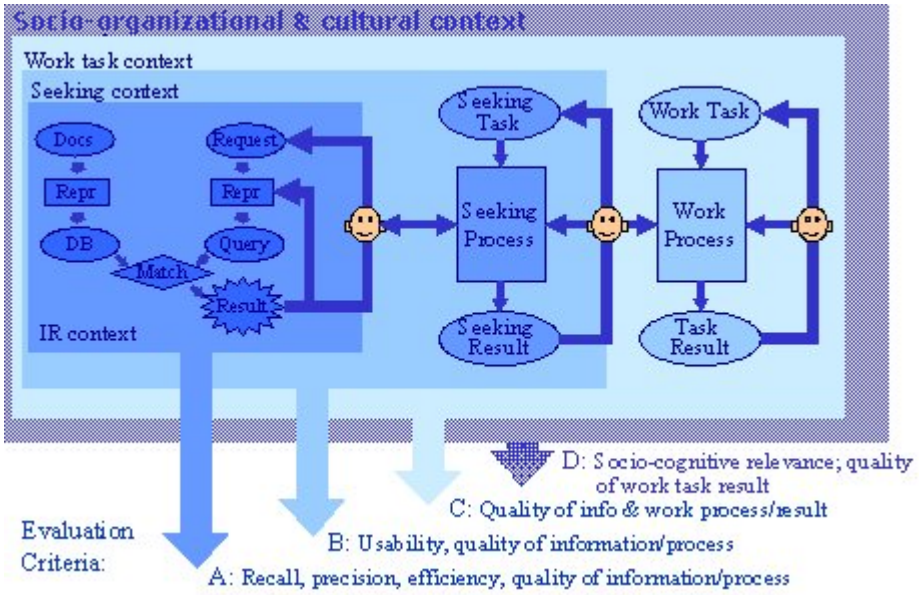


Figure 2.1: Information Seeking context[JI04]

efforts.

The Information Retrieval process starts from a short query, typically expressed as a question in natural language, which has to be well structured and whom response does exist. Then, in a reasonable time frame, a list of documents possibly comprehending the right answer is returned. In this way, for instance, we can easily find the date of birth of Victor Hugo, the height of Pisa tower, or the last Tarantino movie.

However, the research problem is often more complex than a simple extraction of information.

The model by Ingwersen and Javelin[JI04] replaces in a concise and elegant way (fig.2.1) the context in which we are brought to make Information Retrieval: it allows to understand, first of all, that Information Retrieval is one of the means that allow to solve a problem which is situated ahead.

And so, this model allows to avoid confusion between "Information Seeking" and "Information Retrieval" which, even if are linked between them, (an activity of Information Seeking can ask one or more activities of Information Retrieval), remain two different concepts.

As a guideline for illustrating the concepts behind the model of Innerspring and Jarvelin as shown on fig. 2.1, we will use a single example: the purchase of a real estate.

According to the socio-organizational and cultural context in which the purchase evolves, the work task will be different, as far as its various phases. In our example, if the buyer is a private citizen, he will not have to carry out the different tasks that concern a company which has to find stockholders in order to raise capitals. In the same way, if the buyer is a foreigner, he will have to inquire about the legislation of the real estate country.

For each of these phases of the work task, we will have to enter into one or different processes of Information Seeking, that make possible to achieve the final target.

In the same way, if we take the example of a private citizen who wants to buy a house for his and for his family, one of these work sub-tasks will consist in choosing the place in which he desires to live in. To guide his choice, he will need to know the area which offers his the greatest number of professional opportunities according to his field of activity, that offers the main services he and his family need (as schools, hospitals, or transport), and that offers the best weather conditions. (tab.2.1).

In order to answer to these questions, the user can use Information Retrieval, but then we can realize how it is difficult to make the right questions for someone who has only a weak knowledge of the target domain. Other activities, which belong to the area of learning and investigation have to be managed in parallel with the simple Information Retrieval. That is what we call the Exploratory Search.

Marchionini [Mar06] displays a model aimed to show the interaction between these different activities. (fig.2.2).

So, trying to obtain a result from a seeking task involves three big sets of activities: (i) investigation, (ii) learning and finally (iii) lookup (e.g Information Retrieval). These activities can be made at the same time and they interact throughout the search (this explains the usage in the figure 2.2 of

Table 2.1: Work Task: Example: Purchase of a real estate

Context Socio-Organizational: French informatician who desires to live in Italy with his family

Work Sub-task	**Seeking Task**	**Seeking Sub-tasks**	**Exploratory Search**	**Information Retrieval**
Study of the project feasibility	Economic conditions	How is the IT market doing in Italy?	What are the indicators or benchmarks used? Satisfaction survey among Italian informaticians , Survey on the main IT Italian companies	Statistics on unemployment in the IT sector average annual salary in the IT sector, corporate turnover of the main IT companies, number of employees, etc
		What is the cost of living?	What are the everyday products for Italian people?	Cost of different everyday products, evolution of purchasing power in the last year, evolution of the real estate market, etc
		How does work the social system in Italy?	What legal documents have to be consulted? What are the reference organizations?	Consultation of identified texts, consultation of the phone numbers of the reference organization
	Legal conditions	What are the requirements for residing in Italy?	Which are the laws to consult? Who can help in the bureaucratic task(lawyers,etc)?	Consultation of identified texts, consultation of the phone numbers of lawyers,etc
Choice of the place	Professional conditions	Which are the more economically vibrant and competitive regions in the IT sector?	Survey on the geographic distribution and the activities on the main IT Italian companies Survey on the demographic distribution of the (informatician) workers in Italy	Statistics on unemployment by region Which region benefits of European subventions? etc
		In which region the companies are most interesting in work with French companies?	Survey on the different partnerships between French and Italian companies	Who are the clients of these companies, which products are sell?
	Personal preferences	Which region offers more services?	Survey of the Italian educational system Survey of the medical system Satisfaction survey	School rankings....
		In what regions the weather conditions are the mildest?	Survey on the Italian climate	Number of days of sunshine by region...

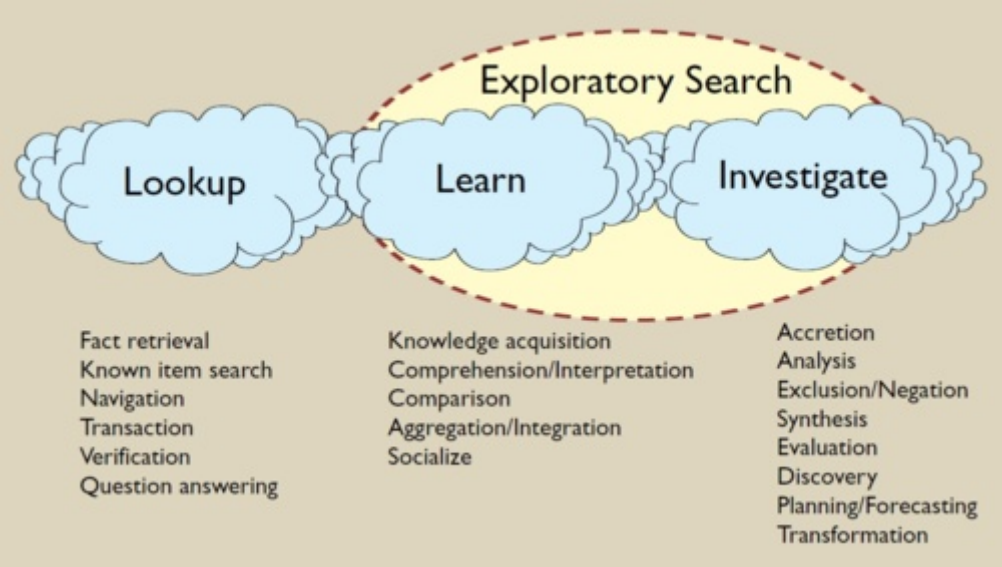


Figure 2.2: Three kinds of search activities [Mar06]

borderlines between the different activities which are deliberately represented as nebulous). Then, the Exploratory Search which principally includes the activities of learning and investigation is essential in the process of Information Seeking (without it we cannot have a global vision of the problem and then we are not able to start on a Lookup task).

The table 2.1 shows the different tasks to carry out in order to solve a complex problem, and allows to understand the interaction that can exist between the tasks of Information Retrieval (Lookup) and Exploratory Search (learning and investigation). Nevertheless, by being fairly "instinctively" elaborated and without any help tool, this table surely miss out some essential points allowing to join the target fixed by the work task. This suggests how much we need tools able to help users in exploratory search tasks. Another interesting element resulting by the observation of this example is the diversity of the domains to study in order to resolve only one work task. Indeed, the given example treats only two phases of the whole process (feasibility study and the choice of the living place), but certainly involves many others like the choice of funding or the choice of the property type. Each sub-tasks requests new knowledge in very different domains (like

the legal area, and the housing market). This diversity of information makes just more obvious the need to create tools in order to help the user in his choice.

2.2 Users motivations

After having placed the Exploratory Search activity in the more global domain of the Information Seeking, it is opportune to identify the characteristics of the reasons which steer users to explore a certain domain.

An Exploratory Search task can be started in the context of an internal or an external motivation. Internal motivations can be various, such as simple curiosity, the wish to learn something, or even the desire to make an opinion. External motivations can be detected in the need to solve a problem or the need to make a decision in an unexplored domain.

By considering these motivations we can draw up the main characteristics of an user. Indeed the complex nature of their goals, which are not reachable in a linear way, determines users:

- who own little or no knowledge in a target domain;
- who have difficulties to formulate a goal, which will likely evolve in the course of the search (thanks to the learning and investigation activities);
- who have little clues on the steps which allowed to reach their goal (if it is defined).

  These conditions, which determine the ill-structured nature of the Exploratory Search problem context, on opposition to the strong-structured nature of the Information Retrieval problem context, contribute to create a general feeling of uncertainty, which can play an important role in the outcome of an Information Seeking.[1].

[1]Kuhlthau[Kuh91] is one of the first who has demonstrated the psychological impact of the Information Seeking process and how this can affect the search itself. Her model includes six steps, each of them matching with different feelings (from the uncertainty to the sense of accomplishment).

2.3 Exploratory Search behavior

As it is shown in figure 2.3, the feeling of uncertainty tends to disappear as far as the knowledge domain is better known. Specific tools are then necessary to allow a user to go over this thankless phase. But before designing these instruments, it is necessary to analyze in a precise way the behavior observed during an Exploratory Search task.

An Exploratory Search task begins most often with a domain discovery phase during which a user undertakes what White and Roth call an Exploratory Browsing [WR09].

In this phase, the user gathers knowledge and analyzes it in order to circumscribe and comprehend the field in which his search is located. At this point, the learning is not focused yet. In fact only the essential piece of information are gathered by the user who makes work of synthesis with the pieces of knowledge that he has collected in order to orient himself more consciously in the information space which he is discovering. Concretely, the Exploratory Browsing is characterized by a navigation spread in a wide information space (in function of the subject), where the user jumps from an hyperlink to another, sometimes using search engines to find a new direction but most of the time not in the aim to obtain an answer to a specific question. Tools like Aemoo[MND+12], which allow to support the Exploratory Browsing are very precious for the user during this phase. Indeed, these kind of tools permit to suggest at the user new directions of exploration or to establish relations between the various pieces of information he already hold and the new ones, and allow to learn in a synthetic way the gleaned information.

Moreover, it is very important to provide the user a way to orient his search, allowing for example to go back and visualize the followed path. That is the reason why the graphic interface and the presentation of the information play a crucial role in this stage. In fact, it is decisive to avoid the user to feel himself as more confused as more the knowledge field gets wider (cf Kuhlthau ISP model [Kuh91]).

As the search task progresses (cf fig. 2.3), investigation and learning

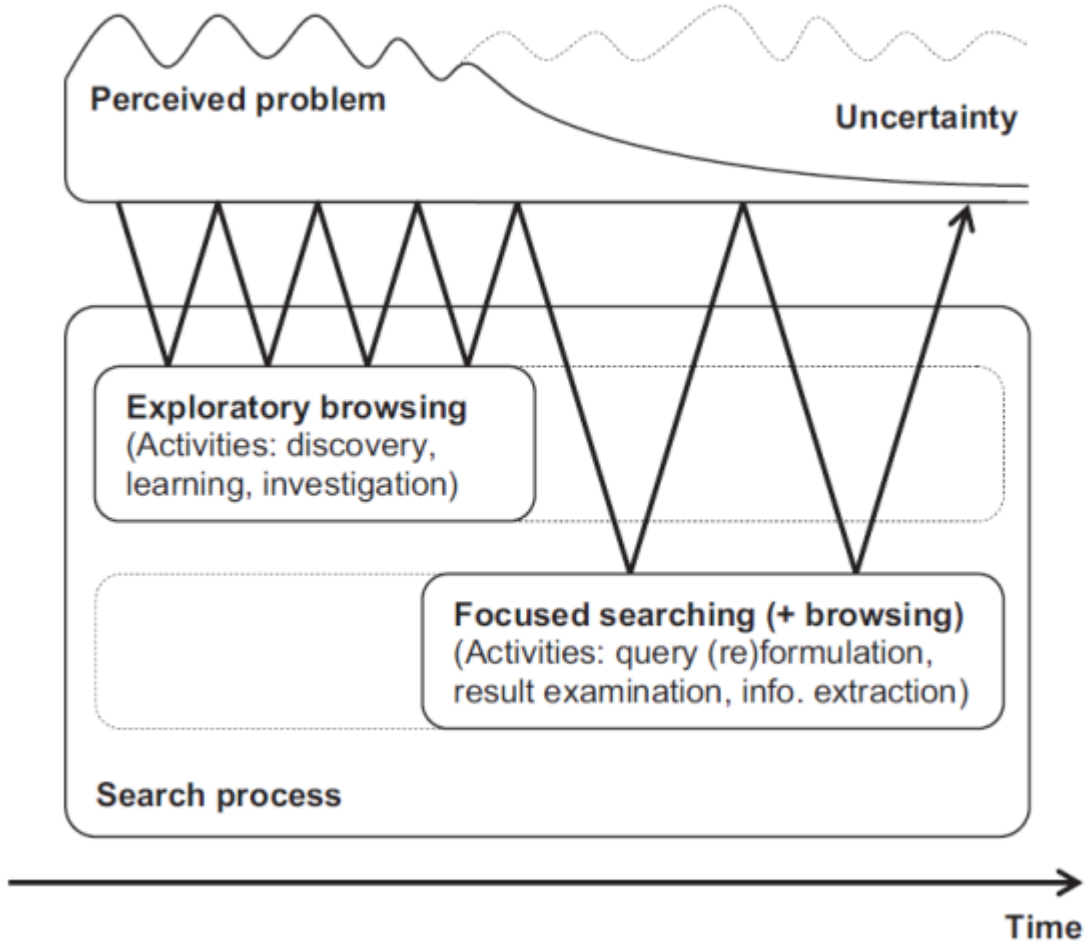


Figure 2.3: A model of Exploratory Search behavior[WR09]

activities are more and more replaced by focused search learning. The user who during his search task has identified the most important subjects of his search in order to reach his initial goal (which is defined more and more precisely) wants now to acquire an in-depth knowledge on specific topics. In the same way, he may want to verify the hypothesis he formulated during the first phase of investigation. He is now able to formulate more precise questions and has recourse to Information Retrieval.

But it will be during this phase that new problems, not foregone before by the user, can appear. These problems come out during the examination of the obtained results. Indeed, the user resuming his activity of Exploratory

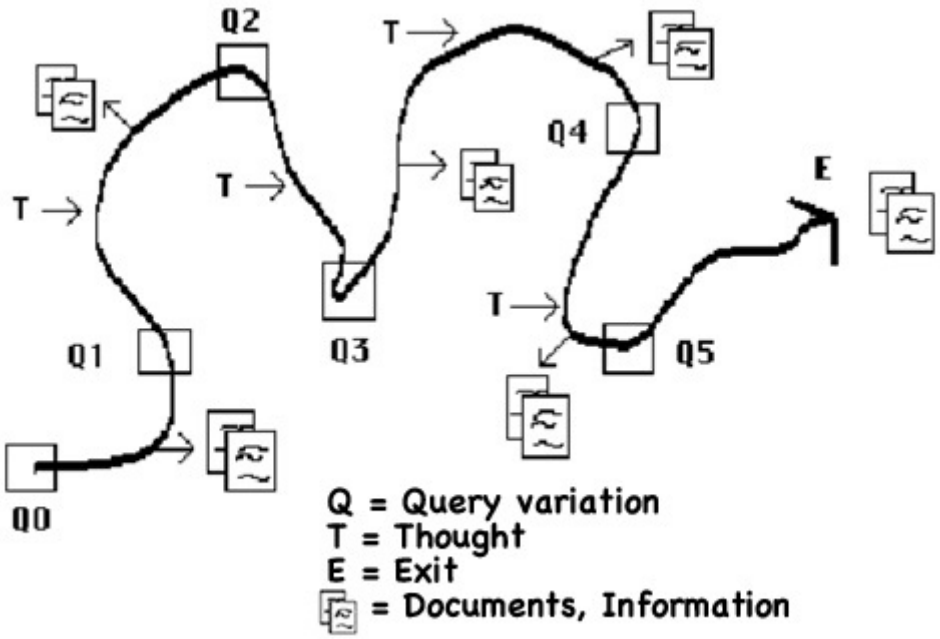


Figure 2.4: Berrypicking Model [Bat89]

Browsing discovers other facets of the studied domain he ignored before. With this activity, in the same manner as outlined above, he can refine his vision of the problem and extract the information he needs. This incremental process, composed of successive iterations, moving from discovery to discovery recalls the Berrypicking model of Blates[Bat89].

The Berrypicking metaphorically reproduces the dynamic nature of the process of the Information Seeking. Blates makes an analogy with the gathering of red berries scattered across a wide forest. Each met element gives new ideas and directions at the search. Nevertheless, this model uses the Information Retrieval like a means of transition from a document to another. They are the queries which in some way guide the search.

Thanks to the instruments supporting Exploratory Search, it is now possible to avoid this "keyword dictatorship", which does not make possible to formulate complex questions and force the user to face the vagaries of the search engines which are not able to satisfy these complex needs. On the

contrary, the presentation of the pieces of information linked to the main topic of interest allowed the user to choose the path he wants to follow and to be more productive in his seeking activity.

The next section will present a collection of technologies whereby it is possible to build such tools: the Semantic Web technologies.

Chapter 3

Web of data

3.1 From a Web of Documents to a Web of Data

The exploration of a new domain of knowledge often requires the identification of its entities, their types and the different relationships existing between them. For instance, if we want to know more about Romanticism, we will have to identify who are the artists who took part in it (e.g. Delacroix, Goethe, Manzoni, Verdi....), the main works of this period, the relationships existing with other artistic movements. This would include even the geographical space in which the movement evolved. Afterward we will surely desire to know more about this or that work and to understand, for example, the influence that it had on other artists.

The Web 2.0, i.e. the Web that we browse nowadays, considers only the documents, both textual or multimedia, and untyped hypertext links in order to establish relationships between themselves. Any tool based on Web 2.0 technologies (HTML, URL), is then unable to identify automatically both the concepts associated to a document (e.g. a person, a project, a description of a movie and so on), and the nature of the link which has been defined between the concepts. By consequence, Exploratory Search is considerably slowed down. Before choosing the next step in his exploratory path, the user has

to read sometimes the whole document in order to extract the information, to understand it, to grasp if it is helpful for his aim, and finally to decide the next direction to take. By analogy, we can say that at the moment an exploratory search user seems to be like a tourist, neither without a map nor without a guide that can be helpful to orient him in the unknown place he is browsing.

This is the challenge of the Semantic Web [BLHL01].

The Semantic Web was firstly introduced by Tim Berners-Lee as *"an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation*". Especially, in the domain of Exploratory Search, the task of reading and understanding a lot of new documents, ordering them and understanding their relations is now entrusted to computers. Computers, thanks to clever technologies, are able to orient the users like a touristic guide would do for a traveler.

In recent years, the Web has evolved from a global information space of linked documents to one where both documents and data are linked. Underpinning this evolution is a set of best practices for publishing and connecting structured data on the Web known as Linked Data [BHB09]principles. These principles are namely:

- to identify a common model in order to identify the resources in the Web. It is the role played by the usage of the URIs- Uniform Resource Identifier

- to identify a common model designed to describe the resources and to establish the relationships that could exist between them. It is the role played by RDF – Resource Description Framework;

- to build vocabularies that made possible to define formally, in an interpretable and interoperable way, the semantic of data: this is the role of the ontologies built thanks to RDFS or OWL

- to determine a common query syntax allowing to find on the Web the

different resources and the documents associated to them according to different selection criteria. This is the role of SPARQL.

The figure 3.1 , that we will use as an illustrative example, allows us to better understand this evolution. The first layer, i.e. the current Web is formed by documents, and then by unstructured information. The addition of semantic annotation by following the Linked Data principles allows on the contrary to define and to establish relationships between these documents, making possible in this way the interpretation of data contained in these documents through software agents or Web Services.

3.2 Identifying and describing resources: URI and RDF

RDF [CK04]- Resource Description Framework- is an essential element of the Semantic Web. Indeed, it allows to describe in a simple and uniform way the Web resources (today it constitutes a standard), and it represents one of the first step to make possible the automatic interpretation of the data.

To do this, each resource is universally identified by a URI [BLFM05](Uniform Resource Identifier), that can be assigned to a data available on the Web (a document, an user account in a provided service), as well as to an object of the real world (a country, a person), to which we desire to associate an identifier in the context of online representation, or even a relationship (belonging, affiliation,etc.).

For instance:

- http://en.wikipedia.org/wiki/Johann_Wolfgang_von_Goethe is the URI identifying an article on Wikipedia treating Goethe (a document)

- http://dbpedia.org/resource/Johann_Wolfgang_von_Goethe is the URI identifying the person of Johann Wolfgang von Goethe

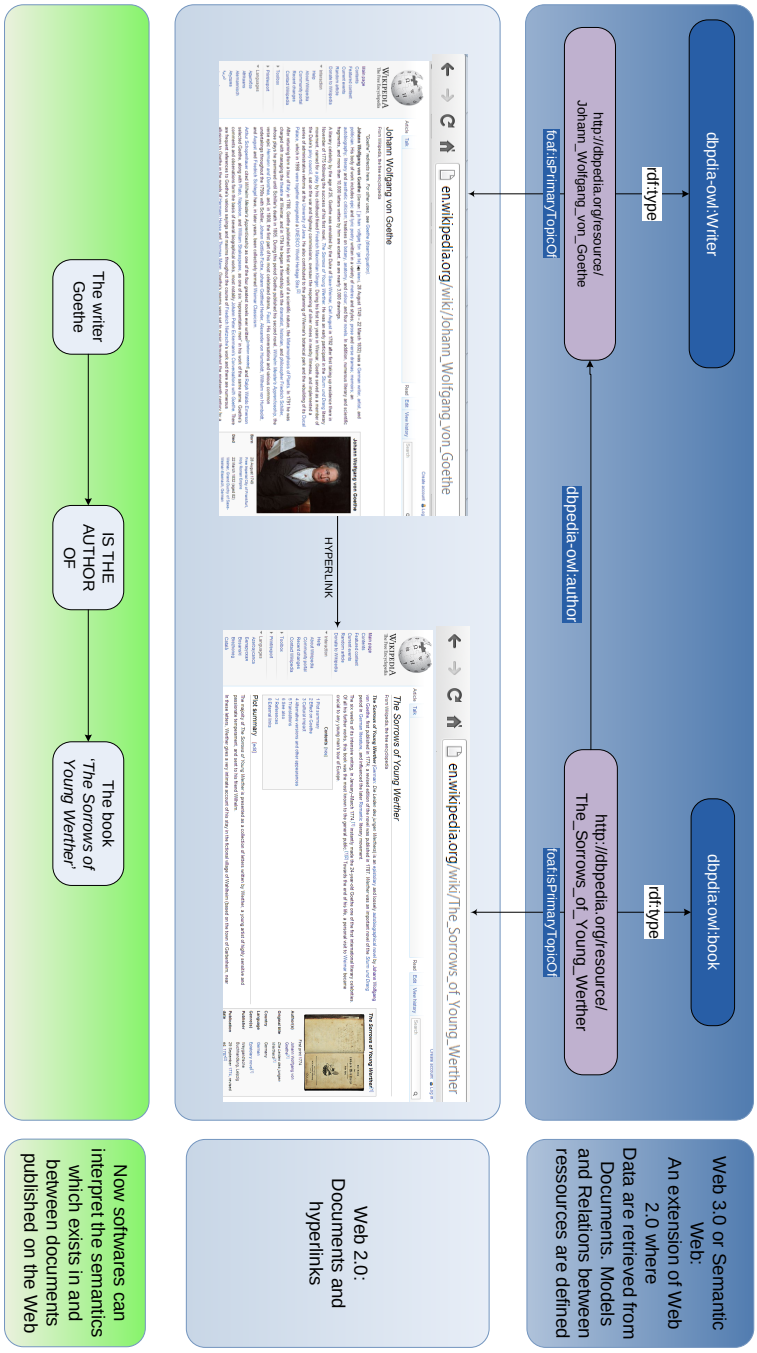


Figure 3.1: Semantic Annotation of Documents

- http://dbpedia.org/ontology/birthDate is the URI identifying the relationship existing between a person and his birth date.

It is important to make a distinction between the URI of a resource (i.e his identifier) and the URL (Uniform Resource Locator) of one or of the many documents describing it. If we reconsider the example previously quoted, we can realize how two kinds of resources exist:

- contents which belong to the web of documents (a document, a post blog, a document describing the semantic relationships...) for which the URL of the document can correspond to the URI of his identifier. Indeed it is coherent to consider that the document identified with this URI correspond to a document placed at this same address;

- the data of the real world (a person, a country, a relationship...). Of course these resources do not have any URL but we can name them and afterward make them to be part of the Giant Global Graph[1] (aka the Semantic Web or The Web of Data).
  Then if it is unlikely that by typing the URI *http://dbpedia.org/resource/Johann_Wolfgang_von_Goethe* in a web browser, the ghost of Goethe come visit us, this virtual identification of real resources allows to link the entity Goethe to documents that, directly or indirectly, may relate to it, and in the same way to establish (well characterized) relationships with other identities of the real world.

The goal of RDF is so to construct this knowledge graph where documents and data are both represented. It is based on the notion of triples, allowing this way to define the assertions concerning resources.

Each triple is composed by:

1. a subject, i.e. the resource to which it is assigned a property, identified by an URI;

[1]The Giant Global Graph or GGG is an expression used for the first time by Berners-Lee in 2007 in his blog to qualify the Semantic Web (and so replaces the WWW acronym). The blog page can be consulted at http://dig.csail.mit.edu/breadcrumbs/node/215

Listing 3.1: Turtle Representation of RDF triples

```
@prefix foaf:<http://xmlns.com/foaf/0.1> .
@prefix  rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dbpedia:<http://dbpedia.org/resource/> .
@prefix dbpedia-owl:<http://dbpedia.org/ontology/> .

dbpedia:Johann_Wolfgang_von_Goethe  rdf:type dbpedia-owl:Writer.
dbpedia:The_Sorrows_of_Young_Werther  rdf:type dbpedia:owl:book;
dbpedia-owl:Author dbpedia:Johann_Wolfgang_von_Goethe .
```

Listing 3.2: RDF/XML Representation of RDF triples

```
< rdf:rdf
 xmlns:dbpedia="http://dbpedia.org/resource/"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dbpedia-owl="http://dbpedia.org/ontology/">
<dbpedia-owl:book rdf:about="http://dbpedia.org/resource/
    The_Sorrows_of_Young_Werther">
 <dbpedia-owl:Author>
  <dbpedia-owl:Writer rdf:about="http://dbpedia.org/resource/
      Johann_Wolfgang_von_Goethe"/>
 </dbpedia-owl:Author>
</dbpedia:owl:book>
</rdf:rdf>
```

2. a predicate, for instance the property assigned to the resource, which is also identified by an URI;

3. an object, for example the value of the property. This one can be of a primitive type (a string, an integer...), or another resource, which can be the subject of another triple leading to the creation of a network, whose nodes, like its edges, are represented by URIs.

Different serializations of RDF allow us to represent assertions according to different syntaxes. That is the case of N3, Turtle (which derived of N3), RDF/XML, or even the graphic representations. Therefore, the "semantic layer" of the figure 3.1 and the two examples of code 3.1 and 3.2 represent the knowledge conveyed by the sentence that are "Goethe is a writer who is the author of the book *The Sorrows of young Werther*".

It should be noticed that there is no requirement for the set of triples of a given resource to be stocked in the same file. On the contrary, since this approach is based on the principles of the Web, and thus of a distributed

architecture, it is perfectly possible to define such information in many documents, because the identification of resources subsequently enables to trace the origin of each assertion.

3.3 Vocabularies and ontologies : RDFS and OWL

URIs and RDF offer a standard support to represent information in the context of the Semantic Web, that is to say a network in which every node and every edge is identifiable. However they do not enable to define the semantics of treated data.

In fact, with URIS we are able to uniquely identify resources in the Web, and RDF provides a data model for representing structured information in the Web.

It is therefore necessary a technology to model the reality (or one of realities), in order to have the possibility to link the used words (the URIs) to a meaning and to define the potential relationships with other words.

3.3.1 Definition

The word Ontology has its origins in philosophy, where it denotes the science concerning *"all the species of being qua being and the attributes which belong to it qua being"* (Aristotle, Metaphysics, IV, 1), which has for aim of answering the question "What does it means to be?".

In the field of IT, many definitions have been suggested. In the context of A.I., Guarino [Gua98] defines one ontology as:

"an engineering artifact, constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words."

Thus, according to Guarino, ontologies are artifacts which describe a vision of the world (or one of the visions, in accordance with the rating scale

we refer to), that afterward can allow the machines to interpret the meaning and to act accordingly, when they encounter a term on the Web. This is the definition that we will adopt from now on.

3.3.2 The components of an ontology

An ontology is composed by:

- entities: each one corresponds to a concept or named class. A class defines a group of objects, either abstract or concrete, that we desire to model for a given domain. Every concept can be associated to various linguistic declinations;
- properties allocated to the classes, among which we usually can distinguish:
  - the relationships existing between the classes or instances of these classes, for example hierarchy relations;
  - the primitive attributes we can associate to the different concepts or to their instances (the class "name of a Person" would be linked with a String, and the class "age of a Person" with an Integer...).
- axioms, that allow to model logic assertions and that are used in the definition of the concepts or of the properties in order to refine them. Thanks to them, we can infer new facts starting from a basic piece of knowledge (for example by defining a symmetrical relationship), or we can prove the consistency of a group of statements (for instance by giving minimal or maximum cardinality to an object).

The knowledge base containing the different instances of the concepts modeled in the ontology is often distinguished by the ontology itself. We can notice, in order to make the difference, that the prefix of the DBpedia knowledge base is *http://dbpedia.org/resource/,* while that one of its ontology is *http://dbpedia.org/ontology/.* Moreover a knowledge base may be related to many ontologies.

Listing 3.3: Basic example of an ontology RDFS (in Turtle)

```
@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dbpedia:<http://dbpedia.org/resource/> .
@prefix dbpedia-owl:<http://dbpedia.org/ontology/> .
@prefix  rdfs:<http://www.w3.org/2000/01/rdf-schema#> .

dbpedia-owl:Writer rdf:type  rdfs:Class ;
   rdfs:subClassOf dbpedia-owl:Person.
dbpedia-owl:Book rdf:type  rdfs:Class ;
   rdfs:subClassOf dbpedia-owl:Work.
dbpedia-owl:author rdf:type rdf:Property ;
   rdfs:domain dbpedia-owl:Book;
   rdfs:range dbpedia-owl:Writer.
```

3.3.3 Ontology modeling languages: RDFS and OWL

RDFS [BG04] - RDF Schema- is a basic means to model ontologies on the Semantic Web. This language introduces the notions of class (*rdfs:Class*) and property (*rdf:Property*), associated with subsumption relationships that can define class hierarchies (*rdfs:subClassOf*) and property hierarchies (*rdfs:subPropertyOf*). RDFS also enables to define the domain (*rdfs:domain*) and the range (*rdfs:range*) of each property. An RDFS ontology is written in the form of RDF triples that define in this way identifiers as a consequence of their different classes and properties, as they are unique being based on URI.

RDFS allows to achieve the first simple rules of inference thus making possible, for a knowledge base, to enhance itself with new assertions, once certain assertions are present in it. These rules in particular include the transitivity of the properties *rdfs:subClassOf* and *rdfs:subPropertyOf*. For example, if the class Person is defined as a subclass of Living Being, and the class Writer is defined as a subclass of Person, we can conclude that a Writer is a subclass of Living Being.

The code 3.3 represents an ontology that models some of the classes and of the properties we have shown in the previous example

But the expressive power of RDFS is rather limited. For instance, it does not permit to express constraints of coherence: for example, with RDFS we could express that a person cannot be both a man and a woman at the same time (class disjointness). Moreover, certain algebraic properties character-

izing the relationships between different classes, like symmetry, cannot be expressed (if a man is married to a woman, then a woman is married to him).

Thus, in order to provide a language for designing more expressive ontologies, since 2001 W3C has established a working group concerning OWL [Gro09]- Web Ontology Language.[2] OWL resumes the notions of classes and of properties defined in RDFS, and precises them respectively with *owl:Class* (subclass of *rdfs:Class*), *owl:dataTypeProperty* and *owl:objectProperty* (subclass of *rdf:Property*), by distinguishing in this way the attributes (primitive type) from relationships (links with other classes). Above all, OWL adds new constructors and axioms allowing to increase the expressive power of ontologies, with a more powerful semantics than RDFS. In fact, OWL is composed by three sub-languages, whose expressive power increases from the first to the last:

- OWL-Lite which extends RDFS and adds new constructors like the symmetry of the properties and of the cardinality constraints (only 0 or1);

- OWL-DL, which adds supplementary constructors, like boolean combinations of classes (like union or intersection), axioms of classes (like disjointness), and extends the cardinality constraints of OWL-Lite;

- OWL- Full, which does not add any constructor in comparison with OWL-DL, but interprets them in a different way, thus giving a stronger expressive power (each class is seen at the same time like a class, an individual, and a group of individuals).

[2]Since 2004 OWL is a recommendation of W3C, and has been updated in December 2012.

3.4 Query Language for the Semantic Web: SPARQL

While RDFS and OWL enable to define ontologies on the Semantic Web and RDF allows to model assertions based on them, it is necessary in order to get access to the knowledge stored in this graph, to dispose of an adapted query language.

SPARQL [PS08]- SPARQL Protocol and RDF Query Language- thus propose at once a language and a protocol to query the modeled data in RDF. SPARQL uses the principle of identification of the paths in a graph, in order to get back the results of a given query.

Thus, a SPARQL query is composed by.

- an operator (which defines the type of the query);
- a pattern (the necessary part for the identification of the corresponding graphs);
- modifiers (for example, ORDER BY).

A query can interrogate one or more documents RDF in several ways: by the use of a FROM clause at the beginning of the query, by the intermediate represented by APIs- Application Programming Interface- that makes possible the simultaneous consideration of several resources, or by the use of RDF warehouses data associated to SPARQL endpoints.

SPARQL disposes of the four following operators:

SELECT that, as its name suggests, selects several elements depending on the specific query pattern. A query designed to recover the list of Goethe's works will be:

Listing 3.4: Query SPARQL SELECT

```
1 SELECT ?book
2 WHERE { ?book dbpedia-owl:Author dbpedia:Johann_Wolfgang_von_Goethe }
```

CONSTRUCT that enables to transform an RDF graph into another graph. For instance, to move from the DBpedia model to our ontology, we can use:

Listing 3.5: Query SPARQL CONSTRUCT

```
1
2 CONSTRUCT {?writer my-ontology:wrote ?book}
3 WHERE {?book dbpedia-owl:Author ?writer}
```

ASK that allows to answer a query, by identifying whether the pattern we look for is in the interrogated graph or not. Thus we can have the answer to the question "Is Goethe a writer?" with the query:

Listing 3.6: Query SPARQL ASK

```
1 ASK { dbpedia:Johann_Wolfgang_von_Goethe rdf:type dbpedia-owl:Writer }
```

DESCRIBE that returns, in the form of an RDF graph, a description of the resource given in argument. This description depends on how the SPARQL engine is implemented, and it can return, for example, the group of the triples whose subject is this resource. In that case, if we want to know the group of the assertions concerning Goethe, we'll write:

Listing 3.7: Query SPARQL DESCRIBE

```
1 DESCRIBE { dbpedia:Johann_Wolfgang_von_Goethe}
```

SPARQL however present several limits, especially if compared to a language like SQL. For example, at the moment it does not offer aggregate functions (like COUNT(..) in SQL), neither the possibility to add data in an existent graph (like UPDATE in SQL). Yet several extensions allow to solve these limits and to add new features (like SPARUL- SPARQL Update -concerning the addition of data to the graph, for example).

Chapter 4

Aemoo

Thanks to the technologies of the Semantic Web, succinctly presented above, we can now build tools that can guide the user no longer only because of the laws of probability, but mainly thanks to meaningful relationships. Moreover, with these technologies, new navigation paradigms can be created. In fact, traditional navigation consisting in sequentially entering keywords can be replaced or completed with a navigation that encourages discovery. Jumping from one link to another, the user can conduct a smarter search, deciding which path he has to follow anytime he jumps a link, while being guided in his browsing according to the relationships between the resources.

This navigation is particularly suitable for the Exploratory Search since as we have previously said, the user has only a limited knowledge of the domain he is exploring, and then he needs suggestions if he wants to continue his browsing without problems. In this section, after a brief overview of the existing tools, we will present Aemoo [MND+12], an Exploratory Search system that takes advantage of the Semantic Web technologies.

4.1 Related Works

In this section, we present a set of tools which have the common characteristic to allow the exploratory search system using Semantic Web technologies.

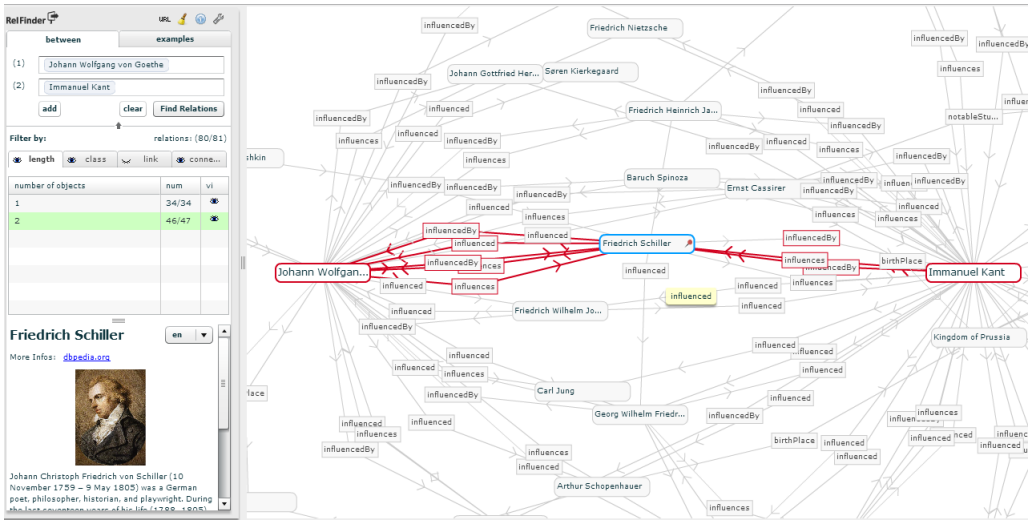


Figure 4.1: RelFinder[HHL+09]

4.1.1 RelFinder

Relfinder [HHL+09] is a tool that after the selection of two or more entities allows to find out the nature of the relationships existing between them. The links between entities are not always direct, but they can call upon other identities (selected or not). Thus, in this way we can find out (fig.4.1) that Goethe is indirectly linked to Kant through many writers and philosophers, e.g. Schiller. We discover the type of entities (philosopher, book, etc...) by reading the abstract proposed by dbpedia. This approach is useful, because we really discover the links between entities we could have suspected, but it does not propose any solution for the information overload. Indeed, as we can see in figure 4.1, with only two entities selected at the beginning, we obtain big graphs very hard to use for an exploratory search task.

4.1.2 Factic

Factic[TB10] is a faceted browser, which allows the user to face the information overload, in order to categorize information by his own interest, and his own need. It uses ontologies and structured data (RDF) in order to achieve carefully this classification. It allows many possibilities of navigation,

like for example the selection of the link on a list after the keyword entry (figure 4.2a) , or the discovery of new relationships through the visualization of a graph with different searched identities (figure 4.2b). At any time, through specific queries the user can filter out the content shown discovering in this way the most relevant information for him. However this approach implies two gaps: (i) the user has to be very active in the definition of these filters while, as previously said, during an exploratory search task the user often has difficulties to know exactly what he wants; (ii) the user needs to have some knowledge of the mechanisms underlying the Semantic Web (SPARQL, ontologies, etc..) to understand the vocabulary used by the tool in order to accurately define queries. This aspect makes its usage difficult for a common user.

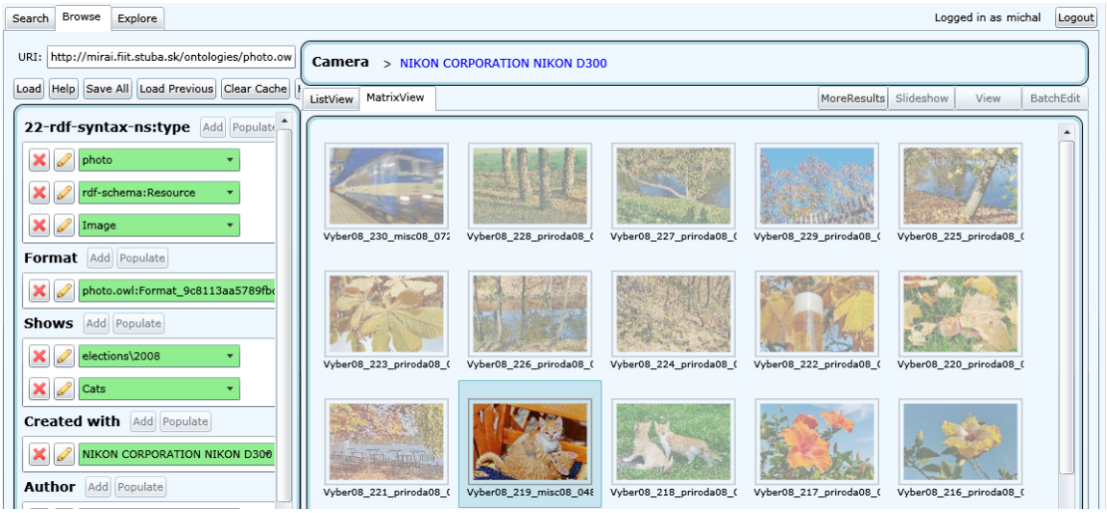
4.1.3 Bletchley Park Text

Bletchley Park Text [CMZ05] was[1] an original mobile application that allows the visitors of the Bletchley Park Museum to deepen by themselves the knowledge acquired during the visit of the museum. Once this is over, indeed, the visitors are invited to select a group of topic of interest that has awake their curiosity. By connecting to the website of the museum, a group of stories is proposed to them. These stories are semantically annotated documents that allow to identify the relationships between the topics of interest retained by the visitor (following the same reflection of RelFinder). This application shows a great ability to transform data in source of knowledge and thanks to this the end-user can really satisfy his curiosity and learn desire (both current stimulus inciting a person to visit a museum). This successful exploratory search tool, although interesting is however limited by the reduced size of the set of used documents (400 stories and 1700 topics of interest available).

[1] Up to now the application is no more online.

Figure 4.2: Factic[TB10]

(a) Browser



(b) Explorer

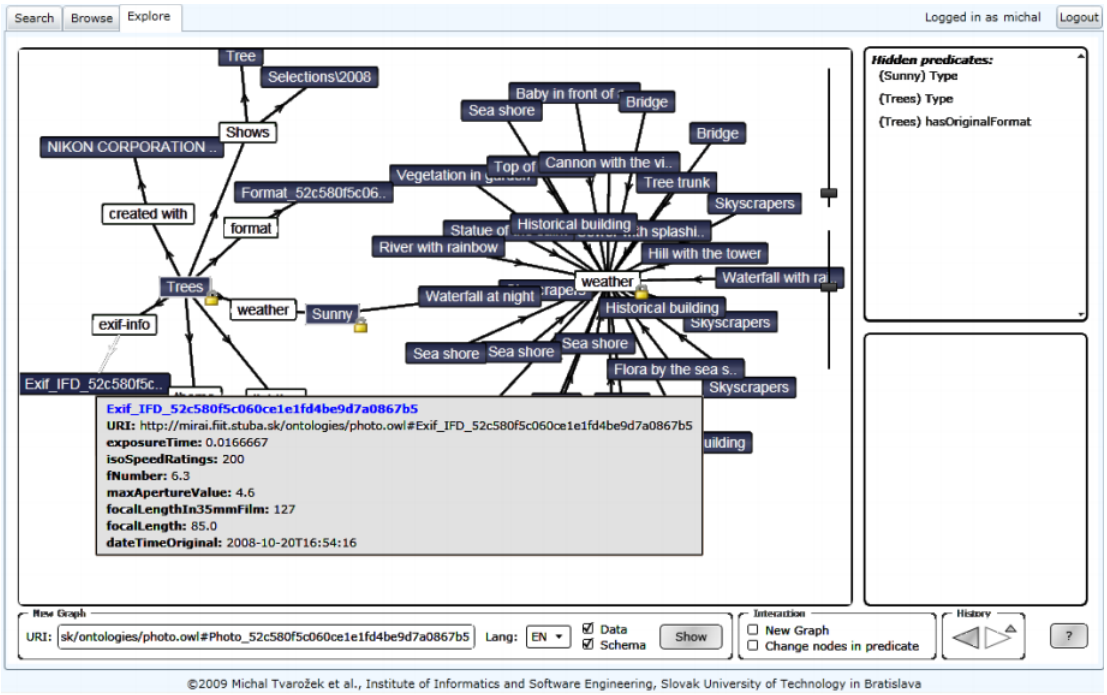
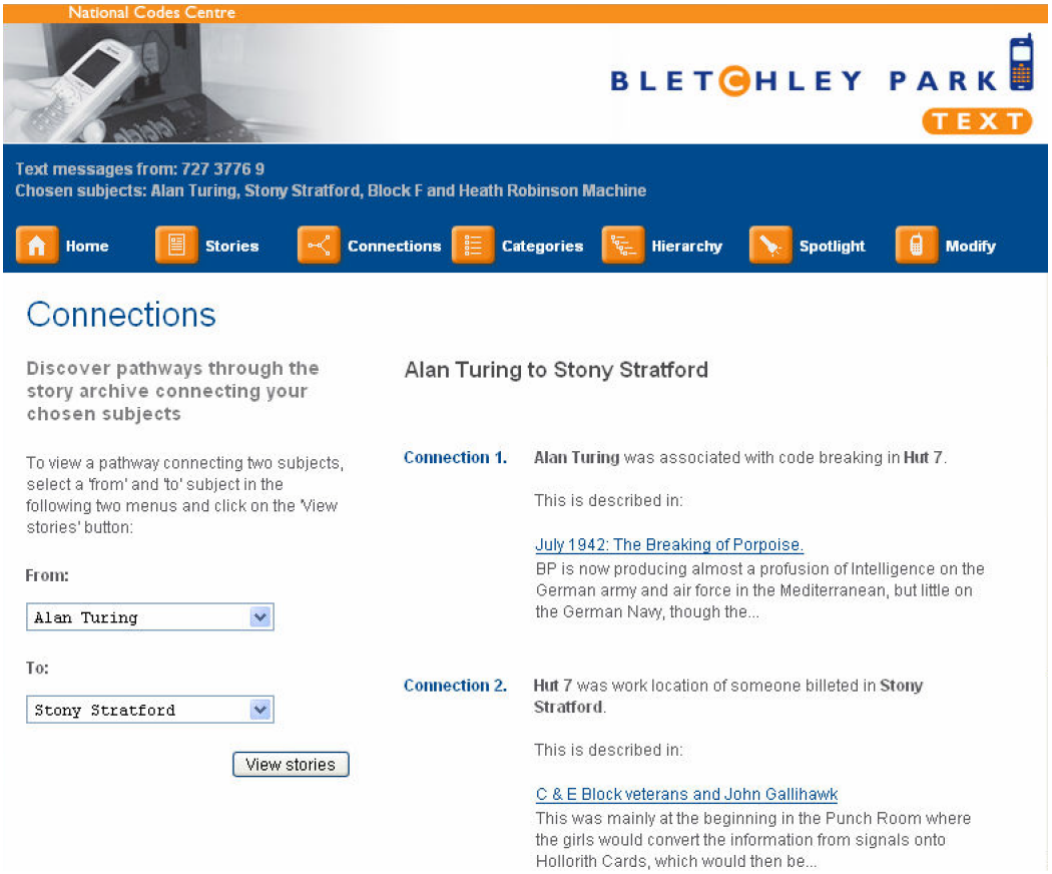


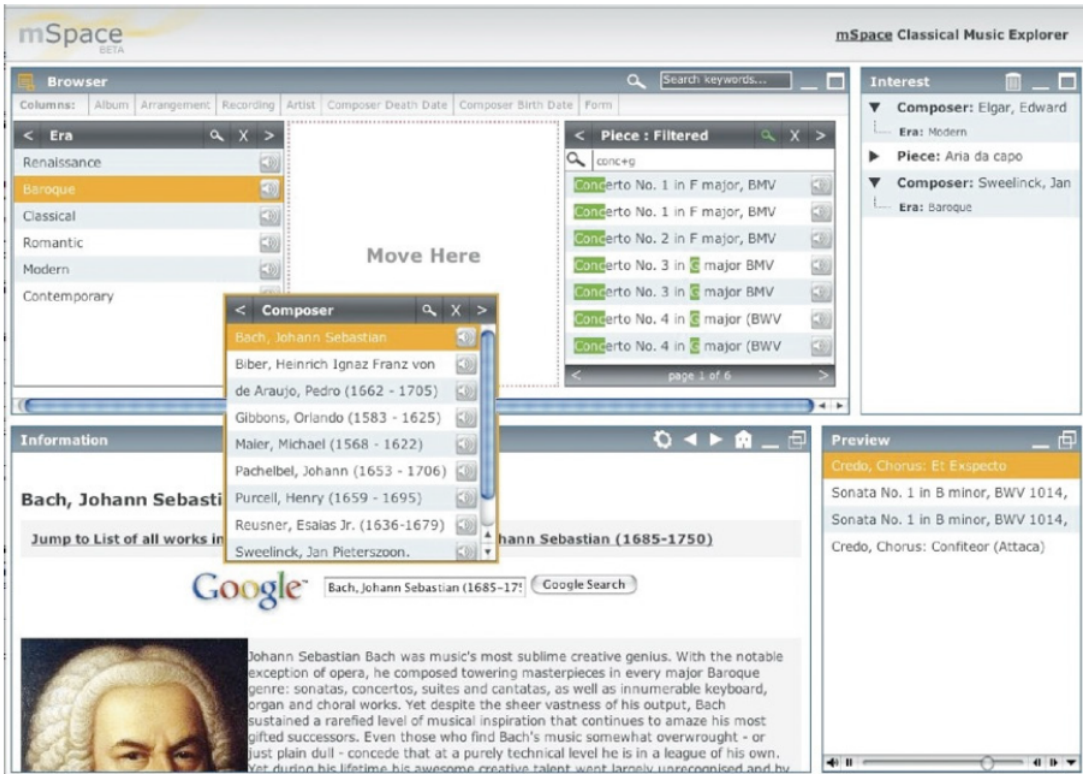
Figure 4.3: Bletchley Park Text[TB10]



4.1.4 mSpace Explorer

mSpace Explorer[GHS04] is a faceted browser based on Semantic Web technologies. The user has to choose between different columns which represent collections of entities of the same type. The filtering of information is then done by the creation of columns (whose title must be selected through a list of the available types). In this way, for each entity selected in the first column, a list of entities appears in the next column (obviously this happens if these entities are linked by semantic relationships). This kind of filtering is a good example about how the user can build queries without any knowledge of the underlying technologies. This way of discovering new data is fruitful in the exploratory search but the user has to know quite precisely what he

Figure 4.4: mSpace[GHS04]



is looking for. Then the incentive of this study is to give to the user the key to avoid the overload of information at the condition that he has already a certain knowledge of the subject of research: he has to understand the true nature of the relationships which link collections between them and to foresee those which will be profitable during his search.

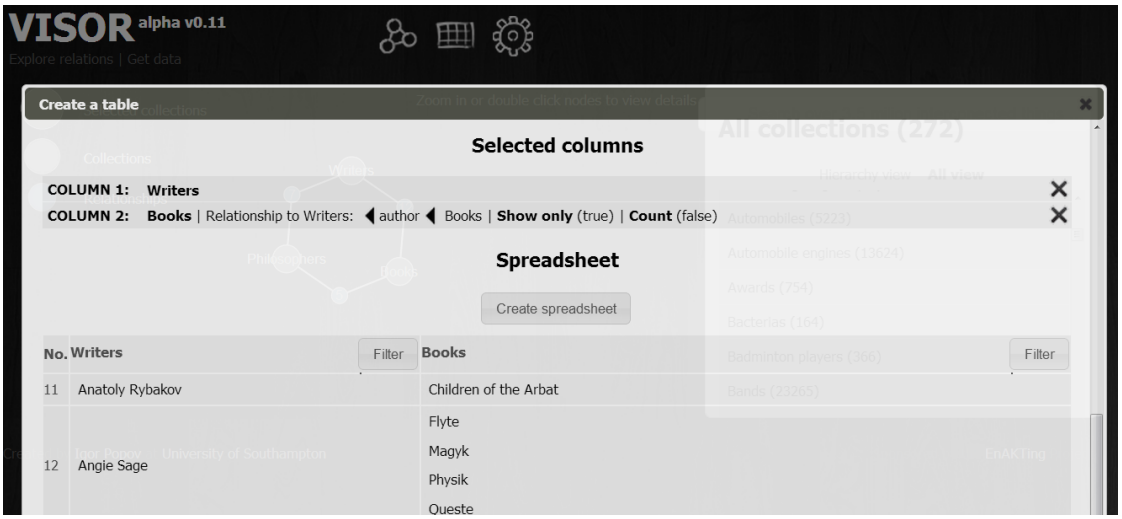
4.1.5 Visor

Visor[PSHS11] designed by Igor Popov under the supervision of M.C. Schraefel and Wendy Hall allows to build a similar presentation of data as mSpace, but enabling the user to specify the type of relationship he desires between the collections. Each collection is composed by entities of the same type according to the DBpedia Ontology[2]. In addition, the results are no

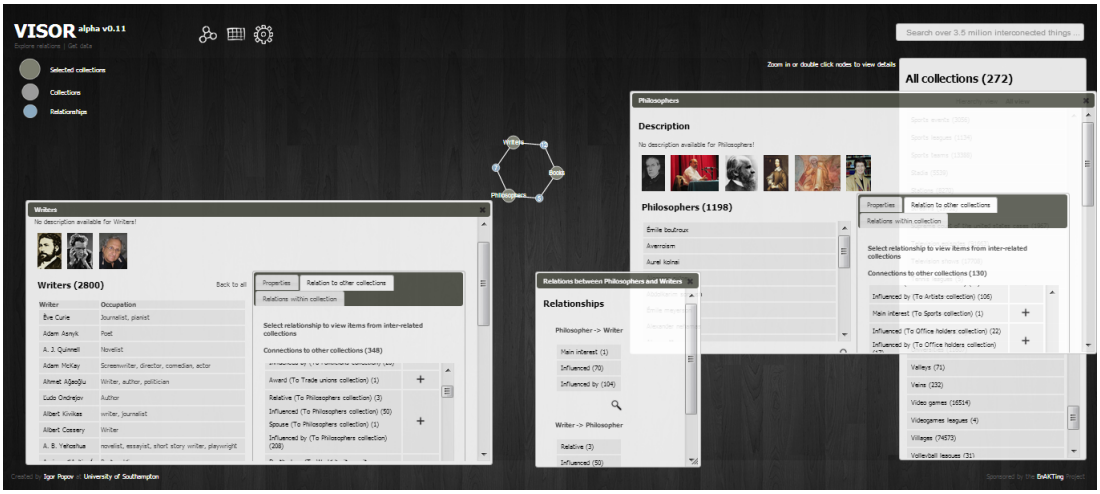
[2]http://dbpedia.org/ontology

Figure 4.5: Visor[PSHS11]

(a) Visor - Table visualization



(b) Visor Collections visualization



more viewed through the selection of the item in the list of each collection, but as in a graphic interface of a DBMS (figure 4.5a). Thus this allows to have an overall vision of the results instead of just one step of the navigation path as in mSpace. Moreover, Visor adds a graphic representation of the links joining the collections themselves (figure 4.5b). Thus, the tools follows the idea of RelFinder, without applying it to the entities but to the whole collections of entities. In this way we can have an overall vision of the relationships that connect the different collections of entities and the information overload can be avoided. By clicking on the node of the graph representing the collections, we can consult the details of each item of the collection and to view the relationships they have with the other items of the collection, or with items belonging to other collections. In the same way, by clicking on the nodes of the graph representing the relationships, we can consult the list of relationships existing from a collection to another. This approach is useful in order to make general exploration on a subject and to find out the relationships that may exist between several collections of entities. However, it does not allow to specialize the search on a specific entity.

4.2 Aemoo

Aemoo [MND+12] is a non-traditional search engine, based on the paradigm of the Exploratory Search. On the contrary of Visor, it is focused on the discovery of relations between entities, even if it puts in evidence the main type of each entity. His particularity is the use of the Knowledge Patterns to counteract big issues encountered by search engines like the negative effect of the information overload and the lack of orientation during the navigation. After having presented the main functionnalities of this tool, we will introduce the concept of Knowledge Patterns and we will expose how Aemoo uses them to produce a smart presentation of the information.

4.2.1 Features and Interface

4.2.1.1 Information presentation

Aemoo proposes an easy navigation, allowing an user without any knowledge of underlying technologies to use the tool in a profitable way. The way in which a user interacts with Aemoo is indeed very simple. The exploration starts with a keyword based search. Once the user has selected the starting entity, the application shows us a star graph centered in this entity, and whose peripheral nodes represent the types of resources linked to it (see figure 4.6). On the left we can see both the type of the resource and the first lines of the related Wikipedia page.

In this way, in a blink we can understand the principal knowledge topics to which a resource is linked. For example, Goethe is related to resources typed as Book or Writer, while Barack Obama is linked to resources typed as Legislature or Office Holder. If we move the mouse on the external nodes we can afterward read the list of resources of each type. In this way we can discover that Goethe is linked to "*The Sorrows of young Werther*". To discover the nature of this link we can put the mouse on the edge that joins the principal node with the external node, or on a link in the list of the selected resources. Then, the sections of the Wikipedia article is displayed on the screen, explaining the nature of the link that joins the two resources. In our example, the part "his first novel, *The Sorrows of young Werther*" immediately enables the reader to identify Goethe as the author of the book.

In this way, in a few minute we can have a general overview of the subjects we will have to approach in order to narrow the knowledge domain we want to explore.

4.2.1.2 Navigation:

With Aemoo, we can conduct the search in two ways:

- with the insertion of a new keyword in the search bar

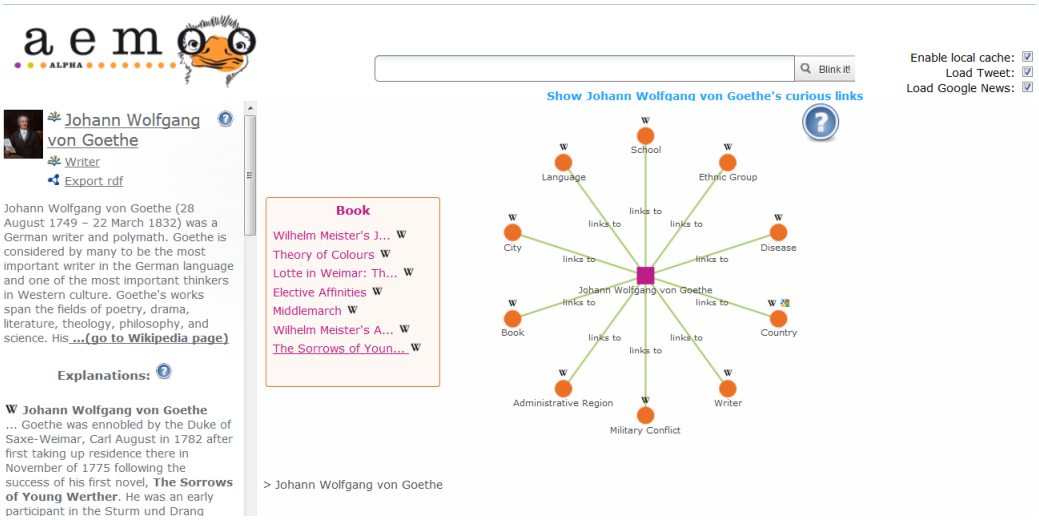


Figure 4.6: Aemoo[MND+12]

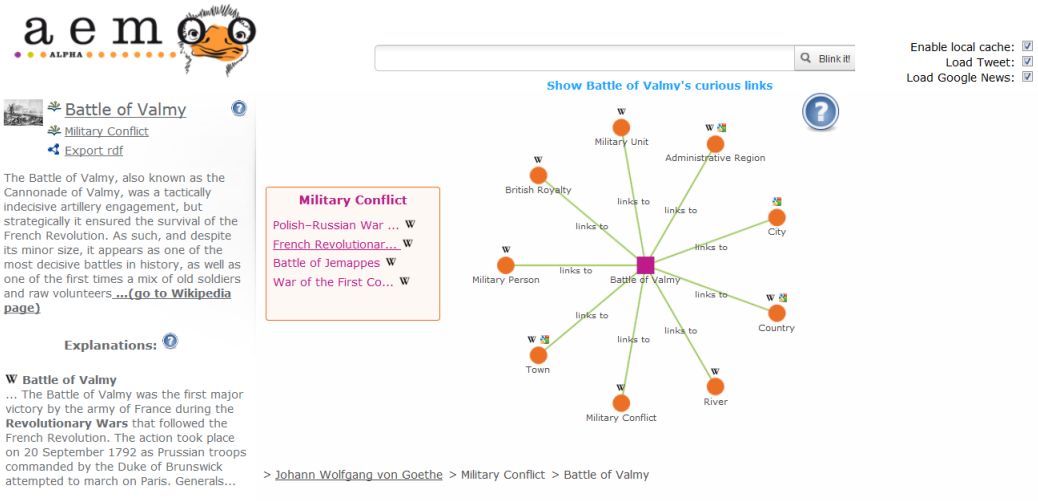


Figure 4.7: Aemoo - Jump towards another entity

- by clicking on one of the links suggested by the star graph (which will lead us to a new star graph): the new explored entity will be necessary semantically linked to the previous one.

The second choice is not offered by traditional search engines. Indeed, even if our current search is linked to the previous one, these tools compel us to enter a new keyword. This restriction is a problem for the exploratory search user: indeed, like we previously said, it can be difficult for a user involved in an exploratory search task to use an accurate vocabulary, and to know a priori the right keyword to employ. Aemoo, on the contrary, permits a more linear navigation and allows to change the current research topic, simply by selecting a link created during the previous search.

These displacements from a search topic to another, that sometimes can be taken as "jumps", correspond very well to the paradigm of the Exploratory Search that, as White and Roth point out in [WR09], evolves in a wider knowledge space, because research is not targeted and known in advance (figure 4.8).

In fact, taking again into account the example shown in figure 4.6, by navigating the links that Aemoo proposes, we can be interested by the link between Goethe and Military Conflicts. If we move the mouse on the node we will discover, for instance, that Goethe took part in the Battle of Valmy. If we do not know anything about this battle, we click on the link, in order to know more about this event, and we will discover that it is a famous battle that took place in Champagne-Ardenne in 1792. In this battle the young French Revolutionary Army was opposed to the Prussian Army, and Goethe took part to the side of the duke Carl-August of Saxe-Weimar (fig. 4.7). This kind of exploration makes arise to the user questions like: "Did the successful outcome of the French Revolutionary Arm influence Goethe in his works?" Thanks to the compactness in which the information is presented, the user is indeed invited to be detached towards the subject, and then he can more easily manage it. If we take again the analogy previously used, Aemoo serves at the same time as a map and as a guide for the tourist that embodies the

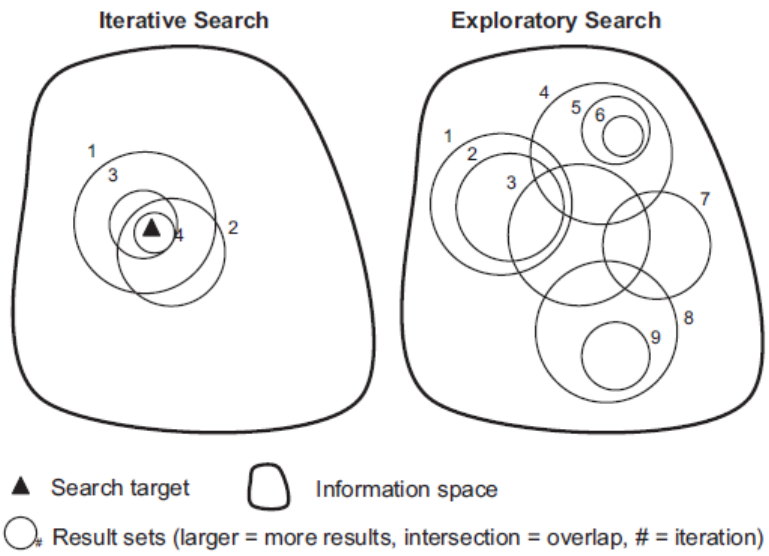


Figure 4.8: Navigation on Aemoo: an Exploratory Search path [WR09]

exploratory search user, while it leaves him a great freedom of evolution. Moreover, the breadcrumb placed at the bottom of the screen enables him at any time to remember the path he has followed and to "retrace his steps" if he wants to. (fig 4.7).

4.2.1.3 Classification between Core and Curiosity Links

One of the distinctive features of Aemoo is that it puts forward the relationships generally existing between a certain type of entity and other ones. In Aemoo that is named the Core Link's. On the contrary, the entities which do not own often straight links with the type of entity in question are listed in the Curiosity Links. Thus Goethe, who is mostly known as a writer, is a resource of the Writer type, and then it presents some "Core Links" towards Book typed resources (figure 4.6). On the contrary, in the "Curious Links", there are links towards resources of the Scientists type because Goethe is equally the author of several scientific essays, even if its reputation (at least for the general public) does not come from this discipline. (figure 4.9).

Thus this categorization gives to the user the opportunity to better orient

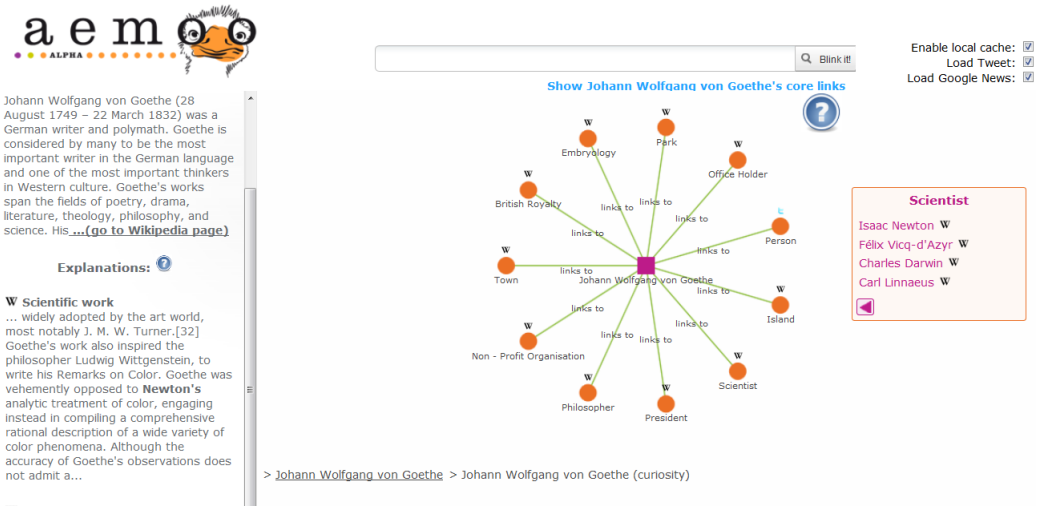


Figure 4.9: Aemoo- Curious Links

himself, and to understand immediately the characteristics of this or that resource, while it allows to graphically isolate the "essential" information linked to a resource.

In the next paragraph, we will introduce the concept of Knowledge Pattern, concept on which is based, and that allows him to present the data in a structured way, and to oppose the problem of information overload by establishing these distinctions between "core" and "peculiar" or "curious" links.

4.2.2 Knowledge Patterns

As defined by [GP10], a Knowledge Pattern (KP) is a formalized schema representing a structure that is used to organise our knowledge, as well as for interpreting, processing or anticipating information. KPs are complex knowledge structures which derives from Minsky's frames [Min74] and can be seen as small, well connected units of meaning which are (i) task-based, (ii) well-grounded, and (iii) cognitively sound. KPs can be used as the base on top of which a variety of tools and tasks can be designed and implemented. These can be recommendation systems, entity summarisations, or, like Aemoo is,

exploratory search tools.

The traditional approach of extracting KP is the top-down one: the experts consider a certain problem, and on the basis of the reasoning and of their knowledge of the domain, compose patterns, in order to guide the future ontologist in the accomplishment of his task.

On the contrary in [NGPC11], a collection of KPs has been identified, according to a bottom-up approach, that is to say starting from a substantial among of heterogeneous data contained in a knowledge base (DBpedia[3]),[4]The study started from the analysis of the relationships that link an entity to other ones by the property *dbpo:wikiPageWikiLink*. By identifying the main type of each entity and the frequency of relationships with other entities, it has been possible to extract automatically a consistent number of KP by following a precise methodology described in [NGPC11]. As the study was based on a knowledge base which contained encyclopedia data, the Knowledge Pattern term has been specialized in Encyclopedic Knowledge Pattern (EKP).

The successful result of this study has been the extraction of 184 EKPs, and Aemoo uses them to identify the relationships corresponding to the Core Links, and on the contrary to distinguish them from Curious or Peculiar Links and to construct dynamically the star graph. In the next section we will see how.

4.2.3 Implementation

4.2.3.1 General Architecture

Aemoo is a distributed web application build on the Ajax model. The client part, based on HTML and Javascript interacts with a Java Web Service, according to the REST architecture. An HTTP GET request that contains the DBpedia URI as parameter, corresponding to the entity sought by the

[3]http://dbpedia.org/

[4]The choice of DBPedia was made because as it depends on the content produced in Wikipedia which contains a lot of articles concerning many different subjects and whose writing is made democratically, it can be considered as a good font to represent many different concepts encountered in the world.

Listing 4.1: fragment of the Writer EKP

```
<#linksToBook> a :NamedIndividual,
                 :ObjectProperty;
                rdfs:comment "Relation between Writer and Book"@en,
                             "Relazione tra Writer e Book"@it;
                rdfs:domain <http://dbpedia.org/ontology/Writer>;
                rdfs:label "links to Book"@en;
                rdfs:range <http://dbpedia.org/ontology/Book>;
                rdfs:subPropertyOf <http://dbpedia.org/ontology/Work>;
               skos:relatedMatch <http://dbpedia.org/property/notableworks>,
                                 <http://dbpedia.org/property/prevtitle> .
```

end-user, is send to the server. This one send back a response message whose body, in XML/RDF format, contains the graph that defines the relationship between the entity initially considered and other relevant entities. This graph is dynamically constructed thanks to the Knowledge Patterns, concept that we have presented before.

4.2.3.2 Knowledge Patterns in Aemoo

In Aemoo, each Encyclopedic Knowledge Pattern (EKP) defines a frame for each type of resource. For an entity of the type Writer, the corresponding EKP establishes relationships with other types of entities such as Book, Newspaper, Country, etc. For each of them, it draws up a list of properties that link them most frequently. The fragment of the code 4.1 corresponds to a part of the Writer EKP used by Aemoo (converted in Turtle serialization for a better comprehension). In this way it defines that a writer is strongly susceptible to have relationships with entities of the Book type and at the same time it defines the nature of these links.

Thus we better understand how Aemoo works: after having received in input the DBpedia URI corresponding to the entity sought by the user[5], the server, through a request at the DBpedia SPARQL end-point identifies the main type of the entity. Then it retrieves the EKP which corresponds to the type of the entity, allowing then to construct the RDF graph according to the

[5]the identification of the dbpedia URI corresponding to the keyword query entered by the end-user is made thanks to the use of the search engine Apache Lucene

scheme suggested by the EKP: each entity which is linked to the sought entity by the property *dbpo:wikiPageWikiLink*[6] is analyzed in order to identify its type. If this one is present in the EKP, then it is added to the graph of the core-links. Otherwise it is dismissed. The construction of the RDF graph of the Curious Links exactly follows the opposite reasoning. The construction of the graph is made thanks to a SPARQL query of the type CONSTRUCT.

So to make an example, considering that the end-user is searching for the entity "Goethe", the client part sends an HTTP GET request with as parameter the URI *dbpedia:Johann_Wolfgang_von_Goethe*. The server, thanks to a SPARQL query identifies the type of the entity (*dbpedia-owl:Writer*). Then it retrieves both the EKP corresponding to *dbpedia-owl:Writer* type and all the entities which are linked to *dbpedia:Johann_Wolfgang_von_Goethe* by the propriety *dbpo:wikiPageWikiLink*. In this list of entities, if an entity has for type *dbpedia-owl:Book* it will be added in the graph of the Aemoo Core Links. On the contrary, like in this EKP the recurring relationship between a writer and a scientist is not expected, the relationship that links *dbpedia:Johann_Wolfgang_von_Goethe* to *dbpedia:Isaac_Newton* will be put in the Curious Links.

Aemoo can also take into account more than one source of data: it can integrate the data coming from Twitter and Google News which, through a Web Service provides unstructured data (tweets and news). Aemoo uses the Apache Stanbol Web Service which syntactically analyzes a character string to extract the named entities: the names of the people, of the companies, of the products, of the places, etc. Then it links these entities with resources contained in the DBPedia knowledge base and assigns them the correspondent URIs. The result is impressive. After retrieving the data, that now is structured, the server applies the same mechanism allowing to identify the correspondence between the EKP and the entities to whom the tweets and the news refer to. The entities whose types are in the EKP are then added to the graph, dynamically.

[6] the list of these entities is retrieved thanks to a SELECT SPARQL query

In this way Aemoo shows that it is now possible to build applications which use unstructured data and semi-structured data, and to present them in such a way that this data makes immediately sense for the final user, allowing him to really improve his exploratory search experience.

Chapter 5

Improving the exploratory search experience through the analysis of the user path

As we have previously suggested, the exploratory search is a process which includes complex activities, during which the user explores a world of new knowledge, i.e he navigates along a tortuous path before reaching the goal. The aim of this thesis is to study how the exploration of the user can be analyzed in order to improve his experience. In the first chapter we focus on the path of the user to demonstrate how it can be exploited to provide effective advices. In the second chapter we explain how Semantic Web technologies, relying on the graph structure of the Global Giant Graph, allow to easily use this source of information, exposing our methodology expressed as an algorithm. Finally, in the third chapter, we provide the design and the implementation of a prototype based on the Aemoo tool to illustrate our theory.

5.1 How the path can give information of the user search preferences

As we have suggested before, an exploratory search user has only a vague vision of the topic of his search and, as suggested by the metaphor of Berrypicking, this topic can evolve during his path. In this way, the initial search topic can easily change during the search session and the observation of just a single query can gives only poor information. Another approach has to be taken into account, in order to observe the whole process of search. In figure 4.8, which presents the user path in the information space, we can isolate the topics (represented in the figure as subset of the information space) which concentrate more interest for the user. For example, in the figure 5.1a, the subset of results generated at the third search iteration includes pieces of information contained in the subsets generated at the iteration 1, 2 , 4, 7 , and 8. We can transform this subset representation in a graph, in which each node represents a searched entity, and each edge represents the direct relationship linking these searched entities as the figure 5.1b shows. A concrete example of this correspondence can be given as follows:

1. at the first iteration, the user is looking for the entity "Romanticism" (iteration 1),

2. the user looks for the entity "Literary Romanticism" (iteration 2)

3. the user wants to discover the entity "Goethe" (iteration 3)

4. the user wants to know more about the Goethe's book *The Sorrows of young Werther* (iteration 4) and about the details of this work, like the biography of Charlotte Buff who has inspired his work and that of her son (iterations 5, 6)

5. afterward the user wants to know more about the Duke of Saxe-Weimar, patron of Goethe (iteration 7), and then he discovers the reason that

leaded the writer to participate at the battle of Valmy (iteration 8). Afterward he wants to consult the map of the battle (iteration 9)

We obtain in this way an undirected graph, designed in figure 5.1, which is in fact a sub-graph of a much bigger graph. Indeed, in the figure, the nodes without number are other entities representing the whole space of knowledge and which do not belong to the sub-graph, because the user has not visited them.

Observing this graph based on a typical example of exploratory search process, we can retrieve two important features:

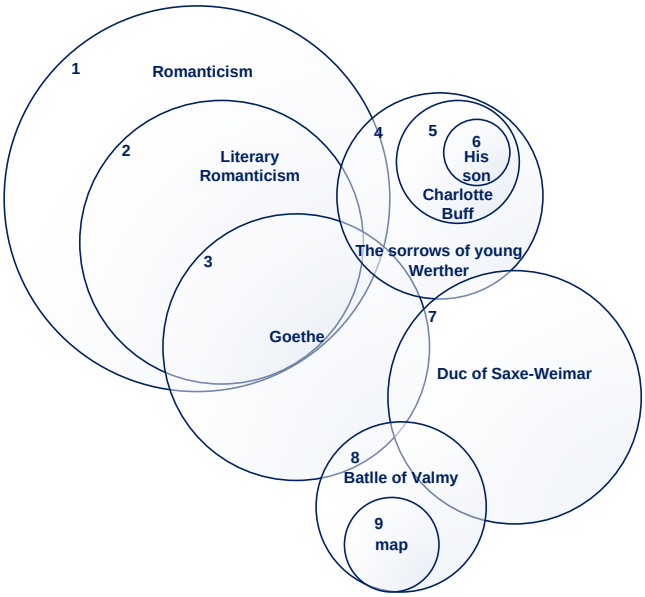
- the graph of the entities visited by the user grows according to a preferential attachment [AB02]; this means that the new nodes added to the graph (the new entities visited) have a greater probability to be linked to the nodes with a high degree. In fact during his search the user tends to concentrate his attention on few entities (in the figure they are represented by nodes with higher diameter). So, if we calculate dynamically the degree of each node, identifying the nodes with a high degree, we can isolate subjects which have more probabilities to interest the user (and ranking them on the basis of their degree);

- on the contrary, nodes which do not present links with other searches entities (except obviously with the previous entity) represent a specialization.

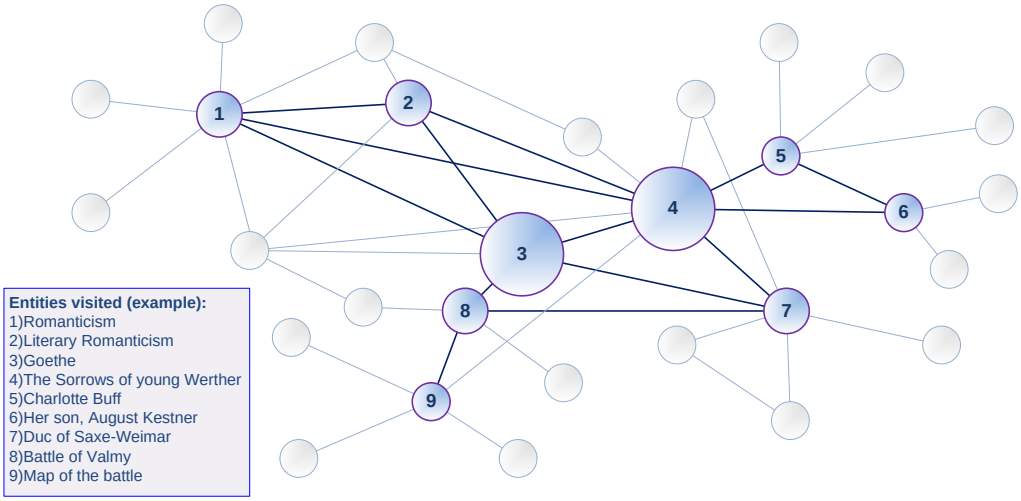
Moreover, if the graph contains typed nodes, it will be a supplementary source of information to determine the most current type of the visited nodes. In the example of the figure 5.1, we can observe that entities/nodes typed as person are the most frequent. Indeed, this can give another piece of information: the user (at this step of the search) is exploring the Romanticism, and especially Goethe, not mainly through its literary work, but through the relationships the writer had with other persons. In our example emerges a "social relationship oriented" search, and this orientation should help the

Figure 5.1: From subsets of knowledge to a graph of visited entities

(a) Subset of results at each iteration, a concrete example (inspired from figure of[WR09])



(b) Graph of the visited entities and their direct relationships



system in suggesting to the user links which can make him discover other "person entities" linked to Goethe.

All these observations give an implicit feedback of the user preference during his search. This aspect of implicit preference is precious because the user has not to interact with the system for other reasons than his current search and so can fully focus on his main task. Moreover, we have seen before that the development of too much sophisticated search filtering tools can dissuade the user, who in the worst case can even totally ignore the proposed aid. Here, on the contrary, the information generated by the user during his search gives us all the data we need to be able to orient him in his future searches.

But in order to implement a system which can exploit this information, we need a structure , i.e a graph, which is based on entities linked between them.

5.2 An algorithm exploiting the structure of the Giant Global Graph

The Semantic Web aka Giant Global Graph as introduced by Tim Berners-Lee[1], is a great opportunity to allow the implementation of tools and application like those just described. In fact, it gives a structure with semantically linked entities. Aemoo, which is a search engine based on this structure, will permit us to test our theory. In fact, at each "jump" between entities, the RDF graph returned by the Server (Aemoo Core) will permit to dynamically build the graph of the entities visited which will be our base structure. From this graph, at each step, we will apply an algorithm that we are going to describe in order to understand the direction of the user and to know which of the entities that have not been already visited can mostly interest him.

In order to rigorously describe our methodology we will firstly formally define some of the concepts suggested above, and after having exposed our

[1]http://dig.csail.mit.edu/breadcrumbs/node/215

algorithm, we will calculate its complexity in order to have the possibility to make suggestions on its usability.

5.2.1 Formal definitions:

5.2.1.1 Knowledge Graph:

According to the choice made in Aemoo, we will focus on the type of the entities, and not on those of the links, so the graph on which we based our study can be defined as $G = (N, Ed)$ with $N = (N_1, N_2...N_i)$ as the set of nodes corresponding to the entities and Ed as the set of edges which linked them.

Each node N_i is a couple $(URI_{N_i}, type\,of\,the\,entity_{N_i})$.

5.2.1.2 Individual Graph of an Entity:

An Individual Graph of an Entity is a sub-graph of the Knowledge Graph. Indeed it is a star network composed of k+1 nodes and k edges where the node at the center (N_i) is the entity currently visited and where the k peripheral nodes are the k entities at which the entity currently visited is linked. The central node N_i has a degree $d_{N_i} = k$ while the other nodes have all a degree $d_{N_k} = 1$.

5.2.1.3 Graph of the Visited Entities:

The graph of the visited entities $G_{visit} = (Ent, L)$ is composed of a set of nodes corresponding to the already visited entities $Ent_1, Ent_2, ...Ent_n$, and a set of edges $L_1, L_2, L_3, ...L_m$ which linked them (without have to pass from an unvisited node). So a link L_k is a couple(Ent_i, Ent_j).

Each entity Ent_i corresponds to a node N_j, and so we have:

$(URI_{Ent_i}, type\,of\,the\,entity_{Ent_i}) = (URI_{N_j}, type\,of\,the\,entity_{N_j})$.

Ent_1 is the initial entity searched thanks to a keyword based query. Ent_n is the last entity search.

5.2.1.4 Neighboring nodes:

We define the neighborhood $Neighbors_{N_i}$ of a node N_i as a subset composed by all the nodes which are adjacent with N_i

5.2.2 Algorithm:

The algorithm that we use is shown in the box 5.1. It allows to foresee the consequence of user's explorations choice, in order to better orient him for next step of the exploration. It would be applied at each time the user visited a new entity. First the new node visited will be added to the graph G_{visit} and the grade of all this nodes will be recalculated and the main type of the entities will be retrieved (step 1-4). Thus, the suggestions of the next path will be made (step 5).

To calculate the complexity of this algorithm, we fix n as the number of nodes in the graph G_{visit}, m as the number of nodes in $Neighbors_{N_i}$, $\sum_{i=1}^{m} h_i$ the sum of the number h_i which is the number of the neighbors for each neighbor of $N_i$and $d_{max}$the maximum degree (with $d_{max} \leq n-1$).

T_i is the algorithm at the step i. The detailed calculus of the complexity is reported in the following box:

In step 1 we have $T_1 = 1$,
In step 2, $T_2 = O(m)$ if $m < n-1$, $C = O(n)$ if $n-1 < m$,
In step 3 $T_3 = O(n \times d_{max}) + O(log_2(n))$
In step 4 $T_4 = O(n)$
In step 5 $T_5 = O(\sum_{i=0}^{m}(h_i * n))$
In step 6 $T_6 = O(2(m \times n) + m)$
So the total complexity of T:

$$T = O(n)+O(n\times d_{max})+O(log_2(n))+O(n)+O(\sum_{i=0}^{m}(h_i*n))+O(2(m\times n)+m)$$

We can observe in this way that the complexity of the algorithm depends

Algorithm 5.1 Suggestion algorithm

initialize graph $G_{visit} = (Ent, L)$ with $Ent = \{Ent_1\}$ and $L = \varnothing$
at each time an entity $Ent_i = N_i$ is visited

1. add Ent_i to G_{visit}
2. For each $Ent_j \in (Neighbors_{N_i} \cap Ent)$ add an edge between Ent_i and Ent_j
3. For each node $Ent_k \in G_{visit}$ calculate the degree of the node and add the couple $(Ent_k, grade_{Ent_k})$ to a list L ordered by degree
4. Calculate the distribution of frequency of the nodes in Ent in function of their type. Retrieve the type T which appears most often
5. Establish for each neighbor $N_k \in Neighbors_{N_i}$, the list of its neighbors which belong to the graph G_{visit} .
6. for each entity $N_k \in Neighbors_{N_i}$,
   (a) if N_k is linked with a node Ent_k which has the highest degree, suggest it N_k as 'recommendation',
   (b) else if N_k is linked with a node Ent_k which owns to the visited graph, suggest it N_k as 'interesting',
   (c) else if N_k is not linked to any node Ent_k (except Ent_i), suggest it N_k as 'specialization'

for mostly on n. The number of neighbors m and the set of variables h_i $\forall$ $i \in [1, m]$ (number of the neighbors for each neighbor) are also important but they are part of the knowledge graph and so we cannot intervene to control the size of these variables.

In order to make this algorithm efficient, we have to control the growth of the graph G_{visit}. To reduce the size of the graph incrementally built, as we have too many nodes, we can calculate the median of the degrees of the nodes. The nodes and all their edges having a minor degree can be removed. In this way, we will maintain the size of the graph at a contained number of nodes, (between 50 and 100) without having a time limit of the search

session. This allows to keep unchanged the main information of the graph. We used the median (and not the average) because as we said before, G_{visit} grows by following a preferential attachment[AB02] of its nodes. Hence, the distribution of the degree of its nodes follows a Power Law. In this case the average would not be indicative.

5.3 Design and implementation of the prototype

In order to test our theory, we have designed and implemented a small server-side module as a REST Web Service , written in PHP.

This architectural choice has been made for three reasons:

- In order to keep the information of the user path for more than one session. For doing this we have to record a file which describes the corresponding graph. On the client side, this record is not allowed for obvious security reasons (the only type of file browsers can be record are the cookies). On the contrary, on the server side, precisely thanks to the variables cookies sent by the client, the server is able to record a file for each client.

- Even if at the moment we implement a personalized recommendation system, based only on the choices that a single user does during his search, in the future it will be interesting to extend it on a social recommendation system, pooling the information contained on each personalized graph, and trying to put in evidence the most frequent choices by the totality of users.

- As a Web Service, this module can be reused by another application.

The figure 5.2 shows the global architecture of the resultant application.

As shown in figure 5.3 , at each time the user is moving from an entity to another (including if he is moving to the previous or to the next page, or

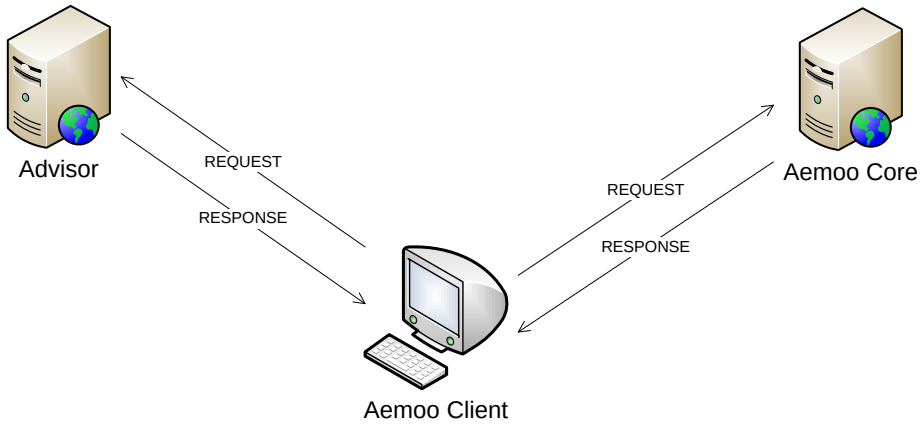


Figure 5.2: Global Architecture

Listing 5.1: RDF Graph sent in input at each time an Entity is visited

```
<uri_main_Entity> < rdfs:label> label_Main_Entity.
<uri_main_Entity> <RDF:type> <Type_Main_Entity>.
<uri_main_Entity> <WhateverProperty1><uri_Entity1>,<uri_Entity2>.
<uri_main_Entity> <WhateverProperty2><uri_Entity3>.
<uri_Entity1>< rdfs:label> Entity1.
<uri_Entity1> <RDF:type> <Type_Entity1>.
<uri_Entity2>< rdfs:label> Entity2
<uri_Entity2> <RDF:type> <Type_Entity2>
<uri_Entity3>< rdfs:label> Entity3
<uri_Entity3> < rdf:type> <Type_Entity3>
```

if he is making a new request as keyword or page reloading), the client part of Aemoo sends at the Advisor the graph RDF which describes the entity newly visited by the user and all the entities to which it is linked (what we call the Individual Graph of the Entity), after having obviously interrogated the Aemoo server in order to obtain this information[2]. The Individual Graph of the Entity must have a determined structure we report in listing 5.1.

A search begins with the first keyword query. During this first step, the incremental graph will be initialized (with the first entity visited) and will

[2]To be more precise, the client part of Aemoo implements a mechanism of history, which allows not to send again Object Requests at the Aemoo server on moving on the page previously loaded at which the user reaches thanks to the breadcrumb or the button 'next' and 'back' of the browser.

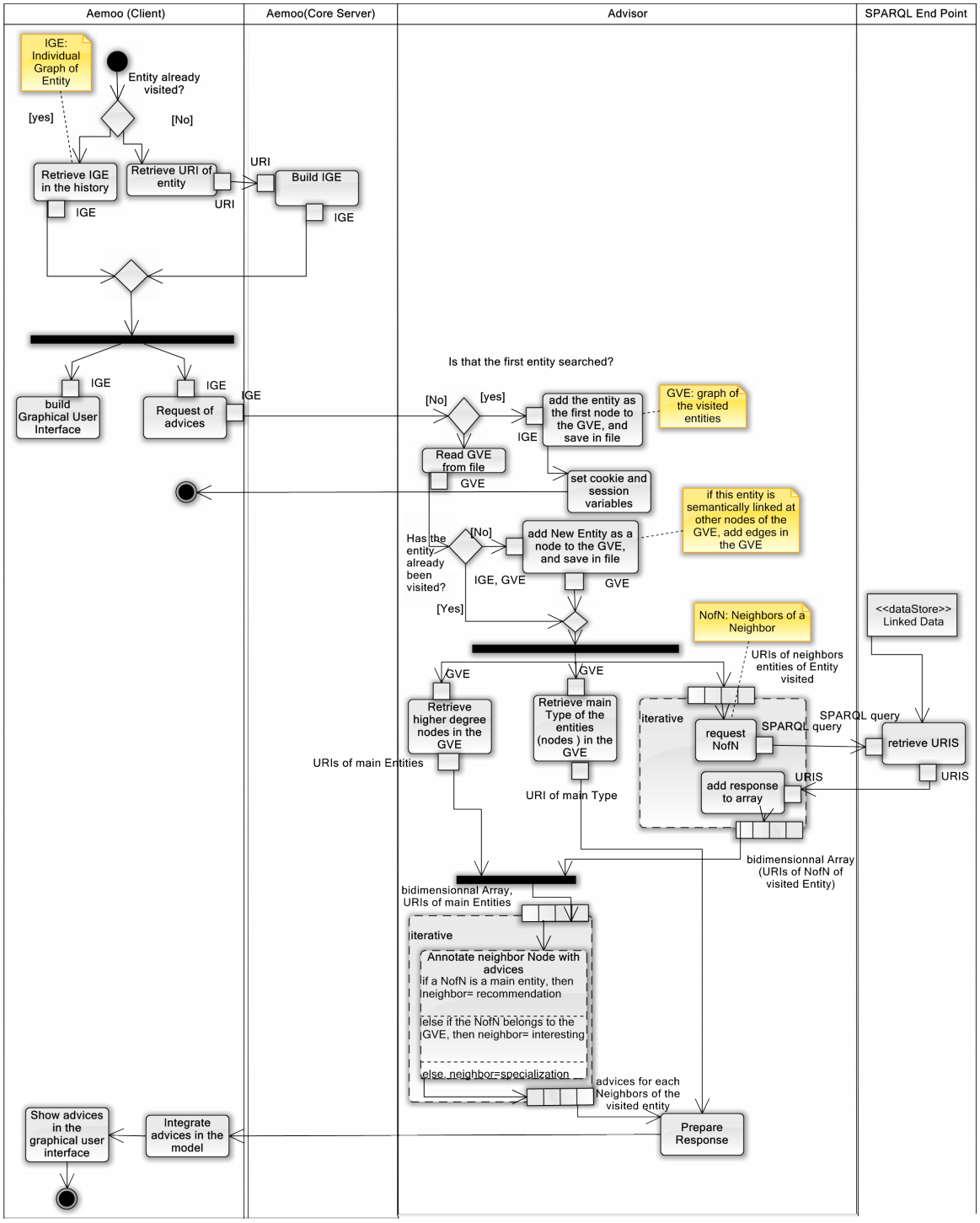


Figure 5.3: Activity Diagram

Listing 5.2: Output of the server

```
{ "mainType": "uriType_xxx",
  "suggestions":[["uri_Entity1","interesting"],
                 ["uri_Entity2","specialization"],
                 ["uri_Entity3","recommendation"],
                  ...]
}
```

be registered in a file associated with a user cookie. In fact, as mentioned above, thanks to the mechanism of the cookie, the gleaned information of a search can be stored for more than a web session (this one inevitably limited in time). This absence of time constraint (a cookie can be conserved in a browser for years) is primordial , because the Exploratory Search is a process characterized by a large temporal space.

After this initialization, at each displacement of the user in the information space, the Individual Graph of the current Entity (IEG) will be send to the server, and the script PHP will apply the algorithm described above. In order to build the incremental Graph of the Visited Entities (GVE), we have used the library EasyRDF, which allows easily to read and write RDF graphs in different format. In order to know if the neighboring entities are linked or not to other entities which concentrate the interest of the user (those which have the higher degree in the graph), the server queries one of the DBpedia endpoints [3].

After having annotated the entities $Neighbors_{N_i}$ linked at the current search (step 5 of the algorithm), the server sends the result to the client in the JSON format like it is shown in the listing 5.2.

The client, receiving the response, applies a different CSS style with each type of suggestion (after have parsed the JSON response and integrate the received data to the model) to the different entities that a user can explore. In this way, the user can visualize the advices corresponding to his search.

In order to realize this implementation, we have produced among 500 lines of code.

[3] http://wit.istc.cnr.it:8893/sparql

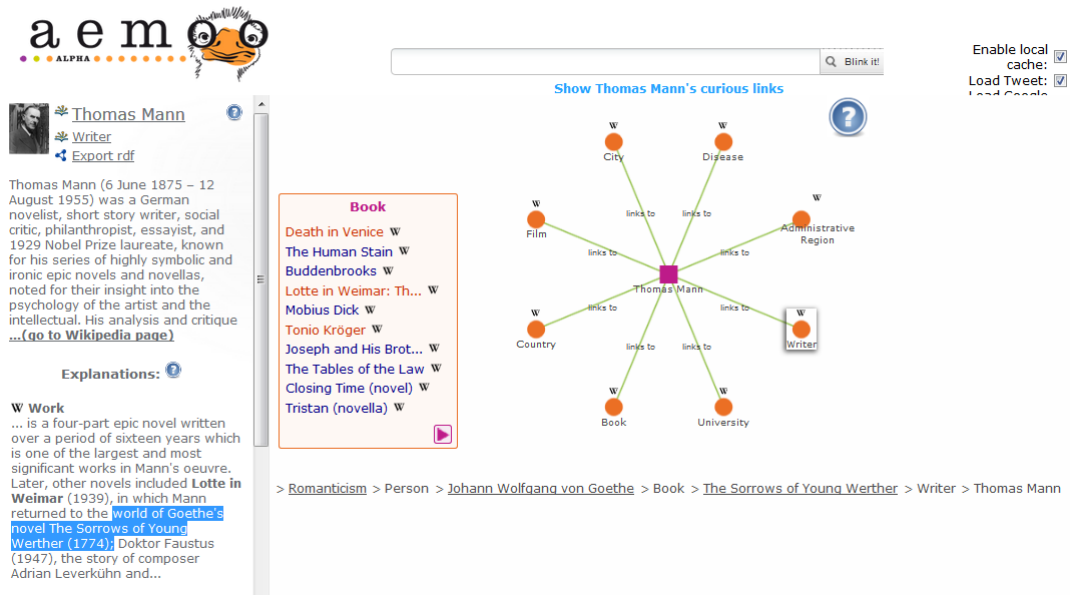


Figure 5.4: Aemoo including advices

5.4 Results

5.4.1 Main results

Using the system, we observe that generally the obtained results are conclusive.

We observe in fact that:

- the advices are meaningful. They are more and more precise as the exploration progresses;
- the user keeps the control on the navigation: all the available content is shown, but only, the recommended links are highlighted (thanks to different colors);
- if during the search, the user changes his topic, the system after a few iterations takes it into account, without the user has to do anything. So after a while, it highlights the links that are oriented towards his new topic of interest.

Listing 5.3: Graph of the Visited Entities (referring to fig.5.4)

```
<http://dbpedia.org/resource/Romanticism>
rdfs:label "Romanticism" ;
a owl:Thing ;
ns0:linksto <http://dbpedia.org/resource/Johann_Wolfgang_von_Goethe>,
            <http://dbpedia.org/resource/The_Sorrows_of_Young_Werther> .

<http://dbpedia.org/resource/Johann_Wolfgang_von_Goethe>
rdfs:label "Johann Wolfgang von Goethe" ;
a <http://dbpedia.org/ontology/Writer> ;
ns0:linksto <http://dbpedia.org/resource/Romanticism>,
            <http://dbpedia.org/resource/The_Sorrows_of_Young_Werther>,
            <http://dbpedia.org/resource/Thomas_Mann>.

<http://dbpedia.org/resource/The_Sorrows_of_Young_Werther>
rdfs:label "The Sorrows of Young Werther" ;
a <http://dbpedia.org/ontology/Book> ;
ns0:linksto <http://dbpedia.org/resource/Romanticism>,
            <http://dbpedia.org/resource/Johann_Wolfgang_von_Goethe>,
            <http://dbpedia.org/resource/Thomas_Mann> .

<http://dbpedia.org/resource/Thomas_Mann>
rdfs:label "Thomas Mann" ;
a <http://dbpedia.org/ontology/Writer> ;
ns0:linksto <http://dbpedia.org/resource/Johann_Wolfgang_von_Goethe>,
            <http://dbpedia.org/resource/The_Sorrows_of_Young_Werther>.
```

The figure 5.4 gives an example of its functioning. The search context is the following: the user has previously searched about the Romanticism, then about Goethe, then about his book *The Sorrows of young Werther* before arriving on the page of Thomas Mann. At this stage, the system will advice preferentially the topics which are linked to Goethe and at his book, because they are the topics which have for the moment hold the user's attention (the Graph of the Visited Entities reported in the listing 5.3 shows in fact that they are the nodes which have the higher degree, because Thomas Mann and the Romanticism are not linked).

Then, the three books *Death in Venice*, *Lotte in Weimar*, and *Tonio Kröger* are highlighted because they are linked to the work of Goethe.

5.4.2 Limitations :

5.4.2.1 Quality of semantic annotation

The semantic annotation of the documents has to be as rigorous as possible. We encountered in fact two issues in this regard:

Omission of quotation: This issue happens when an article A implicitly refers to an article B without explicitly linking it (through a wikiLink) cites an article B but does not signal the semantic link (in our case, the wikiLink) towards the URI of B. If the search focuses on B, logically the system should recommend A to the user, but it has not enough information to do it. Then it is the Wikipedia user who determines the achievement of the system through the quality of his contribution.

Redirects and relation of specialization The other issue concerns the redirects among pages with different URIs which refer to the same main page. This is solved by retrieving the URI identifying the main page by following the property *dbpedia-owl:wikiPageRedirects,* which explicitly asserts redirects. Nevertheless omissions remain.

Finally, there is also an issue regarding relations among articles characterized by specialization. For this issue, none property pointing out this relationship exists. For instance, it is not signaled in the DBpedia datastore that the resource corresponding to the URI *http://dbedia.org/Open-source_software* is a specialization of *http://dbedia.org/Open_source.* Then, even if the system should advice the entities linked to the concept of "Open source Software" when the main topic of search is "Open source", it cannot do it because it does not dispose of the required piece of information.

To conclude this section, the relevance of the obtained results relies on the quality of the content and the data generated by the contributors of Wikipedia and DBpedia.

5.4.2.2 Optimization of response time

This system sends n HTTP requests at the DBpedia endpoint, by iteration (n stays for the number of entities linked to the current entity, with sometimes $n > 100$).

At the moment, the response time to a single SPARQL query is about one second, and then the global response time of the system will take more than n second to be returned.

Then, in order to use the advisor, it is necessary to optimize the response time of the SPARQL endpoint. This can be done by providing an index mechanism to the system, which would allow to increase the performances. The design and the implementation of the index mechanism will be part of our future work.

Chapter 6

Conclusion and development avenues

This work has shown that Semantic Web technologies make it easier the discovery and the learning of new information, recurrent activities during Exploratory Search Tasks. More specifically, this thesis has highlighted how much these technologies are useful to guide the user all along his search. The recommendation system designed and implemented in this work is based on Aemoo, and provides an example of the usefulness of these technologies. By dynamically adapting its responses to the search path of each user, it allows him to discover new pieces of information, linked to the main topic of his search. The fundamental contribution of this approach is to take into account not just only a keyword search request, but the whole range of activities made during a search session, which can take from few minutes to several months.

Several developments can be made to this system, like allowing the user to personalize the recommendation criteria, for example thanks to the addition of filters.

But the most interesting development certainly remains the addition of a social dimension at this personalized recommendation system, by aggregating the data linked to the search behavior of each user. This would then allow

to provide advices at classes of users with similar search paths, pooling in this way the experience acquired by each one.

Finally, applying this kind of tools and systems on the Web scale will fully reveal the advantages that they can bring. Nevertheless, this requires a considerable effort of bringing semantics into existing documents as well as the opening of the silos of structured data. This effort has already been started but it has to be continued. That is why, at the moment, it is so important for the Web content publishers to follow up on the call expressed by Berners-Lee during the TED conference in 2009[1] : Raw Data Now!

[1]Video available at http://blog.ted.com/2009/03/13/tim_berners_lee_web/

Ringraziamenti

.

Innanzitutto, vorrei ringraziare il Professore Paolo Ciancarini e il Dottore Andrea Giovanni Nuzzolese per i preziosi consigli datimi durante la stesura di questo lavoro e per il tempo che mi hanno dedicato.

Ringrazio i miei genitori, che oltre ad avermi sostenuto durante questi anni, mi hanno saputo consigliare ed incoraggiare nonostante la distanza geografica che separa la Francia da questo bel paese che è l'Italia.

Ringrazio Stefano, senza il quale non avrei neanche intrapreso questo percorso.

Infine, vorrei anche ringraziare tutti coloro che mi hanno offerto il loro appoggio, e con i quali ho condiviso questa mia esperienza universitaria.

Bibliography

[AB02] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, pp. 47–97, Jan 2002.

[Bat89] M. J. Bates, "The Design of Browsing and Berrypicking Techniques for the Online Search Interface," *Online Review*, vol. 13, no. 5, pp. 407–424, 1989.

[BG04] D. Brickley and R. V. Guha, "RDF vocabulary description language 1.0: RDF Schema," W3C, W3C Recommendation, February 2004, http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.

[BHB09] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data - the story so far," *Int. J. Semantic Web Inf. Syst.*, vol. 5, no. 3, pp. 1–22, 2009.

[BLFM05] T. Berners-Lee, R. T. Fielding, and L. Masinter, "Uniform Resource Identifier (URI): Generic syntax," *Network Working Group*, vol. 66, no. 3986, pp. 1–61, 2005.

[BLHL01] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, pp. 34–43, May 2001.

[CK04] J. J. Carroll and G. Klyne, "Resource Description Framework (RDF): Concepts and abstract syntax," W3C, W3C Recommendation, Feb. 2004, http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

[CMZ05] T. Collins, P. Mulholland, and Z. Zdrahal, "Semantic browsing of digital collections," in *Proceedings of the 4th international conference on The Semantic Web*, ser. ISWC'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 127–141.

[Com13] ComScore, "2013 europe digital future in focus," 2013, http://www.comscore.com/Insights/Presentations_and_Whitepapers_2013/2013_Europe_Digital_Future_in_Focus/.

[GHS04] N. Gibbins, S. Harris, and M. Schraefel, "Applying mSpace interfaces to the Semantic Web," in *World Wide Web Conference 2004*, 2004.

[GP10] A. Gangemi and V. Presutti, "Towards a pattern science for the semantic web," *Semant. web*, vol. 1, no. 1,2, pp. 61–68, Apr. 2010.

[Gro09] W. O. W. Group, "OWL 2 web ontology language document overview," W3C, Tech. Rep., Oct. 2009, http://www.w3.org/TR/2009/REC-owl2-overview-20091027/.

[Gua98] N. Guarino, "Formal ontology and information systems," in *Proceedings of the 1st International Conference of Formal Ontology in Information Systems*, ser. FOIS '98. Trento, Italy: IOS Press, 1998, pp. 3–15.

[HHL+09] P. Heim, S. Hellmann, J. Lehmann, S. Lohmann, and T. Stegemann, "Relfinder: Revealing relationships in rdf knowledge bases," in *Proceedings of the 4th International Conference on Semantic and Digital Media Technologies: Semantic Multimedia*, ser. SAMT '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 182–187.

[JI04] K. Järvelin and P. Ingwersen, "Information seeking research needs extension towards tasks and technology," *Information Research*, vol. 10, no. 1, Oct. 2004.

[Kuh91] C. C. Kuhlthau, "Inside the search process: Information seeking from the user's perspective," *Journal of the American Society for Information Science*, vol. 42, no. 5, pp. 361–371, 1991.

[Mar06] G. Marchionini, "Exploratory search: from finding to understanding," *Commun. ACM*, vol. 49, no. 4, pp. 41–46, Apr. 2006.

[Min74] M. Minsky, "A framework for representing knowledge," Cambridge, MA, USA, Tech. Rep., 1974.

[MND+12] A. Musetti, A. G. Nuzzolese, F. Draicchio, V. Presutti, E. Blomqvist, A. Gangemi, and P. Ciancarini, "Aemoo: Exploratory search based on knowledge patterns over the semantic web," in *Semantic Web Challenge*, ser. ISWC2012, 2012.

[NGPC11] A. G. Nuzzolese, A. Gangemi, V. Presutti, and P. Ciancarini, "Encyclopedic knowledge patterns from wikipedia links," in *Proceedings of the 10th international conference on The semantic web - Volume Part I*, ser. ISWC2011. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 520–536.

[PS08] E. Prud'hommeaux and A. Seaborne, "SPARQL query language for RDF," W3C, W3C Recommendation, Jan. 2008, http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/.

[PSHS11] I. O. Popov, M. C. Schraefel, W. Hall, and N. Shadbolt, "Connecting the dots: a multi-pivot approach to data exploration," in *Proceedings of the 10th international conference on The semantic web - Volume Part I*, ser. ISWC'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 553–568.

[TB10] M. Tvarožek and M. Bieliková, "Factic: personalized exploratory search in the semantic web," in *Proceedings of the 10th international conference on Web engineering*, ser. ICWE'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 527–530.

[Tva11] M. Tvarožek, "Exploratory Search in the adaptive social Semantic Web," *Sciences, Information and of the ACM Slovakia, Technologies Bulletin*, 2011.

[WR09] R. W. White and R. A. Roth, "Exploratory Search: Beyond the query-response paradigm," *Synthesis Lectures on Information Concepts, Retrieval, and Services*, vol. 1, no. 1, pp. 1–98, Jan. 2009.