NeOn: Lifecycle Support for Networked Ontologies

Integrated Project (IST-2005-027595)

Priority: IST-2004-2.4.7 — “Semantic-based knowledge and content systems”

D4.2.1 Review of methods and models for customizing/personalizing ontologies

Deliverable Co-ordinator: **Klaas Dellschaft (UKO-LD)**

Deliverable Co-ordinating Institution: **University of Koblenz-Landau**

Other Authors: **Martin Dzbor (OU), Jose Manuel Gomez (ISOCO), Carlos Buil Aranda (ISOCO), Dunja Mladenic (JSI), Alexander Kubias (UKO-LD)**

This deliverable describes several techniques for personalization and customization which can be used during navigating and exploring existing ontologies and datasets as well as during the ontology engineering process. An integral part of this deliverable are scenarios which align the possibilities of the reviewed techniques with actual needs of the case study partners.

Document Identifier:	NEON/2006/D4.2.1/v1.3	Date due:	October 30, 2006
Class Deliverable:	NEON EU-IST-2005-027595	Submission date:	December 15, 2006
Project start date	March 1, 2006	Version:	v1.3
Project duration:	4 years	State:	Final
		Distribution:	Public

NeOn Consortium

This document is part of a research project funded by the IST Programme of the Commission of the European Communities, grant number IST-2005-027595. The following partners are involved in the project:

Open University (OU) – Coordinator Knowledge Media Institute – KMi Berrill Building, Walton Hall Milton Keynes, MK7 6AA United Kingdom Contact person: Martin Dzbor, Enrico Motta E-mail address: {m.dzbor, e.motta}@open.ac.uk	**Universität Karlsruhe – TH (UKARL)** Institut für Angewandte Informatik und Formale Beschreibungsverfahren – AIFB D-76128 Karlsruhe Germany Contact person: Peter Haase E-mail address: pha@aifb.uni-karlsruhe.de
Universidad Politécnica di Madrid (UPM) Campus de Montegancedo 28660 Boadilla del Monte Spain Contact person: Asunción Gómez Pérez E-mail address: asun@fi.ump.es	**Software AG (SAG)** Uhlandstrasse 12 64297 Darmstadt Germany Contact person: Walter Waterfeld E-mail address: walter.waterfeld@softwareag.com
Intelligent Software Components S.A. (ISOCO) Calle de Pedro de Valdivia 10 28006 Madrid Spain Contact person: Richard Benjamins E-mail adress: rbenjamins@isoco.com	**Institut 'Jožef Stefan' (JSI)** Jamova 39 SL–1000 Ljubljana Slovenia Contact person: Marko Grobelnik E-mail address: marko.grobelnik@ijs.si
Institut National de Recherche en Informatique et en Automatique (INRIA) ZIRST – 665 avenue de l'Europe Montbonnot Saint Martin 38334 Saint-Ismier France Contact person: Jérôme Euzenat	**University of Sheffield (USFD)** Dept. of Computer Science Regent Court 211 Portobello street S14DP Sheffield United Kingdom Contact person: Hamish Cunningham
Universität Koblenz-Landau (UKO-LD) Universitätsstrasse 1 56070 Koblenz Germany Contact person: Steffen Staab E-mail address: staab@uni-koblenz.de	**Consiglio Nazionale delle Ricerche (CNR)** Institute of cognitive sciences and technologies Via S. Marino della Battaglia 44 – 00185 Roma-Lazio Italy Contact person: Aldo Gangemi E-mail address: aldo.gangemi@istc.cnr.it
Ontoprise GmbH. (ONTO) Amalienbadstr. 36 (Raumfabrik 29) 76227 Karlsruhe Germany Contact person: Jürgen Angele E-mail address: angele@ontoprise.de	**Asociación Española de Comercio Electrónico** (AECE) C/lcalde Barnils, Avenida Diagonal 437 08036 Barcelona Spain Contact person: Gloria Tort E-mail address: gtort@fecemd.org
Food and Agriculture Organization of the United Nations (FAO) Viale delle Terme di Caracalla 00100 Rome, Italy Contact person: Marta Iglesias E-mail address: marta.iglesias@fao.org	**Atos Origin S.A. (ATOS)** Calle de Albarracín, 25 28037 Madrid Spain Contact person: Tomás Pariente Lobo E-mail address: tomas.parientelobo@atosorigin.com

Work package participants

The following partners have taken an active part in the work leading to the elaboration of this document, even if they might not have directly contributed writing parts of this document:

- Atos Origin S.A. (ATOS)
- Intelligent Software Components S.A. (ISOCO)
- Food and Agriculture Organization of the United Nations (FAO)
- Josef Stefan Institute (JSI)
- Open University (OU)
- University of Koblenz-Landau (UKO-LD)

Change Log

Version	Date	Amended by	Changes
v0.1	25.08.2006	Klaas Dellschaft	Creation of document and initial outline.
v0.2	31.10.2006	Klaas Dellschaft	Improved scenarios and initial reviews.
v0.3	17.11.2006	Klaas Dellschaft	Improved version of the scenarios and the algebraic operator review.
v0.4	21.11.2006	Alexander Kubias	Introduction extended.
v0.5	21.11.2006	Alexander Kubias	Executive summary added.
v0.6	24.11.2006	Klaas Dellschaft	Integrating some figures.
v0.7	25.11.2006	Klaas Dellschaft	Integrated several contributions with regard to the scenarios.
v1.0	27.11.2006	Klaas Dellschaft & Alexander Kubias	Final corrections to the deliverable. Ready for the review process.
v1.1	04.12.2006	Klaas Dellschaft	Copied the review comments to the corresponding sections in the deliverable.
v1.2	13.12.2006	Klaas Dellschaft	Incorporating the comments from the review process.
v1.3	15.12.2006	Klaas Dellschaft	Incorporating the QA comments.

Executive Summary

The customization or personalization of ontologies tries to solve the problems that arise if an ontology becomes larger and more complex. These problems are of a high importance for NeOn where several ontologies should be combined in a large network. In this deliverable several customization and personalization techniques will be reviewed.

The first technique is user profiling. It is a mechanism for collecting and representing the context of a user. For example, a user profile might contain information about which parts of an ontology are relevant for the current user and thus which parts may be hidden during browsing. Often this information is gathered by analyzing the previous user behavior. But also other approaches exist where the user explicitly states her/his preferences, e.g. by filling out a questionnaire. The majority of the current systems for user profiling exist in the area of browsing web documents. For the context of NeOn, this has to be transferred to analyzing previously viewed/constructed ontologies. Another question is in how far this collected information helps to better support the user in handling complex ontology networks.

An example where the information about the context of a user may be useful is giving a personalized view on an ontology while exploring it. There exist several techniques for exploring ontologies, like the faceted browsing. It is a means for iteratively formulating a query for finding information in a dataset. Faceted browsing is very intuitive as it uses common navigation paradigms like following links. Current faceted browsers suffer from the scaling issue as they are restricted to a few well-defined facets along which the user can navigate. This problem may be solvable to some extent by personalizing the set of facets for certain user groups, e.g. by offering only the most useful and most frequently used facets.

The faceted browsers tackle the issue of conceptual and relational complexity in datasets. But another important issue is the visualization of large datasets. There exist several metaphors which try to tackle this problem like the fish eye projection, crop circles or simply the navigation along a concept hierarchy. Requirements while exploring and interacting with large datasets are the need to find a particular item, to explore its context in the dataset and to establish differences between two or more items.

None of the techniques and metaphors for exploring conceptual complex and/or large-scale data spaces is perfect for every purpose, as all of them address different subsets of the previously mentioned requirements. So it seems more promising to use the right technique at the right point during the process of navigating and exploring ontologies. One thing which isn't covered by current techniques is the actual visualization of the ontology content in terms of what topics it covers and where this ontology is located in a larger map of topics. An initial idea is to enhance the notion of concept clouds and spotlight browsing.

Besides the customization during navigating and exploring existing ontologies and datasets, there also exists the customization of ontologies through ontology engineering. This comes in two different flavors. On the one hand, one may integrate several ontology modules into a customized network of ontologies. On the other hand, one may start with a given ontology and reduce it to parts relevant for a specific task or user group.

Customizing ontologies through ontology engineering should be based on a sound formalism like the algebraic operators. It can be distinguished between binary and unary operators, depending on the number of input ontologies. Those two categories of operators also correspond to the flavors of ontology customization identified above. The most common binary operators in the literature are the union, intersection and the difference of two ontologies. An important prerequisite for applying these operators is the availability of a mapping between the two input ontologies. Compared to the binary operators there exists less work about

unary operators. Especially, there doesn't exist common kinds of operators among the different works.

A problem of the algebraic operators, as they are described in the literature, is that a kind of best practices is missing which would help non-expert users in using those operators for a specific task. Other tools, which are specialized for a specific task and which guide the user step-by-step through performing the task, are on the other side not based on a sound formalism. Thus, an important goal would be to close this gap between the theoretical research and pragmatic implementations.

Today, one of the problems is that most of the available documents are unstructured and not machine processable. By annotating those documents with concepts coming from an ontology one makes them accessible through the above described techniques for personalization. Another view on the annotation process is that one populates an ontology with instance data. If a document was annotated with concepts coming from two different ontologies one may use this information for inferring a mapping between them, e.g. if two concepts annotate the same set of instances it may be inferred that they are equal.

There exist several annotation tools with different levels of automation. One example are tools where a human annotator defines patterns for the extraction and annotation of instances. Other tools try to learn such patterns from example documents which were manually annotated. Generally, one can say that a similar layout of the documents is required in order to get good results from an automated annotation process. With increasing heterogeneity of document layouts also the manual effort during the annotation process has to be increased. Thus, the choice of the best tool is mainly influenced by the heterogeneity and type of the documents which should be annotated.

An important aspect of this deliverable are the scenarios. These scenarios are used for mapping the requirements coming from the NeOn case studies with the described tools and techniques for ontology customization and personalization. These scenarios can be further divided into scenarios for personalization during viewing and exploring ontologies/knowledge bases and into scenarios for personalization during ontology engineering.

It is the goal of this work package to develop applications which help the case study partners to solve the problems identified in the scenarios. In most of the identified scenarios more than one of the above mentioned techniques are useful. In fact, it can be foreseen that the combination and integration of different techniques will add most of the benefits which can be achieved through ontology personalization (e.g. integrating user profiling and ontology exploration or integrating algebraic operators and annotation tools).

Nevertheless, there will be two main directions of research: On the one hand we will work on finding appropriate user interfaces and navigation paradigms for exploring ontologies. Amongst others we will work on techniques like faceted or spotlight navigation. On the other hand we will research how to support the user during ontology engineering in identifying and selecting relevant parts of existing ontologies which may then be re-used in another context and in a network with other ontologies. An important success criterion for this work is in how far we were able to improve the user experience with ontology visualization and engineering tools and whether we were able to cover the scenarios taken from the case studies.

Contents

1 Introduction **9**
1.1 Relationship to WP1 10
1.2 Relationship to WP2 10
1.3 Relationship to WP3 11

2 User Profiles **12**
2.1 Definition: What is a Profile? 12
2.2 Creating Profiles: Methods 12
2.2.1 Manual Profile Construction 12
2.2.2 Semi-Automatic Profile Construction 13
2.2.3 Automatic Profile Construction 13
2.3 Reasoning on Profiles 14
2.4 Existing Tools for the Collection of and Reasoning with Profiles 14
2.5 Looking Forward 15
2.5.1 User Profiles for Networked Ontologies 15
2.5.2 Personalized View of Networked Ontologies 16

3 Exploration Techniques **17**
3.1 Faceted Browsing 18
3.1.1 Flamenco 18
3.1.2 mSpace Framework 20
3.1.3 Discussion: Issues and Benefits 21
3.2 Large-Scale Datasets 22
3.2.1 Metaphor 1: Fish-Eye Projection 22
3.2.2 Metaphor 2: CropCircles 23
3.2.3 Traditional Ontology Visualization Techniques 25
3.2.4 Discussion: Issues and Benefits 26
3.3 Looking Forward 27
3.3.1 Large-Scale Visualization for Ontology Exploration 27
3.3.2 Navigating Ontologies vs. Datasets 28

4 Algebraic Operators **30**
4.1 Theoretical Frameworks for Operators 30
4.2 Tools for Integrating Ontologies 33
4.2.1 ONION 33
4.2.2 PROMPT 34

4.2.3 Chimaera . . . 34
4.2.4 FONTE . . . 35
4.3 Conclusion . . . 35

5 Annotation Tools **37**
5.1 Initial Requirements . . . 37
5.2 Annotation Frameworks . . . 38
5.3 Types of Semantic Annotation . . . 38
5.3.1 Manual Annotation . . . 38
5.3.2 Automatic Annotation . . . 39
5.3.3 Problems During Semantic Annotation . . . 41
5.4 Presentation of the Annotation Results . . . 41
5.5 Annotation Scenarios . . . 42
5.6 Concluding Remarks . . . 42

6 Use and Test Cases of Customization and Personalization **43**
6.1 Ontology Exploration and Integration in the FAO Library . . . 43
6.2 Customized Resolution of Redundant and Inconsistent Information in Integrated Knowledge Bases . . . 46
6.2.1 FAO Case Study . . . 47
6.2.2 Semantic Nomenclature Case Study . . . 48
6.3 Personalized View on a Network of Ontologies . . . 48
6.3.1 FAO Case Study . . . 49
6.3.2 Invoicing Case Study . . . 50
6.3.3 Semantic Nomenclature Case Study . . . 50
6.4 Ontology Integration . . . 51
6.4.1 Invoicing Case Study . . . 52

7 Conclusion **54**

List of Figures

3.1 An example screenshot of Flamenco after clicking through *Affiliation* and *US* facets in the Nobel Prize collection . . . 19
3.2 An example screenshot of CS AKTiveSpace after clicking through a geographic region and *research topic* facets in the UK Computer Science collection . . . 21
3.3 An example screenshot of SpaceTree visualization applied to an animal diversity data sets (incl. an example of how sub-trees are indicated using a triangle/arrow metaphor) . . . 23
3.4 An example screenshot of CropCircles visualization applied to an ontology . . . 24
3.5 An example screenshot of TreeMaps visualization applied to a taxonomy with coloured attribute distribution . . . 25

4.1 The assembly process of FONTE (taken from [SS03]) . . . 36

6.1 Mock-up of the initial summary of related terms and ontologies covering these terms. Typefaces reflect the trustworthiness of terms against ontologies with same typeface on the right. . 44
6.2 Mock-up of the repositioned focus of related terms and ontologies covering these terms. 45
6.3 Mock-up of a 2D rendered landscape with two ontologies broadly covering and mapping different sections of it. Green areas roughly correspond to different ontologies and red crosses to selected terms whose distance/mutual positions depend on a particular corpus. . . . 46
6.4 Parts of an invoice model and their level of heterogeneity between the models of different partners . . . 52

Chapter 1

Introduction

In this deliverable we provide a survey of methods and models for customizing ontologies. We also discuss the issues related to customizing user interaction with ontologies – whether during the design time or during application/re-use. As ontologies become more and more complex and as they are integrated into networks of ontologies, it is reasonable to investigate the means, which would be capable of making a large network of complex ontologies more manageable. The customization and personalization of ontologies includes, in principle, two areas relevant to the NeOn project:

- The customization of the view on an ontology, e.g. during exploring a network of ontologies. This customization is more or less ad-hoc and the results of the customization may be discarded once the user proceeds with exploring the ontology. This customization during exploring an ontology tries to reduce the complexity of an ontology and only show parts which are relevant for the current user.

- The customization for the purposes of reusing ontologies and integrating them into a network with other ontologies according to specific needs (e.g. during the ontology deployment, reasoning or design phases). Here the results of the customization will often be integrated into the edited ontology.

The methods and tools reviewed in this deliverable address different needs that are typically associated with the different types of the customization process. There is also a partial overlap with the other work packages of NeOn; i.e. techniques reviewed in other work packages may be used in the context of personalization and the other way round. Therefore, as a starting point, we describe in chapter 1 the relationship of this report to other work packages; in particular, we stress where the points of overlap and potential interaction can be found and which differences exist between the different work packages.

As the basis of the customization, we analyze and briefly overview user profiles in chapter 2. This chapter contains information about the user's preferences and also about the methods that may be used to perform manual, semi-automatic or automatic acquisition of such preferences. To illustrate the techniques, several tools are presented that can be used in order to build up user profiles and also to reason with those profiles.

In chapter 3 we offer a brief overview of a range of ontology exploration techniques and attempt to interpret these techniques in terms of handling large and complex structures, which are typical to real-world ontologies. All these techniques aim at reducing the size and/or the complexity of ontologies. For instance, we look at the strategy of faceted browsing, user interface modularization, visualization and navigation through large-scale datasets that subscribe to an ontology, and similarly. Since there is a large number of similar techniques, tools and methods, we do not intend to make an exhaustive overview. On the contrary, we review the key principles rather than instances.

In order to customize and personalize ontologies, algebraic operators can be used to manipulate the topology or content of the ontologies. Because of that, we provide an overview of several unary and binary algebraic operators in chapter 4 that can be found in the existing literature. Furthermore, we survey four tools partly based on a sound formalism for merging or mapping two or more ontologies.

As explained in chapter 5, annotation tools are also capable of adding useful semantic annotations. Annotations can be added not only to the documents but also to other structured or unstructured resources. In both cases, the objective is that a machine can interpret and manage the annotated information. After identifying initial requirements regarding semantic annotation, we describe two annotation frameworks. Then we classify existing annotation tools according to their type of semantic annotation.

Finally, in chapter 6 we introduce a broad test case that is followed by several smaller scenarios focusing on particular operations on networked ontologies. These scenarios, which are based upon fisheries or invoice management case studies, are included for the purposes of demonstrating possible uses of the customization and personalization techniques reviewed in this report. The objective of this final section is to explore the relevance of different types of customization techniques to the NeOn case studies.

1.1 Relationship to WP1

Work package 1 is concerned with managing networked and evolving ontologies. One particular form of creating networked ontologies is their modularization. An ontology module is seen as a tuple of imported ontologies, certain import and export interfaces, and a set of mappings. While the actual research into defining modules is carried out in WP1, there is an interaction with this work package in terms of module combination. In both work packages, we reserve the term “operator” or “algebraic operator” to describe useful ways of combining or amending the modules.

While individual operators may be restricted to different parts of module definition (e.g. only to content or to mappings), we believe that there is sufficient overlap of interests here to justify further investigation. Hence, this report could contribute to the formalization of module algebra and its operators on two levels:

- review of state-of-the-art in the ontology algebra research and
- provision of example test cases for validating operators in the future.

Moreover, the work in this work package is likely to complement the formal definitions of the ontology algebra carried out in work package 1.

1.2 Relationship to WP2

When using knowledge discovery techniques for personalization of human-ontology interaction, as addressed here, we focus on a single user (or on a user as a member of a well-defined group). On the other hand, the related work in WP2 is focused on supporting collaboration between the users, who are possibly anonymous to each other. In both cases, machine learning and knowledge discovery methods may be applied to analyze the users’ activities.

In this work package we will address the activity of the same user *in the past*, while in WP2 collaboration between the users in the networked ontology setting will be addressed. Furthermore, while in WP4 the view on the ontology will be adjusted based on the user profile, in WP2 the user will be offered some suggestions based on the actions other users performed when constructing an ontology on a similar topic or with a related content.

Furthermore, one area of potential collaboration with our colleagues in WP2 concerns the topic of *navigational patterns*. In WP2 the research is focusing on a more generic set of patterns; including those used during ontology design phase [Gan05] and those used during navigating an ontology. The latter can act as group-level summaries of preferred or trusted sequences of steps to move through or between ontologies.

The navigational patterns will typically be influenced by the current role of a user or the task at hand (see section 6.3). Navigational patterns may be used e.g. to provide an initial seed for choosing appropriate facets in an exploratory user interface or for initializing such metaphors as concept clouds, spotlights, etc. (all these are discussed in further chapters of this report).

A third field of cooperation with WP2 is upgrading database content for making it accessible with Semantic Web technologies by mapping database schemas to an ontology. This amends the possibilities of the annotation tools described in section 5 so that not only instance data from typical documents like web pages or Word documents can be extracted but also from databases.

1.3 Relationship to WP3

Customizing/personalizing ontologies via adjusting the user interface or the view on an ontology based on the user profile can be seen as bringing an ontology in the context of a particular user. Consequently, user profiles and preferences can be seen as *contextual modifiers*. There exist several possibilities how to construct a user profile (see also chapter 2) but in the work of WP3 on context sensitivity, a method for constructing the user profile based on the previously created and edited ontologies of a user will be developed and implemented. This work will extend the existing tool OntoGen [FMG06], which can be used for semi-automatic ontology construction.

In WP4, a method will be developed for using this user profile or context for personalizing the view on an ontology that is currently under construction. This method will be developed in collaboration with WP3 and it will extend the functionality of OntoGen. It will also be restricted to modifying or amending the view on the ontology instead of the semantic content of the ontology.

Providing good visualization of ontologies is an important aspect of ontology construction tools. Moreover, it was found in several user studies (including our own reported in D4.1.1 [DMA+06]) that the visualization supported by the existing tools is rather poor, being too general and providing limited support for problem-specific needs. In D4.1.1, based on the user study we suggested several possible extensions: We want to investigate visualization as a means to support comparison of different interpretations and versions of ontologies; furthermore, to apply visualization on a more abstract level than a concept (for instance, the notion of concept cloud), and also to support performing operations in a graphical as well as a textual mode.

In work of WP3 on context sensitivity, visualization support will be developed to support performing operations in a graphical as well as textual mode extending the existing tool for semi-automatic ontology construction OntoGen. In order to visualize the context in WP3, methods will be developed for placing different ontologies onto the same “base space” (which might be based on one of the compared ontologies or on a feature map of a particular domain). These strategies would allow to characterize an ontology as a whole existing within a particular space, rather than visualizing its individual concepts in a form of a RDF graph.

In work of WP4, we will investigate the conceptual as well as the topographic characteristics of such visual representation of several ontologies. We will also investigate the usage of the visualization methods to support user-specific views on the set of ontologies via changing the concept landscapes based on the user profile. Here we would like to use similar ideas as used for visualizing a set of documents in the Document Atlas [FMG05]. Based on that, we will further investigate the usage of such visualization for positioning a part of the ontology within a larger landscape (e.g. a corpus), for finding ontologies that cover a particular group or cloud of concepts, for formulating a navigation query by spotlighting the cloud, generalizing or otherwise adapting the cloud, exploring different ontological states of the items in the cloud, navigating the cloud alongside some identified dimensions (e.g., regionalize, generalize, find similar).

Chapter 2

User Profiles

2.1 Definition: What is a Profile?

User profiles are seen here as a way to describe some user properties or characteristics and thus as a representation of the context of a user. Such a profile may for example provide information about the role of a user, the domain of interest or the current task. This information about the context helps in a user-tailored information delivery, e.g. by offering personalized ontology views or supporting the user in combining different data sources (see also the scenarios in chapter 6).

When talking about the user, it is important to mention that we can decide to have an *abstract user* – this would be, in principle, corresponding to any member of a group of users in a particular situation. The same user can belong to different groups depending on the task at hand. For instance, in the fisheries domain such roles as editor, validator or viewer were identified. The profile should be operational; this means that it should enable its usage in different settings depending on the specific problem. We commonly need to compare the user to the profile by measuring the similarity between them; similarly, we may compare two profiles or measure similarity of the user to a stereotype (a profile of stereotypical, abstract user) based on a group of similar users.

A user profile can be constructed in different ways depending on the data it includes and the methods used for its construction. In the following section we briefly describe different classes of methods of profile acquisition and construction including manual, semi-automatic and automatic methods. Each of them has some advantages and disadvantages. While manual methods have no problem with providing benefit to the new users, automatic methods can only make guesses. Hence, automated methods may sometimes provide very limited or no benefit to the new user, while they are waiting to collect enough data for automatically constructing a reasonable user profile. On the other hand, one of the important dimensions of the user profile is also its adaptation to the changes over time, which may be important to some applications (e.g., recommending clothes or movies to the user). The adaptation requires updating of the user profile, which is easier if we have automatic methods for user profiling.

2.2 Creating Profiles: Methods

2.2.1 Manual Profile Construction

A simple way to construct a user profile is to ask the user to provide some preferences by filling in a questionnaire; for instance, to tick-off the topics of interest, to provide demographic data, etc. This is especially useful for new users, where there is no data available on the past actions/interests of the user and consequently the systems for automatic user profiling cannot be applied. A major disadvantage of this approach is that it works well if the profiling dimensions are (i) well-defined, (ii) not too many in number, and (iii) known in advance and unlikely to change.

If the number of potential profiling dimensions grows larger or users vary in how they interpret any particular dimension, manual profile acquisition may be a daunting and fairly time-consuming activity. Nevertheless, it is a popular and simple way of acquiring a rough idea of what users prefer – as can be seen by a frequent use of various “Options...” or “Preferences...” items in the pull-down menus of many standard software packages.

2.2.2 Semi-Automatic Profile Construction

Semi-automatic methods for creating a user profile combine manual methods with automatic methods. The combination can be performed in several different ways, depending on the level of detail acquired in the manual profiling and on the stages when it is performed. A common way would be constructing the initial user profile manually by asking a few simple, well-defined questions, and then update and maintain this seed profile automatically as the system starts collecting data about the user and his/her interaction with the system. An automatically constructed profile can be then periodically shown to the user for manual adjustments. For instance, the user can be asked about class preferences and the instances can be obtained to match the user preferences.

An alternative way is to use active learning as a semi-automatic learning method. For instance, in order to construct a user profile capturing the user’s preferences on some items, the system can generate the profile by asking the user to manually label some, carefully selected items as interesting or non-interesting. The careful selection of items enables generation of a user profile with limited amount of manual work. A similar strategy has been used e.g. in the domain of text annotation [VVMD$^+$02], where the MnM annotation tool learned from a few manual annotations performed by the user about his/her preferences to interpret a particular sequence of terms.

2.2.3 Automatic Profile Construction

Automatic methods for user profiling automatically construct a user profile based on the user’s interaction with the system. They often complement this automated part with collecting explicit or implicit feedback from the user. Explicit feedback may be in asking the user to provide a rating on the suggested items (e.g. movies or books) or to simply confirm that s/he has found the information while using our system for search (see e.g. Amazon.com’s statement “I found this review helpful” next to each item review). Implicit feedback is collected without putting any additional work on the user. For instance, by assuming that all the documents from a digital library that the user has viewed for more than 30 seconds were interesting or, that all hyperlinks the user clicked on a particular web page are interesting for the user while the other hyperlinks from the visited web pages that the user did not click on are not interesting for the user. Regardless of the kind of feedback, these methods are usually based on machine learning techniques [MC$^+$94].

A known example of applications that rely on user profiling is recommending items to the user based on estimating the relevance of items (e.g. books) to the user (and his/her interests). In the context of recommendation systems we usually talk about *memory-based* or *model-based* methods. While model-based methods explicitly construct a model that represents the user profile, the memory-based methods just store the data about the user. If that data was obtained automatically via explicit or implicit user feedback on the recommended items we consider it here as automatic profile construction.

An example of the memory-based machine learning methods is k-nearest neighbor (sometimes also referred to as lazy-learning). In the collaborative recommendation setting it stores all the data with the user feedback into a memory and predicts relevance of an item to the user by finding k other users that are most similar to the user. Then the method applies the ratings of those neighbors on the item to get the relevance estimate. In the content-based recommendation setting, the technique would store all the items (e.g., documents) marked as relevant or non-relevant for each user separately. Relevance of a new item to the user is estimated by finding k other items that are the most similar to the item in hands and using their ratings to get the relevance estimate.

On the other hand, model-based methods generate a model that represents a user profile, as for instance a decision tree or a support vector machine. The profile can be based on the content of the items or on the ratings of the items provided by different users. On a request for estimating the relevance of an item to the user, the model representing the user profile automatically assigns the relevance to the item based on its match to the user profile.

2.3 Reasoning on Profiles

The user profile can be used in the analysis of the users providing some insights into the population using the system but more importantly the user profile is used to provide some benefits to the user. Based on the current user profile, we may change some action/interface of the system, meaning that we have a possibility to influence user's further actions based on the current profile of that user.

For instance, in recommendation systems, a user profile can be used to help the users browsing the web by classifying web pages against the profile to find those matching the profile (content-based recommendation). It can also be used to modify the web page that the user has requested by adding that recommendations. The user profile can help to compare different users and to share knowledge between them. For example, instead of having a friend with similar interest sending me addresses of interesting web documents, the system can automatically suggest documents that were found interesting by other users that have interests similar to mine. In order to secure the privacy, only knowledge and not the user identity can be exchanged or the cooperation could apply only to users that explicitly agreed to take part in knowledge sharing. This sharing of knowledge is related to the collaborative approach of recommendation systems.

An alternative is to have a predefined description on the actions to be applied by the system depending on some predefined user profile characteristic. For instance, in tutoring systems, the system can have a set of predefined user interface layouts for some basic types of the users, e.g. novice user, knowledgeable user, expert user. Each of them can be further adjusted based on the current user's interest or activity.

2.4 Existing Tools for the Collection of and Reasoning with Profiles

There is a number of systems and tools that were developed for user profiling. Here we provide a short description of several systems just to illustrate the different approaches to user profiling described in 2.2.

Lifestyle finder [Kru97] is an example system that is based on a manual approach to user profiling. It forms a user profile from demographic data provided by the user via a questionnaire. The system is a collaborative recommendation system as it groups similar users based on the similarity of their manually constructed user profiles. It recommends potentially interesting web documents to the user based on the ratings of the documents provided by similar users using the memory-based approach for collaborative filtering.

An example of the semi-automatic approach to user profiling is via using OntoGen [FMG06] which is a tool for semi-automatic ontology construction. OntoGen gets as an input a collection of documents provided by the user. The user profile is then constructed in interaction with the user, by grouping documents into a hierarchy based on their content similarity. The approach is based on the previously proposed idea of capturing interest of the user into a topic hierarchy that is automatically constructed from the web documents visited by the user. The same idea was also used in an automatic user profiling setting to enable the visualization of the interest-focused browsing history [GMG05] of the user.

Automatic user profiling can be performed on different types of data, not necessarily text. Calendar Apprentice [MC^{+}94] is a system that helps the user to schedule meetings. The system is connected to the user's electronic calendar and using machine learning methods it generates sets of rules capturing the user's scheduling preferences and some other information about the individual attendees of the meetings (e.g., is the attendee of the meeting a student, a faculty dean, etc.). It uses these rules to provide advice to the user for scheduling new unscheduled meetings suggesting the duration, the location, the day of the week, and the time of the meeting. The user profile is generated by learning a decision tree for each of the listed features, as a method that produces models that are intelligible to humans. This enables the user to inspect, edit or approve the learned model, if we want to go in the direction of semi-automatic user profiling. As the user is interacting with the system the user profile is updated (e.g., every night) and the new learned rules are integrated with the existing rules based on their performance statistics.

Another example of an automatic user profiling system for recommending clicked web documents is Personal WebWatcher. Personal WebWatcher [Mla96] uses machine learning methods in order to learn a separate model for each user and uses the learned model for highlighting the promising hyperlinks on the requested web documents. The system is used on a local machine of the user in the following setting. The user is on one end and the web is on the other end, between them is Personal WebWatcher acting as a proxy server. It consists of: the *proxy* that gets HTTP requests from the browser and fetches the requested web page; the *adviser* that gets the original web page and extracts the hyperlinks from it and composes the modified web page by highlighting the promising hyperlinks based on the scores assigned to all the extracted hyperlinks; the *classifier* threats each extracted hyperlink as an example and uses the induced model of the user interests to assign a score to each of them; and the *learner* gets a collection of the visited documents and induces a model of user interests based on the web documents. In Personal WebWatcher, browsing the web is supported by highlighting promising (i.e., interesting) hyperlinks on the requested web documents. The assumption is that the interesting hyperlinks are the hyperlinks that are highly probable to be clicked by the user. In the machine learning setting the problem is defined as predicting clicked hyperlinks from the set of web documents visited by the user. All hyperlinks on the visited documents are used for constructing machine learning examples. Each is assigned to one of the two class values: positive (the user clicked on the hyperlink) or negative (the user skipped the link).

Another example is NewsWeeder [Lan95], a system for electronic Usenet news altering that uses machine learning methods to generate a user profile. The system uses a World Wide Web interface to enable the user to access the Usenet news in a usual way and to enable the system to collect user's ratings as feedback. NewsWeeder additionally assigns a predicted rating to each article and shows the user a personalized list of the most interesting articles.

2.5 Looking Forward

The possible usage of user profiling methods in the NeOn case studies is described in chapter 6. To summarize their benefits, we see at least two roles of user profiling: one is supporting the combination of different data/knowledge sources and the other is offering a personalized view on the ontology.

User profiling is one of the important aspects for supporting customized human ontology interaction. User profiles can be used for customization when combining different data sources as outlined in section 6.2, where the preferences for a data source are based on the user profile that can be provided manually and also later adjusted using machine learning techniques based on the user's activity. The potential approach for using user profiling for networked ontologies is described in section 2.5.1.

User profiling can also be used for providing a personalized view on an ontology based on the ontologies previously constructed by the same user as outlined in Section 6.3. The idea is to use machine learning and knowledge discovery methods to analyze the user's past activities. Such a personalized view on ontologies based on the profile of a user can be seen as bringing in the context of this user. The potential approach for using personalized views of networked ontologies is described in section 2.5.2.

2.5.1 User Profiles for Networked Ontologies

In work of NeOn:WP3 on context sensitivity, a method for constructing the user profile based on the ontologies previously constructed by the user will be developed and implemented as an extension of OntoGen, an existing tool for semi-automatic ontology construction [FMG06]. The idea is to automatically build a user profile based on the previous work of the users. This is done by recording previous choices that the user made when constructing ontologies and using them as an extra input to provide a personalized view on the underlying data and the ontology constructed so far through "personalized word weighting schemas" (instead of using predefined schemas, such as Term Frequency Inverse Document Frequency (TFIDF)).

In work of NeOn:WP4, we will develop a method for using this user profile (providing context of the user based on the past user's activities) for adjusting the view of the user on the ontology under construction. The method developed in WP4 will be implemented in collaboration with WP3 inside OntoGen.

OntoGen[1] [FMG06] is a system for semi-automatically constructing topic ontologies from a collection of documents. During the ontology construction process, the system provides suggestions for topics (ontology concepts), assists by automatically assigning documents to topics or suggesting names for topics. The user can use those suggestions but s/he may also decide to e.g. further split or refine concepts or to move concepts to another place in the ontology. The system also supports the extreme case where the user can ignore suggestions and manually construct the ontology. All those design decisions of the user can be recorded for constructing a user profile. Additionally, one may infer the domain of interest of a user from the data s/he provided as input for semi-automatically constructing the ontology.

2.5.2 Personalized View of Networked Ontologies

As visualization is an important part of handling ontologies, the system called Document Atlas (for visualizing larger collections of documents) is loosely integrated into OntoGen. This helps the user to comprehend and to understand the topics covered by the instances inside a specific concept. This is done by visualization the instances using the Document Atlas tool [FMG05]. Document Atlas is a tool for creating, showing and exploring the visualizations of text corpora. The documents are presented as points on a map and the density is shown as a texture in the background. Most common keywords are shown for each area of the map. When the user moves the mouse around the map a set of the most common keywords is shown for the area around the mouse (the area is marked with a transparent circle). The user can also zoom-in to see specific areas in more details.

In work of NeOn:WP3 on context sensitivity, visualization support will be developed to support performing operations in a graphical as well as a textual mode extending the existing tool for semi-automatic ontology construction OntoGen. In order to visualize the context in WP3, methods will be developed for placing different ontologies onto the same space based on characterizing ontology as a whole within a particular space rather than visualizing its individual concepts.

In work of NeOn:WP4, we will investigate conceptual as well as topographic characteristics of such visual representation of several ontologies. We will also investigate the usage of visualization methods to support a user-specific view on the set of ontologies via changing the concept landscapes based on the user profile. Here we would like to use similar ideas as used for visualizing a set of documents in Document Atlas [FMG05]. Based on that, we will further investigate the usage of such visualization for positioning a part of the ontology within a larger landscape, finding ontologies that cover the found group/cloud of concepts/tags, formulating a navigation query by spotlighting the cloud, generalizing or otherwise adapting the cloud, exploring different ontological states of the items in the cloud, navigating the cloud alongside some identified dimensions (e.g. regionalize, generalize, . . .).

[1]OntoGen is available as a free download from `http://ontogen.ijs.si`.

Chapter 3

Exploration Techniques

One of the key characteristics of the Semantic Web when compared to the Web is the emphasis on separating the content from its presentation. Unlike mark-up language for the Web (e.g. HTML), which focus on how information should be visually presented to the user, Semantic Web languages are in principle presentation-free. This has a major advantage - information and knowledge becomes comprehensible to computers and artificial agents. However, it also has a major disadvantage - any semantically annotated chunk of formal knowledge needs to be “transformed” into a form or shape comprehensible by a human user.

Such a transformation of Semantic Web content, which includes formally represented ontologies, is rarely straightforward. First, datasets typical for the Semantic Web are relatively large; they contain thousands of statements the user may need to interact with. For example, a fairly simple geographic ontology of regions in New York state[1] contains as many as 59,000 unique statements just about congressional districts in a single state. The challenge of large open information spaces is in providing a simple, yet effective way of finding, sorting and viewing relevant information. Second, ontologies that conceptually underpin some Semantic Web applications could be complex structures representing different types of relationships. If each of such potential relations is treated as a dimension in which allowed values could be depicted, then even a moderately complex ontology leads to a multi-dimensional space, which poses challenges for navigation and interaction - in particular, when human cognition naturally prefers (and is able of coping with) two or three dimensions.

One of the most successful strategies for handling large and/or complex conceptual spaces involves their reduction. An alternative way to handle complexity and/or size is an appropriate projection. The key difference between these two strategies can be put in the following terms. Where reduction is concerned with *showing less* (in our case, perhaps fewer concepts, entities or relationships), the projection tackles the same issue by showing the same set of concepts, entities and relations *differently*. The two strategies are somewhat complementary, and although there are typical techniques for size reduction and typical techniques for projections, there is also a certain amount of functional overlap between them. In other words, a reduction or projection technique (such as e.g. a sequence of facets – for more details see section 3.1) may benefit from some form of annotation to express how a particular technique can address a particular task. These “technique annotations” may then be used e.g. to recommend a user interface module or to associate certain patterns with a certain user or user group (see also section 2).

In the following we review the roles of several approaches to reducing size and complexity, as well as approaches to amending projections/views on the ontologies and datasets. In particular, we concentrate on those approaches that are applied to (or at least applicable to) ontologies. We look at the strategy of using so-called faceted browsing and navigation, user interface modularization, and visualization and navigation through large-scale datasets that subscribe to an ontology.

[1]One serialization of this ontology (data set) can be found at `http://www.daml.org/2003/02/fips55/NY.owl`

3.1 Faceted Browsing

The notion of faceted browsing is not new. In fact, everyone has at some point come across some form of facets – only, we don't usually call it this way. Consider, for example, department stores or their mail order catalogues. The products have to be shown according to some patterns meaningful to the customer; if they were completely random, we would not be able to buy anything in any larger store. Those patterns reflect broad category of products, e.g. clothing vs. food. Once we get into clothing department, we can shop by designer labels, and look for fashion by Gucci vs. Levi's. Alternatively, we may want to shop by occasion – e.g. sportswear vs. casual clothing. Yet another view on the same collection is by the target shopping audience – we have clothing for boys vs. girls, men's fashion separate from women's fashion. Sometimes we take into account specific merchandise sponsors the clothing relates to – e.g. "Bob the Builder" collection vs. "Tiger Woods" collection. And sometimes it is easier to find the right piece of clothing by checking an alphabetic index. These different shop navigation strategies help us get to the right piece of clothing.

The same idea has been adopted by information processing tools. Large datasets – for example, art collection in galleries, book collection in libraries or university courses – have many dimensions according to which they can be viewed, browsed, searched or navigated. Thus, faceted navigation is an interaction style whereby users filter an appropriate set of items by progressively, step-by-step selecting from valid dimensions of a particular classification. That classification can be created according to many principles. In case its dimensions are orthogonal (e.g. designer labels is orthogonal to social occasion), we are talking about faceted classifications. In case we have nested dimensions (e.g. boys clothing – age group 12yr), we talk about hierarchical classifications. In the user interaction (UI) terminology, these dimensions are referred to as facets.

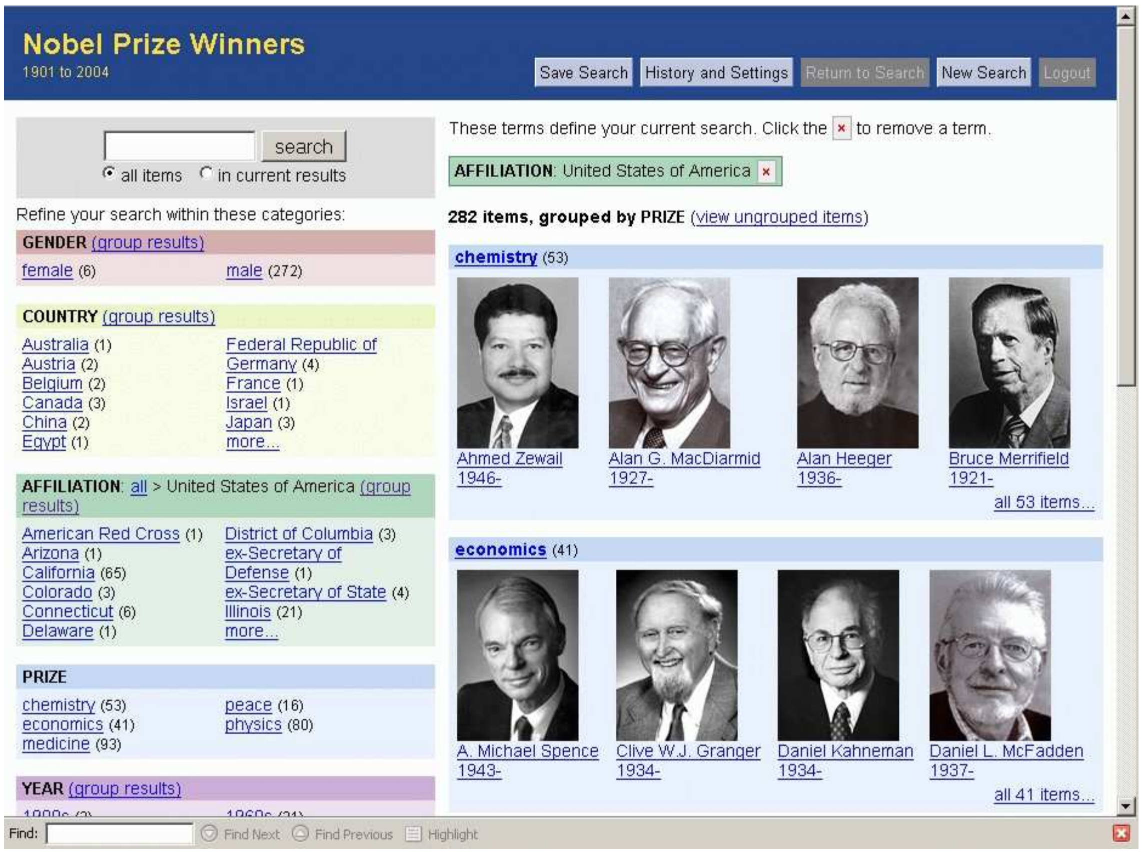
Some examples of strategies used in faceted navigation include tools like Flamenco – a portal for browsing fine arts collection, Epicurious – a recipe browser, and CS AKTive Space – an access point to a semantic repository about UK's computer science. There are obviously many more examples of faceted browsing/navigation but these are sufficiently informative for the purpose of reviewing the key functional aspects of this user interaction method. From the more recent ones, we shall mention e.g. Longwell and Fresnel [PBKL06] from MIT's Simile project as representatives of generic frameworks and vocabularies (respectively) for faceted navigation through RDF collections and for specifying facets. Other recent examples include BrowseRDF [ODD06], a generic RDF browser, or `/facet` [HvOH06], an RDF browser used in a manner similar to Flamenco, but in the context of Dutch cultural heritage project.

3.1.1 Flamenco

Flamenco – FLexible information Access using MEtadata in Novel COmbinations – is a work done by Marti Hearst's research team at the University of California in Berkeley. As a standard faceted browser, it allows users to move through large information spaces, and as the authors add "in a flexible manner without feeling lost". Flamenco exposes metadata about item categories, and these are used to guide the user through next steps, possible choices, or simply to organize a large underlying collection. Technically, Flamenco uses multi-layer facets; i.e. dimensions are not mutually exclusive as in our shopping illustration, but could be hierarchically nested. Such a strategy allows users to not only view the collection from different angles, but also allows refining user queries. An example of a typical user interface of Flamenco applied to a collection of Nobel Prize winners' biographies is shown in Figure 3.1[2].

As the authors of Flamenco applications emphasize (e.g. in [YSLH03], [Hea00]), faceted browsing is in principle an alternative way for searching and browsing large data collections. More traditional approaches include keyword-based (as well known from any search engine: Google or Yahoo) and, less frequently, example-based. The latter strategy is commonly used in image navigation and search - e.g. finding images with the same face or house. What Flamenco adds is an opportunity to navigate not alongside the actual content of the items in a collection, but alongside conceptual meta-data associated with those items.

[2]Picture re-used from `http://flamenco.berkeley.edu/example\_screenshots`


Nobel Prize Winners
1901 to 2004

Save Search | History and Settings | Return to Search | New Search | Logout

search
all items | in current results

Refine your search within these categories:

GENDER (group results)
female (6) male (272)

COUNTRY (group results)
Australia (1) Austria (2) Belgium (2) Canada (3) China (2) Egypt (1) Federal Republic of Germany (4) France (1) Israel (1) Japan (3) more...

AFFILIATION: all > United States of America (group results)
American Red Cross (1) Arizona (1) California (65) Colorado (3) Connecticut (6) Delaware (1) District of Columbia (3) ex-Secretary of Defense (1) ex-Secretary of State (4) Illinois (21) more...

PRIZE
chemistry (53) economics (41) medicine (93) peace (16) physics (80)

YEAR (group results)

These terms define your current search. Click the × to remove a term.

AFFILIATION: United States of America ×

282 items, grouped by PRIZE (view ungrouped items)

chemistry (53)
Ahmed Zewail 1946- | Alan G. MacDiarmid 1927- | Alan Heeger 1936- | Bruce Merrifield 1921-
all 53 items...

economics (41)
A. Michael Spence 1943- | Clive W.J. Granger 1934- | Daniel Kahneman 1934- | Daniel L. McFadden 1937-
all 41 items...

Find: Find Next Find Previous Highlight

Figure 3.1: An example screenshot of Flamenco after clicking through *Affiliation* and *US* facets in the Nobel Prize collection

While Flamenco might not perform very well with respect to speed of retrieving an image from a collection - the core tests have been made on the fine arts collection of 35,000+ items, it seems to be much better way for actually learning something about the collection, its structure, and in general, about the art pieces. Obviously, classifications are very often fairly simple ontologies, nevertheless, the ideas of faceted browsing do apply to navigating also more complex conceptual structures. Another interesting thing that makes Flamenco work interesting for navigating ontologies is the approach to creating and managing the metadata classifications. In Flamenco applications, the actual metadata descriptors have been i) mined from the actual text labels of the art pieces, and ii) normalized into an ontology-like structure.

From a user interaction perspective, Flamenco design exhibits a neatly structured interaction with its user. The interaction has several stages - an overview of possibilities at the beginning of navigation is (typically) followed by a series of manipulations with the retrieved results, with the original queries or both. The purpose is to induct the user into the collection and offer rapid rewards early into interaction. This often leads to many results matching the query, so they may need to be organized according to selected categories (e.g. locations, styles, themes, etc.). Alternatively, the user may narrow the view down by referring to hierarchical classification (if available). The navigation may end with accessing a particular item from the collection. We use term “may” because alongside the item the user always sees all other categories and metadata that provide bridges to alternative collections.

Flamenco application in fine arts (in particular) re-uses the faceted and hierarchical classifications also for another purpose. In addition to enabling navigation into the collection through different paths, it also features the notion of lateral links connecting different parts of the collection. The notion of lateral navigation is particularly relevant for the complex conceptual structure such as ontologies - the reason being that there may be multiple relations between different concepts or objects in the ontology and using one relation for browsing may hide the other. The lateral navigation is complementary to the hierarchical or faceted one (which typically occur "within topics") chosen by the user. This notion has been elaborated in more depth e.g. by [BR02] who refers to it as horizontal navigation "between topics" or between different sub-spaces.

3.1.2 mSpace Framework

A slightly different view on the principle of faceted navigation is advocated by the authors of CS AKTive Space (and a family of related applications). The common feature in this family is their conceptual subscription to the mSpace theory [SSO+05] – a theory seeing Semantic Web as a rich, multi-dimensional hypertext system. One of the first applications, which is used as a basis for the review, was CS AKTive Space [SGG+04]. As the authors point out, the faceted views and multi-pane browsers are in the mSpace applications extended by the functional operations such as slicing, sorting, swapping, adding, subtracting, and similarly.

Rather than developing a faceted or hierarchical classification for each application separately, mSpace (and CS AKTive Space) rely on the ontological knowledge and use this knowledge as a basis, on which the above-mentioned operations could be realized – regardless of what is the domain or scope of the application. Essentially, mSpace methodology represents a formal view of the above operations, which enables their use on any organized structure – including ontologies (but also database schemas, etc.)

mSpace, similarly as Flamenco before, conceptually facilitates exploration of a complex semantic space – this usually means knowledge or data-bases populated with a large number of records/statements, but it also applies to ontologies per se. In mSpace, exploration is seen as the capability offered to the user to make information selections within a structured domain [SCRH04]. These selections could either declared or inferred from the underlying conceptual model.

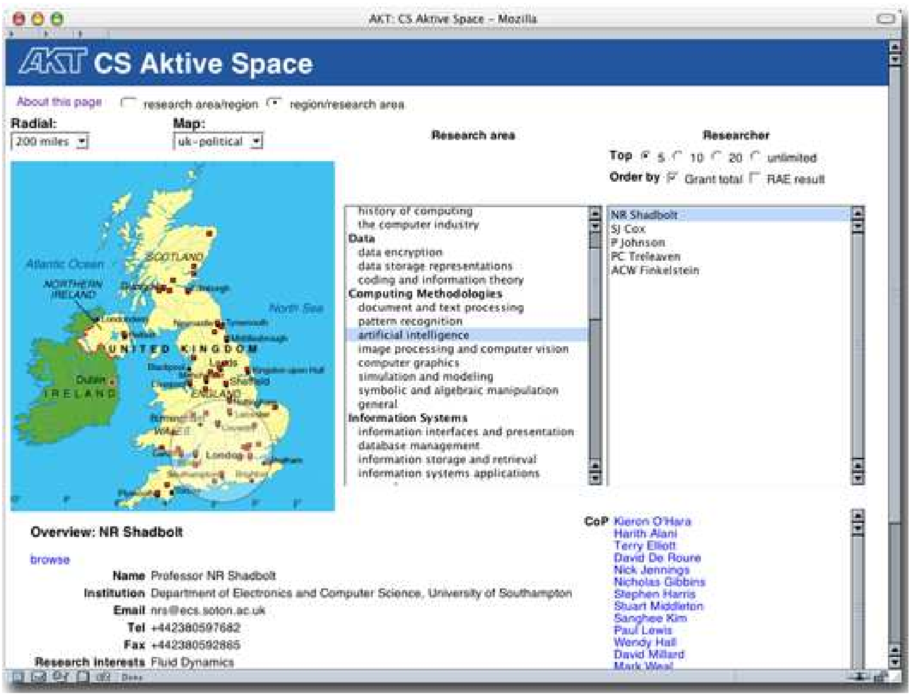
mSpace-based applications, e.g. CS AKTive Space, use the ontologies as follows. First, they allow a standard faceted browsing induction (similar to Flamenco's overview); i.e. the user may choose how to start an interaction. In CS AKTive Space one can choose to start with *a spatial view* – a region in a traditional geographic map, a topic-based or a university-based view (in the first slice of multi-dimensional space). At the same time the user sees all other slices that have not been used so far – assuming we started with geographic, regional selection, the other slices would include e.g. topic view and institution view. The selection made in the first slice (in our case regional) only emphasizes the items in the next slice that are semantically relevant to the selection made in the earlier slice. In other words, CS AKTive Space does not attempt to generate the next level of facets every time a user chooses a new facet. On the contrary, items in other slices are visible, facet selection merely makes them "stand out" (e.g. by highlighting).

One useful side effect comes from combining the visual facets (such as e.g. spatial, geographic views) with the textual facets (e.g. topic hierarchies or university lists). First, spatial navigation can express many relations very succinctly (incl. the fuzzy ones such as *near* or *in the South*). Second, the notion of spatial representation of some conceptual data is a strategy commonly used by many users. As e.g. authors in [MS97] point out that given a pile of standard printed articles, prints or pictures, people tend to space them out – e.g. into smaller piles on a desk or into separate boxes. However, people not only create subsets of materials, they also tend to express some form of (semantic) closeness between different piles; e.g. by locating them next to each other. Yet another spatial clue they observed was an attempt to express some kind of ordering within each of the smaller stacks.

In line with their argument, such spatial stacking of information conveys certain preferred navigation strategies – for instance, items closer to hand are interpreted as "more important", items on one side are "recommendations to a manager" as opposed to the other side, where they put "to-be-looked-at-later materials". Interestingly, the experiment in [MS97] showed that people are capable of devising a variety of ways to communicate their intentions, preferences and recommendations w.r.t. navigating a complex space. The mentioned experiment has been with a tool called VIKI, a spatial hypertext navigation prototype [MS97].

Since then more work has been done in deploying the spatial element in navigating through (mostly) hypertext. For example, Mancini carried out a study and some experimentation with the notion of cinematic hypertext [Man05]. In this book Mancini presents and compares how they same content may yield different interpretation if presented (and navigated) in a spatially different manner. Nevertheless, the use of such techniques needs further research before we are able to link them to particular use case scenarios and requirements.

Returning back to faceted browsing, the authors in [SCRH04] argued that beyond arranging the domain/ontology in different views, mSpaces associate domain information with selected instances. This is referred to as first-level zoom, and its purpose in the application is that each selected attribute may call up all information relevant to the selection made in earlier slices. This zoomed information is not a part of any other facet, but in the world of complex ontologies this is a valuable addition. In principle, this strategy distinguishes between some relationships that are so dominant as to be associated with respective facets, and other relationships that are less-dominant but nonetheless informative and/or discriminative. In case of CS AKTive Space, these zoomed relationships include e.g. projects known in a given region and on a given theme, or grants awarded, people's communities of practice, and many more linked information. An example screenshot of mSpace applied to UK's computer science domain is shown in Figure 3.2[3].



history of computing
the computer industry
Data
data encryption
data storage representations
coding and information theory
Computing Methodologies
document and text processing
pattern recognition
artificial intelligence
image processing and computer vision
computer graphics
simulation and modeling
symbolic and algebraic manipulation
general
Information Systems
information interfaces and presentation
database management
information storage and retrieval
information systems applications

NR Shadbolt
SJ Cox
P Johnson
PC Treleaven
ACW Finkelstein

Overview: NR Shadbolt

Name	Professor NR Shadbolt
Institution	Department of Electronics and Computer Science, University of Southampton
Email	nrs@ecs.soton.ac.uk
Tel	+442380597682
Fax	+442380592865
Research Interests	Fluid Dynamics

CoP Kieron O'Hara
Harith Alani
Terry Elliott
David De Roure
Nick Jennings
Nicholas Gibbins
Stephen Harris
Stuart Middleton
Sanghee Kim
Paul Lewis
Wendy Hall
David Millard
Mark Weal

Figure 3.2: An example screenshot of CS AKTiveSpace after clicking through a geographic region and *research topic* facets in the UK Computer Science collection

The model followed by mSpace-based applications supports two levels of user interaction: manipulation of the ontology representation itself, and selection of the instances of the data associated with those configurations. The logic of the model also provides for automatic reasoning across the domain to ensure that only meaningful attribute orderings/selections can occur [SCRH04].

3.1.3 Discussion: Issues and Benefits

In general, what faceted browsers like Flamenco support rather well is the iterative formulation of the search queries or navigational goals. Unlike standard search engines that often expect user to render their query on the first attempt and do not provide major support for query re-writing, faceted browsers are particularly adept at supporting exploration.

What the researchers working on faceted browsers and navigational tools emphasize as the key advantage of their technology is the step away from forcing the user to go through deep, complex hierarchies in order

[3]Picture re-used courtesy of `http://triplestore.aktors.org/demo/AKTiveSpace/`

to find items they are interested in. The best example of such hierarchy-based interaction is the file system structure of folders and sub-folders that in some operating systems may become so deep that it is unwieldy and almost unusable. An alternative offered by faceted navigation is an iterative filtering – users navigate to the next slice by following some conceptual clues (e.g. sub-categories or orthogonal views). Arguably, faceted navigation seems to be a more natural way of coping with messy, conceptually complex space than a rigid, hierarchical folder-like structure.

An important feature of faceted navigation is that it is in no way different from a common navigation by means of following links, to which a majority of web users is accustomed. Faceted navigation is thus one of relatively simple ways to transform the heavy semantic representations that are typical for ontologies into slices and chunks humans can cope with. Thus, the strategy of “dividing and conquering” also works in the context of complex conceptual spaces such as ontologies. What is hard to visualize at once because of variability and differences between different relationships, can be split into sequences of partial visualizations through which it is easier to move and which are also more comprehensible to the end user.

On the other hand, faceted browsers suffer from the scaling issue; i.e. they work reasonably well with a few well-defined facets that can be arbitrarily combined by the end user. For instance, in CS AKTive Space there were three key (i.e. navigable) dimensions (location, topic and institution). In Longwell deployed for MIT OpenCourseWare[4] there are similarly three key dimensions (level of study, course team member and keywords). To some extent, this issue is resolvable by customizing the user interfaces and opting for the most useful or most frequently used facets. Yet, what is useful to one group of users may not be as helpful for others. So there is an ongoing tension between offering as many facets to the user as possible while simultaneously helping to reduce navigational complexity.

In this section we identified several approaches and outstanding challenges to using faceted views (whether temporal, spatial or combined) to achieve different projections of the data sets. However, one of the generic issues with faceted browsing as an approach is still its underperformance in terms of coping with large numbers of views/facets. Hence, in the next section we consider the implications of dealing with large-scale datasets, and in particular, we will review some techniques relying on the semantic structure of the dataset(s) in question.

3.2 Large-Scale Datasets

In addition to the conceptual and relational complexity that has been tackled by the research into faceted navigation, another similarly hard task is to navigate through large datasets. While one ontology may not be an extraordinary large entity, in NeOn we are dealing with networks of multiple ontologies. Joining several, apparently simple ontologies may lead to much larger networked structures, and the size of the structures may further increase thanks to operations of mapping and reification [Mik05]. Hence, another perspective reviewed would draw upon the work done by the visualization community, especially with respect to handling large datasets.

3.2.1 Metaphor 1: Fish-Eye Projection

Fish-eye projection is one of the navigation and user interaction metaphors familiar to all interested in photography, but also to car drivers. This technique is used, for instance, in [KS03] to fit large semantic networks into limited-size display and in parallel to make them user-navigable. The fish-eye metaphor enables customizable navigation; it uses different properties of the objects in a knowledge base to create clusters of different granularity and of different semantic meaning. For example, authors in [KS03] illustrate this capability to handle large networks of diverse but conceptually related data in the context of visualizing the 200k strong student population of The Open University in the UK, which can be shown on a geographic, per-faculty, per-program or per-course basis.

[4]See `http://simile.mit.edu/longwell/demo/libraries/`

The strategy relies on showing the contextual fringe of a part of the semantic network not corresponding to a particular user's query or intention using more coarse-grained clusters than the part which actually corresponds to the query and is currently in focus. The authors also open up the context-focus metaphor [LRP95] so that each particular focus (fine-grained view) can be embedded into an arbitrary context (coarse-grained view). This approach promotes the modularity of an application and allows for actually using Semantic Web technologies to capture the context-focus relations. Indeed, the application enabling navigation through the current student population of The Open University goes as far as having an independent representation of visual structures from the original semantic and conceptual assertions. It could be argued that this "transformation" from a formal representation to the representation of visual/navigational relations is akin to translating the meanings of semantic relations to a language the user is more familiar with (e.g. distribution sets).

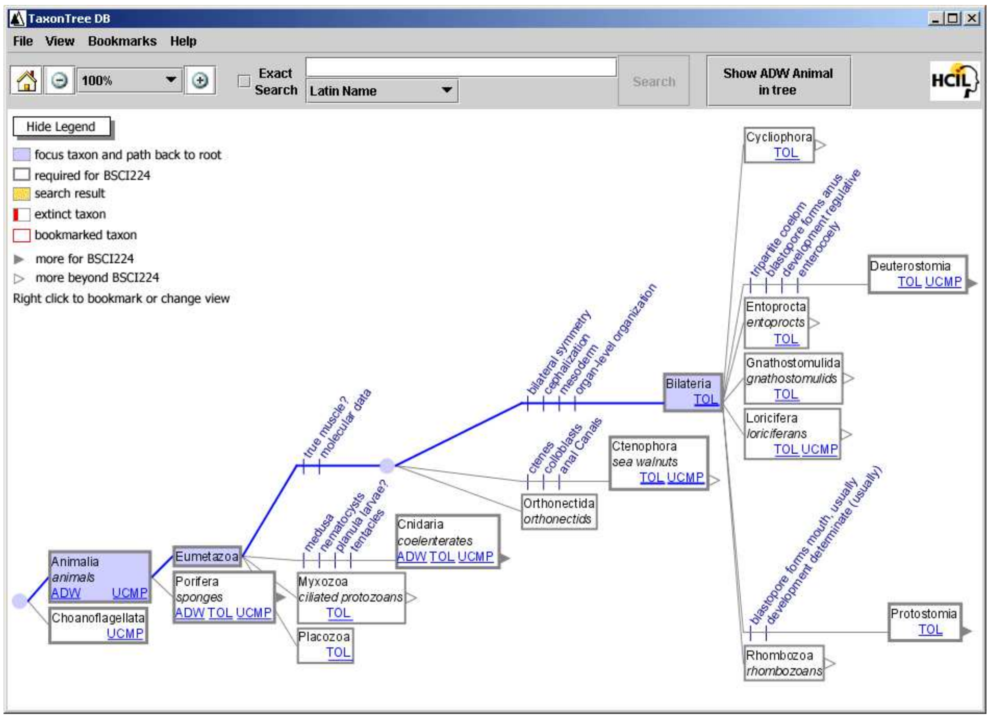

TaxonTree DB
File View Bookmarks Help
100%
Exact Search
Latin Name
Search
Show ADW Animal in tree
HCIL
Hide Legend
focus taxon and path back to root
required for BSCI224
search result
extinct taxon
bookmarked taxon
more for BSCI224
more beyond BSCI224
Right click to bookmark or change view
Animalia animals ADW UCMP
Choanoflagellata UCMP
Eumetazoa
Porifera sponges ADW TOL UCMP
true muscle? molecular data
medusa nematocysts planula larvae? tentacles
Cnidaria coelenterates ADW TOL UCMP
Myxozoa ciliated protozoans TOL
Placozoa TOL
bilateral symmetry cephalization mesoderm organ-level organization
ctenes colloblasts anal Canals
Ctenophora sea walnuts TOL UCMP
Orthonectida orthonectids
Bilateria TOL
Cycliophora TOL
tripartite coelom blastopore forms anus development regulative enterocoely
Deuterostomia TOL UCMP
Entoprocta entoprocts TOL
Gnathostomulida gnathostomulids TOL
Loricifera loriciferans TOL UCMP
blastopore forms mouth, usually development determinate (usually)
Protostomia TOL
Rhombozoa rhombozoans

Figure 3.3: An example screenshot of SpaceTree visualization applied to an animal diversity data sets (incl. an example of how sub-trees are indicated using a triangle/arrow metaphor)

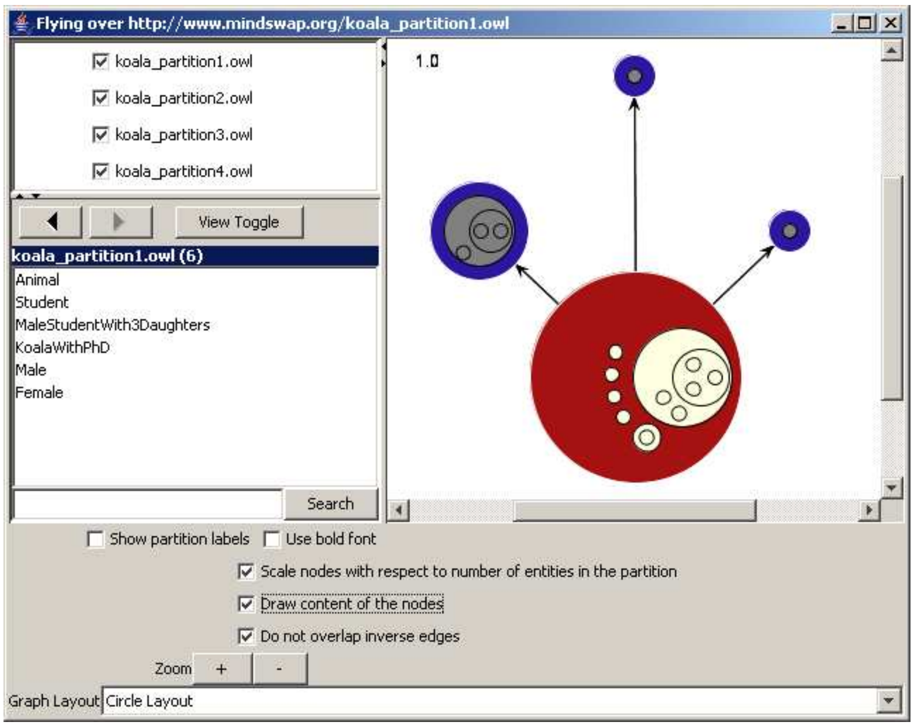
Another algorithm based on the focus-context metaphor is SpaceTree [PGB02]. SpaceTree is a tree browser to some extent similar to hypertrees [LRP95]. It addresses one difficulty of the hyperbolic geometry; namely constant updating of the visual representation, which makes it hard for the user to create a mental map of the ontology, hierarchy or taxonomy. SpaceTree uses dynamic rescaling of tree branches to fit within a constrained space; miniature tree icons are used to indicate the depth, breadth and size of the sub-trees hidden behind a given node. Tree node expansion and contraction are animated, which helps user to cognitively manage the transition between the nodes or drilling deeper into the tree. SpaceTree, however, loses somewhat its edge when it comes to working with multiple hierarchies. An example of a simple SpaceTree visualization of a taxonomic structure is shown in Figure 3.3[5].

3.2.2 Metaphor 2: CropCircles

In a similar tune, another more recent example of actually addressing specific user needs is the "crop circles" metaphor [PWG05]. As with fish-eye, this metaphor also shows some implicit topography in an overview mode. The idea draws on what mathematicians call Venn diagrams, which for the ordinary user means

[5]Picture re-used from `http://www.cs.umd.edu/hcil/taxontree`

overlapping circles. This greatly simplifies the expressions we used above. “Ford is a Car” implies that category (class, set) named Ford is included in the set named Car(s). This still does not make it any more difficult to say that some Fords may actually be Vans – the “Ford” set may simply overlap with two or more supersets (“Cars”, “Vans”, etc.)



Flying over http://www.mindswap.org/koala_partition1.owl

☑ koala_partition1.owl
☑ koala_partition2.owl
☑ koala_partition3.owl
☑ koala_partition4.owl

koala_partition1.owl (6)
Animal
Student
MaleStudentWith3Daughters
KoalaWithPhD
Male
Female

☐ Show partition labels ☐ Use bold font
☑ Scale nodes with respect to number of entities in the partition
☑ Draw content of the nodes
☑ Do not overlap inverse edges

Graph Layout Circle Layout

Figure 3.4: An example screenshot of CropCircles visualization applied to an ontology

In CropCircles classes and partitions are represented as circles. One can hover over a particular node in the visualization to see the class it actually represents. By clicking on a class one can quickly highlight its immediate neighborhood (children, parents). Additionally, zooming in and out is easily supported in this view and, as the recent study from the same authors [WP06] showed, the metaphor in some cases can outperform other visual techniques (esp. in the context of viewing richly interlinked and deep ontologies). An example of what a (fairly simple) ontology looks like in CropCircles visualization is shown in Figure 3.4[6].

From the same family of methods used to contain a large and complex taxonomy (or sometimes ontology) in a screen-restricted space we would like to also mention TreeMap [JS91]. Similarly as CropCircles this method also relies on the containment relationship (*isA*) and represents trees as contained rectangles. Rectangles may have different colors and/or labels to associate some characteristics with them. The method was first used to visualize folder and directory structures, but since then its conception was applied for several other purposes. The resulting images, however, may get a little bit daunting if the hierarchies are unbalanced (e.g. one deep branch and several shallow ones) or the structures are too large (the rectangles get too small to distinguish them easily). On the other hand, this method performs quite well if one is interested in seeing a distribution of a particular attribute over the hierarchical structure; the aforementioned coloring of the nodes can provide a rapid overview without delving into details. An example is shown in Figure 3.5[7].

The SpaceTree method mentioned in section 3.2.1 was found to perform slightly better in terms of cognitive payload associated with exploring and comprehending the topography of single ontologies than TreeMaps and to some extent also CropCircles[WP06]. This seems to be attributable to its more exploratory strategy – one can easily move through the ontology, yet get an overview at the same time. As mentioned earlier, CropCircles performed better when multiple ontologies or multiple modules with cross-links where involved.

[6]Picture re-used from `http://www.mindswap.org/2005/cropcircles/`
[7]Picture re-used from `http://www.cs.umd.edu/hcil/treemap`

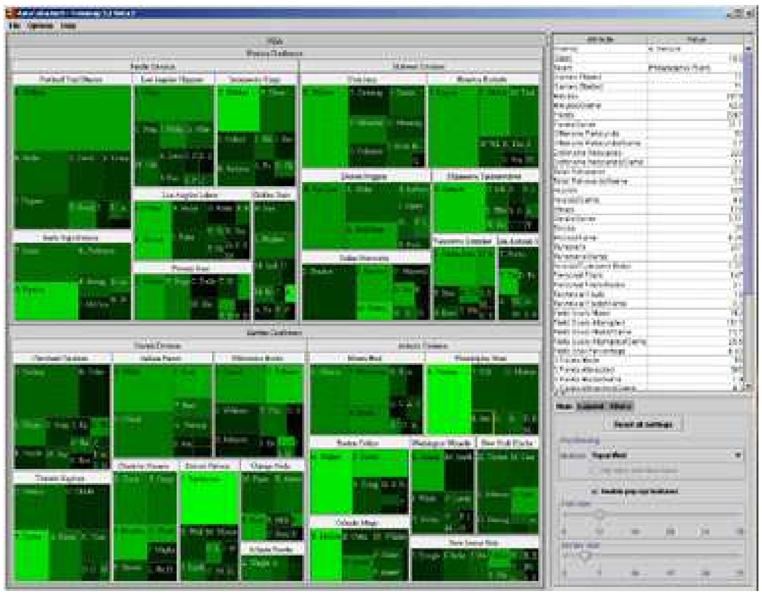


Figure 3.5: An example screenshot of TreeMaps visualization applied to a taxonomy with coloured attribute distribution

The CropCircle algorithm fits if one investigates the topography (especially of multiple ontologies) and has limited interest in the actual content.

3.2.3 Traditional Ontology Visualization Techniques

On a more traditional level, ontologies are often perceived by many developers, researchers and users as predominantly hierarchies of subsumed concepts; i.e. structures where one concept is a kind of another concept (as in "Ford is a Car"). Hence a lot of effort was put into navigating these so-called isA structures. A good example of this strategy is the classic IsaViz technique [8]. This method simply focuses on one dominant relation in a semantic structure; namely, the hierarchical relation denoting subsumption. Although the isA relation is dominant in many existing semantic networks and ontologies, it has two shortcomings: First, its usefulness rapidly falls with the depth of a hierarchy, and second, very few graphs actually have a neat hierarchical structure.

Already the visualization or the navigation of as few as 15 objects forming a semantic network may become hard to interact with, when following purely the *isA* links. The *isA* graphs make visually interesting demonstrations, but by definition they do not contain various lateral or horizontal relations [BR02]. Another problem with the *isA* hierarchy arises when one allows not only the defined structure to be visualized but also the inferred one. This can significantly increase the complexity and the depth of the actual visual representation.

While the isA visualization metaphors have their own, fully justifiable use cases, they are clearly driven by the technology. Simply, they take advantage of the graph semantics of RDF as the underlying Semantic Web technology. Whilst mathematically correct, these kinds of visualization applications do not offer much support for ordinary users interested in their tasks. The trick with most of these abstract, graph-based visualizations is that they are essentially for a single purpose and do not have a semantically clear interpretation. On the other hand, graphs are good at showing overall patterns among the entities, such as classification clusters, graph topography (i.e. depth, breadth, etc.) and breakdown of a more complex structure into a series of disconnected or sparsely connected sub-graphs [MG03].

One of the main issues of the generic graph visualization techniques is their over-reliance on one specific relationship – *isA*. Since this relation expresses the fact that one category subsumes another (e.g. "a sheep

[8]See `http://www.w3.org/2001/11/IsaViz`

isA mammal"), the tree and other hierarchical metaphors are particularly suitable. However, the same trees or graphs are much harder to interpret if one chooses to cluster objects by other relationships. For instance, authors from UK Ordnance Survey reported on their use of a range of ontological relationships that may easily create issues if inappropriately visualized [DHG06].

In particular, they highlight issues with fairly common geo-spatial relationships like *contained within*, *next to* or *surrounded by*. In each of the cases illustrated, merely showing two nodes from the low-level data representation linked with a declared or inferred labelled arc is not of much use. For instance, in some cases objects such as fields may be both *surrounded by* and *contained within* and *be inside* of a wall. However, if field F is *contained within* something else (e.g. wall), by definition it cannot be *next to* another field F', since they would need to share the "container". However, to anybody visualizing a dataset containing fields F and F' it makes perfect sense to "ignore" the dividing walls and talk just about the fields.

Although it makes sense to humans to ignore certain objects and/or relationships, ontology-based visualization and navigation tools are not proficient in this direction. They either opt for well-defined but not always useful relationships (such as *isA*) or they rely on making application-specific assumptions, using domain-specific heuristics and similarly – none of which is easily re-usable. Hence, we face a similar choice as with the ontologies and statistical techniques: we can have either high precision thanks to ontologies but possibly low recall due to their brittleness or we can alternatively achieve a high recall with a low precision.

Nevertheless, one possible way out of this situation would be to rely more on the fact that certain relationships may not be logically sound but perfectly sensible to human users. For instance, by allowing the users to express the fact that *in the context of* relationships such as *next to* can be ignored in the case of a wall being between two fields. Furthermore, these amendments might be easily captured in user profiles (as mentioned in section 2). Hence, the pragmatic interpretation of a certain declared relationship may need to be made only once and thereafter it would be applied to each situation satisfying a particular *context*. However, this requires further research into how such contextual factors might be exploited, captured, etc.

3.2.4 Discussion: Issues and Benefits

Many researchers attempted to analyze the principles and goals people have when visually interacting with (large) datasets. Their cognitive studies, one of the recent examples is e.g. [DF04], show that there are several mutually not fully compatible requirements on interacting through visual user interfaces:

- a need to find a particular item (e.g. when knowing some of its properties)
- a need to explore the context in which an item is defined (e.g. what does it mean if we say that "Ford is a Car")
- a need to establish the difference between two or more items, which may include temporal differences due to evolution or various conceptual differences (e.g. "Ford Transit is a Ford, but not a Car, and this is because. . . ")

The simple IsaViz and related techniques basically address only the second need identified above, and even that to a very small extent. Some of the later developments in the field of visualization for Semantic Web took an approach more centered on the user needs. A good example of this is Jambalaya [ESAM03], which started in a traditional, technology-centered style. However, at the application re-design stage, the needs of real users were considered for particular audiences comprising the biologists in a large national research center. These requirements were gathered by observing the actual biologists and conjecturing potentially useful functional requirements from these observations. As a result, Jambalaya is more balanced in addressing those three types of user needs mentioned above.

Jambalaya's visualization is still based on a graph metaphor, but allows more customization of what can be visually depicted. Particularly its FilmStrip metaphor [ESAM03] shows an interesting compromise between data overviews and its specific context. Yet, due to realizing this idea through showing the relevant information as nodes, the outcome is full of boxes and overlapping edges. These often achieve the opposite of a positive user experience because they may easily obscure much of the underlying semantic structure. Furthermore, as mentioned in the previous section on faceted browsing, Jambalaya and other navigational techniques focused on the containment (*isA*) relationship would in general have difficulties to cope with the multi-dimensional nature of ontologies and Semantic Web space.

More recent examples of visualizing Semantic Web content include metaphors that are moving away from the actual technology that needs to be visualized. An example is the application of a metaphor that considers all objects in a semantic structure to be charged particles connected with springs (i.e. attractive and repulsive forces) [MG03]. While not entirely new, the “springs” metaphor makes the notion of navigating through complex semantic structures very much comparable to cleaning up a desk full of papers, notes and devices. Pushing one pile of papers to the side has an effect on other piles in the vicinity. Some items are connected more tightly than others (e.g. computer and its network cable), and this shows in how far they can move individually. Hence, the metaphor presents some of the implicit topography.

3.3 Looking Forward

3.3.1 Large-Scale Visualization for Ontology Exploration

The implications of the discussion in the above paragraphs are that there is unlikely to be one perfect method or technique for scaling up the navigation through structured datasets. What is more likely to work is reusing familiar metaphors, such as the FishEye projections or CropCircles. However, it seems equally important to use these metaphors at the right point during the process of navigating ontologies. Crop Circles, for instance, seem to fit best if one is interested in seeing broad relationships among several ontologies. Map-like FishEye projections, on the other hand, seem to show a finer level of granularity – e.g. when one wants to explore the relationship of networked ontologies to a particular concept in one ontology.

One approach that has not been mentioned so far, but which actually could combine the need of dealing with large-scale datasets with the need to simplify the ontological definitions, is inspired by maps and mapping metaphor. By definition, any map is essentially a projection of a particular world (most often a landscape) onto a paper (or a screen). Different maps differ in what landmarks or features are projected and how is the projection calculated.

Having said that, one can extend the approach described in section 2.5.2, where the authors mined a large dataset for a subset of repeated terms. They calculated latent distances between different terms and finally, spread the resulting vector field onto a two- or three-dimensional space. Now taking this idea further, one can imagine creating such domain landscapes from several different perspectives. For instance, a landscape of research topics in Europe is likely to look somewhat differently from the landscape of UK’s football or the landscape of great maritime voyages.

Assume we have several pre-computed landscapes available that show the key terms of a particular domain, their links, relationships, closeness, etc. When we take one or several ontologies, we can cover these domains with the given ontologies. In some cases, the coverage would be better and more precise than in others. Different ontologies would be positioned into different regions of the landscape - dependent on which landscape does the user take as a foundation for his or her navigation.

Furthermore, the user would be obviously allowed to swap one virtual landscape for another, thus seeing what the effect of changing an emphasis or focus is. In other words, the same visual metaphor may contribute to several different requirements of different user groups. Perspectives of the two user groups might be approximated by two distinct domain landscapes. Any ontology the user wants to consider – e.g. to annotate or structure data, measurements or general articles – can then be seen as a map that could be more or less true to the original landscape. Obviously, as with any other transition from a landscape to a map, maps would take the granularity of the landscape only to certain limited extent. Nevertheless, by “snapping” ontology to the landscape, one can intuitively and visually simply appreciate the fit of a given ontology (or a network of ontologies) to the domain of interest.

Another useful side-effect of turning our attention to maps and presenting them as a potential remedy to large-scale navigation issues is the capability to zoom in and out. Any landscape may be looked at from a closer distance or from farther. Similarly as a photograph taken from a hot air balloon shows more details than a picture taken by a satellite, our virtual landscapes may also show finer or coarser granularity. In consequence, one would be able to project ontologies onto overview landscapes and investigate them more in depth in landscape cutouts – without necessarily needing a different navigational toolkit.

3.3.2 Navigating Ontologies vs. Datasets

In the previous sections we briefly discussed several approaches to aid users in navigating through (i) conceptual complex and (ii) large-scale data spaces. However, most of the tools and metaphors that were mentioned earlier are in fact techniques for navigating and exploring structured data - personal records, library content, museum collections, etc. While some of the techniques use ontologies as the conceptual schemas informing the definition of facets (for example), these ontologies tend to be fairly simple. Above we highlighted the fact that in order to keep the complexity of navigation to the minimum, tools like CS AKTive Space but also MIT OpenCourseWare opt for around three facets.

In order to navigate the actual ontologies, such restrictions may be inappropriate. A different strategy is therefore needed in order to make sense of several facets possibly applicable to a particular ontology. One interesting strategy to this extent was proposed by Mulholland, Collins et al. as spotlight browsing [CMZ05]. The principle of this navigation strategy is again based on a metaphor – a torch throwing a beam of light. The user selects a resource or a concept from a particular collection; then the collection is dynamically restructured so that it conveys interesting properties, clusters, etc. that may be relevant to the initial "spot". These additional items and concepts are then structured around the original spot by calculating their semantic closeness. The navigation is then equivalent to shedding a light beam, which puts certain concepts into light (i.e. into navigable focus) and certain other items into shadow (i.e. into non-navigable periphery).

Although the application of spotlight browsing is similar to faceted browsers; primarily to browse and explore museum collection, the role of ontological structure is much stronger. As a consequence, the need to navigate through ontological structures is much more visible than in some other museum exploration tools. Indeed, as the authors of the above technique said in the context of one of their most successful implementations:

> Bletchley Park had been a [World War II] code breaking and intelligence center. During this time Alan Turing developed a mechanical decryption machine called the Bombe and Tommy Flowers developed Colossus, the valve-based semi-programmable computer. Bletchley Park became a heritage center in the early 1990's. As is the case with many heritage centers, Bletchley Park has a large number of digital resources that they are unable to display during the physical visit. [We wanted] support a post-physical visit experience, where recent visitors could use the content to follow-up interests developed while at the Park.

In other words, the motivation was not only to allow content browsing but tell something about how different items, stories and people relate to each other, as well as to broader historic themes. Unlike faceted browsers, spotlight technique actually supports the user to take path alongside any type of relationship from one concept to another. Moreover, the technique is likely to co-exist well with some of the algebraic operators – especially those for summarization. The outcome of this symbiosis between spotlight-like navigation and e.g. techniques for summarizing ontology regions can then lead to scaling the process up. At this moment this is only a hypothesis, but spotlight technique is used with between eight and twelve items forming a neighborhood to the central concept. If instead some of these direct concepts we introduce summarizing notions, this might indeed improve the speed and scalability, while still reducing the overall complexity of ontologies.

What seems to come out of this brief review of exploratory techniques can be summed up as follows: Users benefit from a range of different techniques and they use them at different stages of interacting with ontologies (taxonomies) and for different purposes. A majority of methods looks either at the content of ontological instances (the faceted browsers) or at the topology (the large-scale visualization techniques). In our opinion there is a gap in actually visualizing the ontology content in terms of what topics it covers. This shall be further explored in the context of work package 4 and in collaboration with work package 3. Our initial idea is to enhance the notion of concept clouds and spotlight browsing [CMZ05] into perceiving ontologies as maps of particular domain landscapes.

Similarly as a term occurrence diagram can be seen as a map of a (part of) real world, we suggest generating term occurrence landscapes as simplified worlds against which different ontologies could be visualized as “maps” reflecting some specific aspect (or context) of making sense of the world. A *hypothetical* scenario and a mock-up image of such a landscape and its potential role in interacting with networked ontologies is further elaborated in section 6.1.

Chapter 4

Algebraic Operators

One key aspect of NeOn is to network different ontologies with each other according to the specific needs of a task at hand or in order to cover a complex domain. The network of ontologies will itself be a large ontology consisting of several ontology modules. Already reusing and arranging those smaller ontologies in the network can be seen as a kind of customization or personalization of the used ontologies. Another kind of ontology personalization is the generation of a view on an existing ontology or filtering some of its elements. Examples for this second kind of personalization are the summarization of ontologies or removing parts which are not relevant for the domain the user is interested in.

The difference between those two kinds of ontology personalization is that in the former case the personalization deals with several ontologies and how to connect them. In the latter case, the personalization involves only one ontology. Of course, in real world scenarios both kinds of personalization will interact with each other. For example, one may remove irrelevant parts from one ontology before networking it with other ontologies.

The algebraic operators in this chapter are a means for supporting these two cases of ontology personalization. One can distinguish between binary operators which help to integrate an ontology into a network with other ontologies and the unary operators which help to adapt a single ontology. Several examples of these operators will be listed in section 4.1. It identifies the different kinds of operations available in the literature like the *union* or *intersection* of two ontologies. This overview of the theoretical background will be followed by a description of several concrete tools which support the user in reusing and adapting ontologies. However, only a few of the tools described in section 4.2 are based on a sound formalism.

4.1 Theoretical Frameworks for Operators

One of the first works in the area of algebraic operators for ontologies is [Wie94]. It proposes to have a more modularized approach to creating ontologies as this would ease ontology reuse and would help to breakdown the required effort into smaller, manageable pieces. In the following, it describes the general idea of algebraic operators which support this modularized approach and help to combine the modules to larger ontologies. Later work on this topic in [MW04] further elaborates the idea of the operators and formally describes them. Those operators were designed for use in the ONION system which is described below in section 4.2.1. One can distinguish between binary operators and unary operators, depending on the number of the used input ontologies. The following operators are proposed in [MW04]:

- SELECT: There exist four different *select* operators which help to select those parts of an ontology module which are of interest.
- INTERSECTION: The *intersection* of two ontologies contains their common elements. This operator will be explained in more detail below.
- UNION: The *union* of two ontologies contains all their elements.

- DIFFERENCE: The *difference* of two ontologies contains all elements of the first ontology minus those contained in the intersection of both input ontologies.

Another work going into a similar direction is [KWA06] and [KFWA06]. It also defines a *intersection*, *union* and a *difference* operator. Additionally, the following operators are defined:

- DIVISION: This operators computes a quotient ontology of two other ontologies where one ontology is a sub ontology of the other. The elements of the sub ontology are shrunk to a single element in the quotient ontology.
- PIVOTED PRODUCT: This operator is more or less the inverse of the *division*. It takes also two ontologies as its input and additionally one element from the first ontology. In the pivoted product, the given element will be replaced by the second ontology.
- MERGE: There exist two kinds of *merge* operations. They work very similar to the *union* operator. The exact differences will be explained below.

There are important differences between those two approaches to algebraic operators which influence their capabilities and also the outcome of the operations. For example, although both approaches define a union, intersection and difference operator, they will lead to different results. One important difference is the ontology format on which the operators work:

The ontologies in [MW04] are represented as directed, labeled graphs and a set of rules. This format is based on the work in [GPdBvG94]. The nodes in the graph describe the concepts of the ontology while the edges are the relationships between those concepts. The nodes as well as the edges are labeled, with “SubClassOf”, “InstanceOf” and “hasA” as examples for edge labels. There doesn’t exist a predefined set of relationships between concepts. Properties like the transitivity of the subClassOf-relation have to be explicitly defined as rules for the ontology. The rules of an ontology are expressed as Horn Clauses. This is a very general ontology format which allows easy transformations from other ontology formats.

Another approach is taken in [KWA06] and [KFWA06]. Those operators work on RDF ontologies for which the authors defined a formalization in terms of the set theory. The semantics of this formalization is close to RDFS(FA), as it is described in [PH03]. This approach has the advantage that it deals with all the specifics of a certain ontology format (e.g. the reification in RDF) which may be lost by transforming it into a general format. But this may also be seen as a disadvantage as it complicates the adaptation of the operators to other ontology formats.

The second difference between the two approaches is with regard to how mappings between ontologies can be defined. Mapping two ontologies means that they are kept separate and only linkages are built between them. This is in opposite to the merging where the involved ontologies are affiliated into a new ontology [Kle01]. These mappings are essential for binary operators because they are required to identify those elements in two ontologies which are equal or stand in a relation to each other. Otherwise it would be impossible to e.g. compute the union or intersection of two ontologies. In absence of an explicit mapping one may also use name-equality of e.g. concepts. This would correspond to a very simple implicit mapping between the ontologies.

In [MW04], the mapping must always be given explicitly. The authors call their mapping mechanism the *articulation of ontologies*. Such an articulation is expressed as a set of Horn Clause rules, called the *articulation rules*, which introduce edges between nodes in the two ontologies. It is important that this articulation rules may introduce new edges but no new nodes as this would influence the properties of the algebraic operators. An advantage of using rules for defining the mapping is that one can introduce new edges dependent on whether a certain condition is fulfilled (see [MW04] for an example which makes use of this feature).

The paper doesn’t impose a certain way how articulation rules should be generated. The articulation generating function may be an automatic procedure or even a human who generates articulation rules. But in order to ensure certain properties of the algebraic operators like idempotence or commutativity, the articulation generating function has to fulfill some conditions (see [MW04] for more details).

In [KWA06] and [KFWA06] the mapping between two ontologies is represented as a homomorphism. This mechanism allows to map e.g. concepts with different names to each other. With this mapping mechanism one can define a subset of the mappings which are possible if one would use the Horn Clause articulation rules of [MW04]. For example, it isn't possible to map two concepts with each other depending on a condition.

In [MW04], all algebraic operators rely on the existence of an explicit mapping and make use of it in form of articulation rules. This is not the case for the operators defined in [KWA06] and [KFWA06] where only the *merge* operations make use of an explicitly defined homomorphism. The other operations rely on intensional similarity, i.e. ontology elements like the concepts and properties have to be name-wise equal. This is an important restriction of the operators and their most obvious difference to those defined in [MW04]. The other differences have their cause in the used mapping mechanism.

Another work with regard to algebraic operators is [JMN+99]. It describes four binary operators and four unary operators. While the binary operators are very similar in their scope to the previously described operators there are some interesting new unary operators introduced, namely the *summarize* and *glossarize* operator (see below).

The ontology format used in [JMN+99] is an object model for semistructured data and several primitive operations like constructors and connectors. The object model consists of references, values, attributes and atoms. The different parts of this model are related using the primitive operations that are performed on the object model.

In order to ensure semantic consistency, two semantic relationships are introduced: congruity and similarity. The congruity measure determines which part of the source ontology is relevant to the target application. In contrast, the similarity measure identifies equivalent objects in both source ontologies that can be merged without being identical. While the unary operators transform a source ontology according to a congruity measure, the binary operators use a similarity measure to combine two source ontologies.

The following unary operators are described in [JMN+99]:

- SUMMARIZE: This operator centralizes the source ontology into groups of similar concepts.
- GLOSSARIZE: This operator lists all terms that are subordinate to a concept without any of the sub-structure.
- FILTER: This operator reduces the instances from the source ontology according to a selection predicate.
- EXTRACT: This operator reduces the concepts and the corresponding instances from the source ontology according to a selection predicate.

The *filter* and *extract* operators are in their purpose very similar to the *select* operators in [MW04] but the *summarize* and *glossarize* operator are unique compared to all the previously described operators. They can be used for identifying meaningful parts of an ontology thus leading to a kind of personalized or task specific view on ontologies.

The following four binary operators do not offer new functionality compared to the previously described binary operators. The differences will only be in the detail of how the operators work. These are the binary operators defined in [JMN+99]:

- MATCH: This operator returns the information from both source ontologies and a new object that contains a sequence of pairs of references in order to match concepts in both source ontologies.
- BLEND: This operator extends the *match* operator and adds the attributes to each object in a source ontology that are present in the other source ontology.
- INTERSECTION: This operator returns only the common elements of both ontologies.

- DIFFERENCE: This operator returns only the elements which are not common to both ontologies thus computing a symmetric difference, which distinguishes this operator from the difference operator described above.

While the preceding binary operators need two ontologies with an overlapping domain, the assembling operator described in [SS03] expects two orthogonal ontologies that do not overlap:

- ASSEMBLING: Assembles the concepts, relations and axioms of both ontologies following specific rules. The output of the *assembling* operator is very similar to the Cartesian product of both ontologies.

The assembling operator is non-deterministic and has no closed definition. Applying the operator is an iterative, interactive and semi-automatic process. It has been deployed in form of the FONTE tool (see 4.2.4). The assembling operator is, like the operators in [JMN$^+$99], much less formalized than the operators taken from [MW04] and [KWA06]. This formalization allows the latter also to make statements about the properties of the operators (e.g. whether they are commutative). This formalization is an important step for defining an ontology algebra.

4.2 Tools for Integrating Ontologies

In this section several tools will be presented which help to integrate two or more ontologies with each other. From all these tools, ONION is the only one which implements more than one of the previously described operators. All other tools are specialized on a specific task and operator. PROMPT and Chimaera were specifically designed for merging two overlapping ontologies with each other thus coming close to the *union* operator. The main focus of those two tools is not so much on a formalization of the merge process in form of an operator but on the heuristics for identifying similar concepts in the ontologies and guiding the user in the merge process. FONTE also implements only one operator but it keeps a balance between describing and formalizing this operator and guiding the user in order to apply it.

4.2.1 ONION

The ONION (ONtology compositION) system is a tool for semi-automatic integration of ontologies. It helps the user in defining articulation rules (i.e. mappings) between ontologies which should be integrated. An overview of the ONION system is available in [MKW00]. It consists of a set of algebraic operators (see [MW04] and section 4.1) and a tool for semi-automatically generating mappings between the used ontology modules. ONION works on ontologies which contain concepts as well as instances.

The main idea behind ONION is that, instead of building single, monolithic ontologies from scratch one should create and reuse smaller ontology modules and combine them with the help of the algebraic operators according to the specific needs at hand. An important goal of ONION is to reduce the costs for creating and maintaining this ontology. This should be achieved by two factors:

The first factor are the algebraic operators which declaratively describe how ontology modules should be combined. Having a declarative description of the ontology composition task has the advantage that it can be repeated with a reduced effort in case that one of the ontology modules has changed and that this change should be propagated into the combined ontology. This also requires that the ontology modules logically remain unchanged by our needs and we only define how to combine them according to our needs.

The second factor is to only map those parts of the ontology modules to each other which are required for the current needs, even if further mappings may be possible. This reduces the dependencies between the modules and especially during the maintenance phase it helps to reduce the effort which is required to adapt the composed ontology to changes in the modules. The basics of the mapping generator used in ONION are described in [MWJ99] and [MW02].

4.2.2 PROMPT

PROMPT is an algorithm and tool which helps the user in the task of ontology merging and alignment [NM00]. It is a further development of SMART [NM99]. Later on, the tool was integrated into a whole suite of tools for ontology merging and mapping [NM03].

There are two major differences to ONION: The most important difference is that PROMPT merges the source ontologies into a complete new ontology while ONION establishes links between the source ontologies keeping them separate from each other. The second difference is that the operators in PROMPT have not such a theoretical background and not the same level of formalization as the algebraic operators in ONION. Instead, they are described in natural language text. Examples for operations supported by PROMPT are “merge classes”, “merge slots” or “perform a deep copy of a class from one ontology to another”.

The ontology format of PROMPT is frame-based and compatible with OKBC. The ontologies may contain classes, slots, facets and instances. The classes are arranged in a subclass-superclass hierarchy where multiple inheritance is allowed. To each class slots may be attached which are binary relations between a class and either another class or a primitive object. They are inherited by the subclasses. The facets impose further constraints on the slots (e.g. the cardinality or value type of a slot). Instances are individual members of classes. Because of its very general ontology format that is expected PROMPT can be applied over a lot of knowledge-representation systems.

The PROMPT tool can be used for interactively merging two ontologies (consisting of concepts and instances). For this purpose, it creates an initial list with operations which are then suggested to the user. For example, PROMPT may suggest to merge two concepts or instances coming from separate ontologies because of their lexical similarity, e.g. of their names or descriptions. If the user wants to merge two concepts where no such similarity is given then the corresponding operations has to be manually created.

Applying an operation triggers further automatic updates of the ontologies, for example all references to the merged concepts or instances have to be changed accordingly. While applying an operation on the ontologies there may arise conflicts or inconsistencies in the ontology which are detected by PROMPT and shown to the user who decides how to resolve them. But also further suggestions may be generated by PROMPT: For example if there were major changes in a certain part of the taxonomy PROMPT may suggest the user to revise this part.

The main advantage of the tool is that it systematically guides the user through the ontology merging process and that it helps by suggesting and automatically applying specific operations on the ontologies. In a user study it was shown that experts agree to a large extend with the suggestions made by PROMPT and that it helps to significantly speed up the ontology merging process compared to the time needed if it would be done manually with an ontology editor like Protégé-2000 (see [NM00]).

An additional advantage is, that it is possible to re-do the merging process if there exists a new version of one of the source ontologies. For this purpose, PROMPT records all operations performed with the tool and it has the ability to replay them. Only those parts of the merge process have to be adapted by the user which involve ontology parts which are changed in the new version of the source ontology.

4.2.3 Chimaera

Chimaera is a browser-based editing, merging and diagnosis tool [MFRW00]. Because it was developed as a pure OKBC application, one is able to load any OKBC-compliant file (from different ontology editors). Chimaera was developed around the same time as PROMPT. Like PROMPT it supports the user in merging two ontologies with each other by identifying edit points where action by the user is required. Compared to PROMPT the support for automatically performing operations in Chimaera is restricted. As operations it only supports changing names consistently and merging two concepts with each other. Everything else is restricted to guiding the user’s attention to possibly inconsistent parts in the ontology where the user has to do the required action. Additionally, Chimaera offers advanced browsing and searching functionality. Like in

PROMPT the suggestions for merging two concepts are based on the similarity of their names.

Although only a limited support and automation is given, it was shown in a user test that Chimaera can speed up the merging process (compared to doing it manually in a usual ontology editor) by a factor of 3.4 in average over the whole merging task. The task was performed around 14 times faster for those tasks where further support was given by the automatic operators (see [MFRW00]). This shows that having such a tool is extremely valuable even if it only gives basic support for the most tedious and repetitive tasks.

Besides the support for merging two ontologies, Chimaera also contains functionality which helps the user to detect inconsistencies in the ontology. These inconsistencies are usually introduced during the merge process. For example, those inconsistency checks detect incomplete information in the ontology like missing argument names or missing documentation. Other inconsistency checks detect problems in the concept hierarchy (like redundant superconcepts) or problems in the semantic of the ontology. Examples for the latter are cycles in the concept definition or domain/range mismatch for the instances contained in the ontology.

4.2.4 FONTE

FONTE (Factorizing ONTology Engineering complexity) introduces an operator which can be compared to the Cartesian product of two ontologies. The operator is introduced in [SS03] where it is used for introducing the notion of time to domain ontologies which were previously “time-less”. Compared to all the previous operators and methods there are two important differences: First, while the previous methods required a certain domain overlap of the ontologies there is no such requirement for the FONTE operator. Second, the FONTE operator is not a general operator which can be used for combining any two ontologies with each other. Instead, it has to be adapted to the theory of time which should be introduced to a domain ontology.

This adaptation of the operator is done by writing rules how the theory of time can be introduced into a domain ontology. These rules are put into the configuration files of the tool which applies FONTE. An example of such a rule would be that concepts in the domain ontology can be derived from either of the two concepts *Instant* or *Period* in order to introduce the notion of time to them. This kind of rule has to be created for all concepts in the time ontology and this work has to be repeated if another theory of time should be used. Further fields where FONTE might be useful are introducing notions of space, trust or user access rights into a domain ontology. But FONTE is not feasible for scenarios like those in [MW04], [NM00] or [MFRW00] where two, possibly larger, domain ontologies should be combined.

The operator of FONTE is less formalized than the algebraic operators of ONION. Like PROMPT and Chimaera also FONTE has a list where the tool suggests next tasks which have to be accepted by the user (see Fig. 4.1). Initially, this list is empty. Only after the user started to make some changes to the domain ontology by manually creating and executing new tasks, FONTE tries to conclude further changes. The domain ontologies may contain concepts, relations and axioms. FONTE was evaluated with two domain ontologies. In both cases the user initiated around 10 tasks. Subsequently, in the first case FONTE proposed 135 tasks from which 78 were accepted by the user and in the second case 85 tasks were suggested and 67 accepted.

4.3 Conclusion

As mentioned in [NM03], there is a gap between the theoretical and the tool-oriented research. Both mark the extremes on a scale. The theoretical research tries to identify and formalize general operators which can be used in several scenarios of ontology engineering and maintenance. On the other side is the tool-oriented research which starts with a specific task, like merging two ontologies, and tries to guide the user through this process. These solutions are only applicable to one specific task. All those practical tools work in a semi-automatic manner where the user has to confirm the actions which were proposed by the tool. Often, these decisions of the users are related to confirming a mapping between the two input ontologies.

The advantage of the tool-oriented research is that they guide the user step by step through the task so that even non-experts will be able to do it. This is a problem of the theoretical research in this area. Even if there

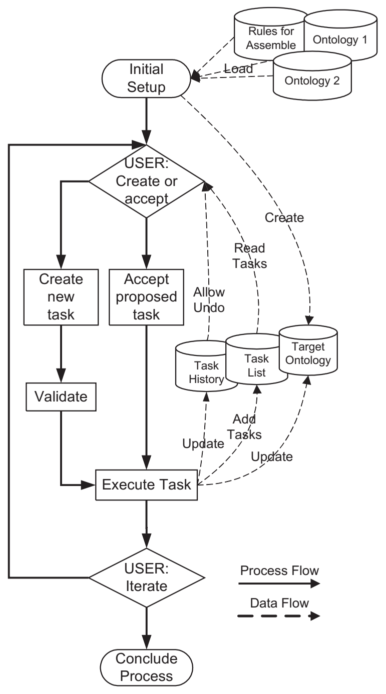


Figure 4.1: The assembly process of FONTE (taken from [SS03])

exists an implementation of the operators (as it is the case for the operators in [MW04] but not for those in [KWA06]) it is not further specified what their best usage is in order to achieve a specific task like merging two ontologies. Thus, they require a deep knowledge of the topic.

An important advantage of having generalized operators is that they can be used in different contexts and for different tasks. But in order to get the full potential one has to define a kind of best practices of how to use those operators for achieving a specific task. These best practices may then lead to a tool which guides the user through the process and hides the details of the operators to non-experts.

Furthermore, the majority of the current tools do not implement unary operators for filtering or summarizing ontologies. They are mainly used for combining two or more ontologies. Depending on the used tool, the combination has other requirements and is differently performed. For example, two ontologies with overlapping domains can be combined with ONION, PROMPT and Chimaera while FONTE is designed for combining two ontologies with orthogonal domains.

Chapter 5

Annotation Tools

Annotation is the process of adding extra information to an existing document in order to allow a machine to automatically interpret it. During this process concepts and relations among these concepts are identified and formally described. Additionally, metadata is produced that describes this information formally in a variety of semantic web languages like RDF or OWL.

Currently, documents have no semantics at all and are only interpretable by persons. Clear examples of these documents are invoices that contain certain fields such as the emitter and the receiver of the invoices, the amount or the price. These fields are only interpretable by persons because a machine cannot understand what price or emitter means. Another example are documents which may describe fish stocks and how depleted they are. By adding semantic annotations to these examples a machine will be able to interpret and properly manage the information, thus increasing the automation of managing the information.

In this deliverable we will only cover the annotation of documents but another requirement coming from the case studies is the annotation of databases. These database may for example contain spatial or time-series data. Annotating those data and making them accessible via an ontology will be covered in more detail in WP2 which, amongst others, deals with upgrading database content. Usually, the annotation of databases will be done by providing a mapping between the database schema and a corresponding ontology.

5.1 Initial Requirements

Several requirements can be identified regarding semantic annotation [UCI^{+}06]. In this document we highlight three of them which we consider most important and relevant for NeOn. Because semantic annotations can become obsolete after a certain amount of time, the first requirement is that annotations have to be maintained in order to ensure the consistency of the information. Information that was valid yesterday may be wrong today. It is therefore important to provide mechanisms for automatically updating the existing annotations.

Additionally, the consistency of annotated data should be checked. A clear example of this requirement is a transaction between a client and a wholesaler. This transaction may be repeated with another client so that some of the data will be the same (for example the legal fields in an invoice) but some other data may differ depending on the type of client that the wholesaler is dealing with. The system should provide an environment that helps the user in maintaining and reusing these annotations.

NeOn deals with networked ontologies and because of that semantic annotations must be stored in multiple and distributed ontologies. Domain ontologies which store the annotations will be composed of several distributed ontologies. In the previous example the legal part of a transaction may for example be stored in a government repository while the data representing the business process may be stored in a repository of the wholesaler [GDM^{+}06]. The support of managing distributed semantic annotations is a key issue.

5.2 Annotation Frameworks

In order to build systems that fulfill the previous requirements there exist annotation frameworks for the Semantic Web. These frameworks are Annotea [KK01] and CREAM [HSS03].

Annotea is a W3C project under the Semantic Web Advanced Development project. It provides collaboration mechanisms for using the generated annotations. Annotations can be any kind of comment, note or explanation attached to a document. Once the user loads a document s/he will find previously inserted annotations. Annotea uses RDF for describing the annotations and XPointer for locating the annotations in the annotated document. Annotea is part of the Semantic Web efforts. It provides an extensible RDF metadata framework for rich communication about web pages while offering a simple annotation and bookmark user interface. The annotation metadata can be stored locally or in one or more annotation servers and is presented to the user by a client which understands this metadata and which interacts with an annotation server via the HTTP service protocol.

The CREAM (Creating RElational Annotation-based Metadata) framework specifies the components that an annotation environment should have. It specifies the most important components of an annotation tool including an annotation inference server, an editor and an information extraction component. For representing the annotations CREAM uses RDF or OWL while XPointer is used for locating annotations in a document.

An example of a tool that is based on the CREAM framework is Ont-o-Mat [HS02] which provides a user interface for annotating web pages by selecting the information which should be annotated and subsequently associating it with a domain ontology. In the following section further annotations will be described.

5.3 Types of Semantic Annotation

There are two different types of annotation tools: manual annotation tools and automatic annotation tools. With manual tools the user has to manually add the annotations to every document. Automatic annotation tools allow the users to annotate a document automatically, without having to actively participate in the annotation process. Both approaches are extreme and there don't exist fully automatic annotation tools and manual annotation tools always offer the user some automation. Usually, automatic annotation tools apply machine learning methods in order to improve the automation but they still require a minimal intervention by the user.

5.3.1 Manual Annotation

As it was previously said, the manual annotation involves the participation of the user who associates the annotation metadata to the documents. The manual annotation tools support this process e.g. by providing access to the ontologies which should be used for the annotation. Examples of manual annotation tools are Amaya, Annozilla, S-CREAM, SHOE Knowledge Annotator, SMORE, Magpie or Open Ontology Forge. In the following, two of them will be explained in more detail:

- Magpie

  Magpie [DDM04] is an annotation tool developed by the KMI of the Open University. Magpie enables the user to browse a web resource, to select the desired content of this resource and to annotate it with the concepts of a domain ontology. The users will select the data which they want to annotate and Magpie will associate these data to the desired entity from ontology. Magpie can be installed as an extension of a web browser. Once it is installed the user can load a lexicon and Magpie will identify the concepts that exist in the currently loaded web page.

- SMORE

SMORE[1] is an annotation tool provided by Mindswap. This tool allows the users to markup web documents with OWL ontologies. The tool provides the user functionalities to browse a web resource and generate from it triples following the subject-predicate-object model. If the users already have an ontology they can select it and store these annotations in the ontology. Additionally, SMORE allows users to compose and send e-mails with context-based semantic markup.

5.3.2 Automatic Annotation

Automatic annotation is the process by which a computer program (agent, browser, etc.) can automatically extract semantic annotations from any given data source. It is known that a data source must be structured in some way in order to achieve acceptable levels of quality and recognition in automatic semantic annotation. If the data source is more structured, it is easier to set-up automatic annotation tools and obtain acceptable results from them. In the following list of automatic annotation tools we consider tools that requires participation of the user in the annotation process but which include automatic annotation components.

- KnowledgeParser

  KnowledgeParser [CBB$^+$04] is an automatic annotation tool developed by iSOCO. This tool provides means to annotate documents (primarily web pages) and to store these annotations in a previously selected domain ontology. The annotation process consists of defining visual relations between the layout elements of a web page and using these relations to identify the elements which should be annotated. These relations are defined in a “wrapping ontology” that describes the layout of the web page. For example, it is possible to define that an element is on the right of another different element in order to identify it. Because the annotation process is based on visual relations (for instance, the field “name” is on the right of the field “surname”) KnowledgeParser can also annotate other types of documents like text files or .DOC documents.

  KnowledgeParser also uses regular expressions to identify the elements of a document which should be annotated. It is possible to define a regular expression in order to indicate that the piece of data we want is an e-mail or a date. By using these relations and regular expressions it is possible for the user to define the elements contained in the document and to store them in an ontology, giving semantics to the previously unstructured data.

- Text-Garden

  Text-Garden[2] is a set of text mining tools provided by the Text Mining and Semantic Web group of the Josef Stefan Institute. The text mining tools enable the user to easyly handle text documents for the purpose of data analysis. These tools include automatic model generation and document classification, document clustering, document visualization, dealing with web documents, crawling the web and many other sources.

  Further tools can be used for preprocessing documents (like HTML to XML converter, HTML to text converter, etc.) document clustering (Bag-Of-Words K-Means clustering or Bag-Of-Words Hierarchical-K-Means clustering), classification of documents (like binary-class or one-class support vector machine training algorithms, Winnow training algorithm, etc.) and tools for constructing/extracting documents (like learning a semantic document space in which documents are represented based on Latent Semantic Indexing).

- GATE

  GATE[3] is an architecture for natural language processing (NLP). It is possible to plug in several kinds of NLP tools into this architecture. These tools can be part of speech (POS) tagging tools, sentence

[1]http://www.mindswap.org/2005/SMORE/
[2]http://www.textmining.net
[3]http://gate.ac.uk/

splitters or named entities recognizers. GATE works with two different types of resources: language resources and processing resources. Language resources refer to any kind of data-only resources such as lexicons, corpora, thesauri or ontologies while processing resources refer to components like lemmatizers, parsers or speech recognizers.

GATE can also be used for annotating documents. GATE allows the users to annotate web resources by loading them into the application. The resources that can be annotated include XML files, text files, web documents, .DOC files or .EML files. We refer to the GATE manual for further information. The users need to specify the location of the resource and load it into GATE. Once the document is loaded it is possible to select manual annotations for parts of a resource or to load a corpus pipeline in order to automatically annotate resources. GATE offers the users methods to automatically annotate the document, for instance, with a thesaurus or to automatically identify specific parts of the documents (for instance, in a XML file it is possible to automatically identify the fields of the file). The annotations can be stored in a data store (Postgres DataStore, Oracle DataStore or Serial DataStore) or exported to XML.

- Solvent

  Another automatic annotation tool is Solvent[4]. This tool has been developed inside the SIMILE project[5] at MIT. Solvent offers the user an annotation environment that is integrated into the web browser as a plug-in. Once Solvent is activated it allows the user to visually select parts of the web page s/he wants to annotate. If the selected part of the web page is repeated, for example a set of cells in a table, it is sufficient to select one of those parts and the tool will select all. It is possible to define which specific fields we want to annotate (using regular expressions) and where to annotate them. Solvent will produce a scraper code in Java once we have selected what we want to annotate. After that we will be able to start the annotation process. It is also possible to directly store the annotations in Piggy Bank [HMK05] in RDF format.

- KIM [6]

  KIM (Knowledge and Information Management) provides an infrastructure for semantic annotation of unstructured and semi-structured documents. The architecture of the system is based on GATE and contains an upper level ontology (KIMO, KIM Ontology) which has concepts like “Object”, “Happening” or “Location” (with typical attributes and relations like “SubRegionOf”). KIM annotates web documents by using an Internet Explorer extension. When the user starts the KIM plug-in a new tab will appear on the left with the KIM ontology and the annotation process will start after clicking on the annotation button.

  KIM is able to recognize the different entities in the web page, like persons, places, currencies, etc. if these entities are represented in the ontology. These recognized entities are then associated with the ontology. Once the annotation is done, KIM offers the user to connect to the KIM server where it retrieves all available information regarding the current user and/or the the relations that are bound to recognized entity.

  KIM has a knowledge base in RDF(S). The knowledge base and the ontologies can be accessed through Sesame. Additionally, KIM offers a web user interface for semantic queries and a knowledge explorer for navigation through the knowledge base. Information retrieval is based on Lucene[7] for indexing documents.

- Armadillo

  Armadillo [CCDW04] is a system for document annotation and creation of knowledge bases from large repositories (e.g. the web). The main idea is to have a system based around Semantic Web Services

[4]`http://simile.mit.edu/wiki/Solvent`
[5]`http://simile.mit.edu/`
[6]`http://www.ontotext.com/kim/semanticannotation.html`
[7]`http://jakarta.apache.org/lucene/`

(SWS) where the inputs and outputs are semantically typed and therefore they can refer to anything having some sort of meaning (i.e. a concrete object, an abstract concept, etc.).

The initial configuration of Armadillo requires an ontology, an initial lexicon and the repository of documents we want to annotate. The lexicon is associated to the ontology and the system will look at the set of documents identifying the information contained in the lexicon. Confirmation is required once the information is identified. Armadillo afterwards learns from the extracted information and continues annotating the source files. Further verification of the newly extracted information is required. Once the process is done the annotations are stored in a RDF repository.

Armadillo is primarily intended for automatic information extraction from web sources. The tool starts with an initial lexicon and an ontology and is able to learn by annotating web pages. The adaptive information extraction used by Armadillo is very effective in reducing the amount of human intervention during the annotation process.

5.3.3 Problems During Semantic Annotation

During the semantic annotation of documents there exist two typical problems, namely (i) keeping the annotated information up to date and (ii) keeping the information coming from different documents consistent. The first problem is caused by annotations which become obsolete over time and which should be refreshed. Usually, new annotations will add new instances but in case of refreshing information one is interested in replacing the old annotations. This may be solved by defining a kind of primary key where new annotations overwrites the old annotations if they are equal with regard to the primary key.

The second problem of keeping information consistent is very similar to the scenario described in 6.2. In case of the annotations the inconsistencies may occur because annotations come from different sources (e.g. web pages and Word documents) which may contradict each other. For resolving those inconsistencies one may apply an approach similar to that described in 6.2 where the user prefers one of the sources over the others.

5.4 Presentation of the Annotation Results

The annotation process produces a large amount of data. For example, if we plan to annotate a set of the CIA World Factbook[8] the produced annotations will be at least about all countries that are represented in that web page. If we plan to annotate the documents regarding a fishery stock of a specific region the knowledge base storing these annotations will contain a large amount of information that needs to be presented in a comprehensible way to the user and exploration techniques are a good way to do it.

Faceted browsing will be a nice option to show the annotated data and to sort or select instances created during the annotation process. Thus faceted browsing will ease the navigation process of the user. For example, if the user wants to annotate the information contained in the CIA World Factbook about several countries s/he will be able to inspect these information by browsing along the economic relations and social relations.

A query to the large set of instances produced in the annotation process will return a large set of results. Algebraic operators may be used to reduce the complexity of the obtained results. For example, operators like *summarize* or *glossarize* may help the user in obtaining more useful results than just viewing the whole set of produced annotations.

Annotations can also be used to automatically infer mappings between different ontologies. Annotating instances with different ontologies may help to find some similarities between these ontologies (for more details see section 6.4).

[8] `https://www.cia.gov/cia/publications/factbook/index.html`

5.5 Annotation Scenarios

Annotation can clearly be applied in the use cases. For example, in the FAO use case an employee from a region, where the fish stocks are under control due to the risk of overexploitation, might send out a document with the latest statistics. These statistics have been gathered during the last days and have been recorded in an unstructured document. In order to be able to infer knowledge from the data it is necessary to annotate the data.

If the document is just a web page it is possible to use tools like SMORE or GATE where we will select the relevant parts of the document and store them into the domain ontology. Once the data is stored, the NeOn toolkit will be able to infer knowledge from the gathered resources. If the document that was sent by the employee has a .DOC format we will select the KnowledgeParser annotation tool to define the layout of this document and to subsequently annotate it.

In the FAO use case there is a special interest on spatial data, i.e. data which is related to specific geographic coordinates. Examples for spatial data are information about the amount of fish remaining in a determined location of the world. This information can be found at the FIGIS site or other web resources. Tools like KnowledgeParser are able to navigate through these sites and annotate the desired data with the concepts of an ontology that fulfills the requirements for representing this kind of data. The same process can also be applied to time-series statistics, which are also of high importance for the FAO use case.

Another possible scenario can be found in the invoicing use case. Invoices can be sent in many ways, from standards like EDIFACT[9] to an ad-hoc format defined for the particular management tool of a stakeholder. These documents currently have to be annotated in order to map them with invoice models. Tools like Text-Garden and KnowledgeParser may be used in order to extract the knowledge contained in these documents. Once the invoices are annotated it is possible to map the most important information (amount, sender, receiver, products, etc) with our current model and to automatically manage the complete invoicing process.

5.6 Concluding Remarks

As we have seen there are many tools for annotating the content of documents (e.g web pages or Word documents). The most advanced annotation tools are the tools that help the users in annotating web pages but we have also shown tools which can be used for annotating any kind of document. In some tools the user only highlights those parts which should be annotated while other tools require a description of what should be annotated. But all these tools require different degrees of human participation during the annotation process.

Manual tools like SMORE, Solvent or Magpie expect from the user that s/he selects the desired in a web resource. This desired data may be just some concrete fields inside the web page like a string in the field “name” or a set of tables that follow a structured layout. Automatic tools like Knowledge Parser, GATE, KIM or Text-Garden allow the user to extract information from documents by using different techniques. These techniques range from defining the layout of the document to visually indicating what we want to annotate. Other tools like Armadillo apply machine learning techniques in order to minimize the required amount of user interaction.

[9] `http://www.unece.org/trade/untdid/welcome.htm`

Chapter 6

Use and Test Cases of Customization and Personalization

There are two main areas where various ontology customization and personalization techniques can be applied. First area broadly covers the customization during viewing and exploring a network of ontologies. This customization is more or less ad-hoc and the intermediate stages of the user interaction may be often discarded once the user proceeds with exploring the ontology. The opposite holds for the second area where customization is applicable – during designing and modifying ontologies. Here the intermediate results of the customization will often be integrated into the edited ontology or otherwise formally represented.

Often, a user may seamlessly switch between designing and viewing an ontology. For example, it may be necessary to explore the edited ontology to find the correct location, where a new concept or a particular mapping should be added. In many situations another ontology has to explored before integrating it into the edited ontology and customizing it for the author's own purposes.

In the following, we start by presenting a broader test case focusing on how different techniques and approaches mentioned in the review apply to specific user requirements. The purpose of this test case is to map the technological solutions onto selected user needs and requirements, which were identified in deliverables D7.1.1 and D8.1.1. In order to make the scenario in section 6.1 easier to comprehend and follow, we use several specific names for the tools. However, these names are merely examples; the actual outcomes of NeOn to support the over-fishing alert case study may be different.

The broad test case is followed by several smaller scenarios focusing on particular operations on the networked ontologies. Hereby we consider potential benefits of specific techniques and elaborate on their roles. These scenarios will demonstrate possible uses of the customization and personalization techniques reviewed in this report and their relevance for the case studies of NeOn.

6.1 Ontology Exploration and Integration in the FAO Library

In this scenario we imagine a librarian from FAO of the UN who wants to annotate a new scientific report coming from a member state. The report draws on the existing knowledge, which already has been formally captured in various departments of FAO. Nevertheless, the report also introduces new topics, issues, and perhaps terminology. One of the librarian's tasks is to select appropriate ontologies that would sufficiently cover the submitted report. His or her challenge is to identify the most useful subset of all possibly relevant ontologies with no knowledge of formal methods and/or measures for "minimal coverage".

This task comprises several sub-tasks that need to be supported by the librarian's ontology engineering toolkit. The first sub-task, for instance, sees the librarian noting a set of terms or tags around what s/he perceives as a central theme of the report. Let's assume the central theme is the "*migration patterns of Northern Atlantic Albacore*". The problem our librarian may face is to formulate and run many separate queries on the NeOn infrastructure in order to gather relevant existing schemas, data sources, ontologies, etc. Even if the librarian was willing to query the infrastructure many times, this is likely to lead to the problem with processing large number of potentially relevant ontologies and semantic data.

In order to provide help, the librarian chooses the customizable Explorer – a part of the NeOn toolkit, which s/he initializes with the FishBase ontology. In the FishBase ontology s/he quickly locates concept "*Albacore*" and invokes an interactive dialog (Tag Cloud Constructor, cf. 2.5.2), which uses the NeOn infrastructure to propose a summary of neighboring concepts, terms or tags. As s/he wants to go beyond internal FAO sources, s/he allows the Tag Cloud Constructor to match "Albacore" against non-ontological resources, such as archives of open source Wikipedia and libraries of other UN organizations and regional fisheries bodies.

The outcome of this initial query explosion would be a large set of terms and tags – of different quality, provenance and ontological roles. The Tag Cloud Constructor takes this vast unstructured space of terms and applies on it the "Module identification using reduction" component, which is one of NeOn's techniques in the customization algebra (see section 4). The reduction aims to find out overlaps and possibly generalizations of term subsets, so that the diversity could be expressed using a smaller number of concepts.

However, after carrying out the reduction, the librarian realizes that s/he can actually provide a more precise criterion for the compounding technique. In particular, s/he needs to add a spatial reference to capture the basic concept of a stock. For that purpose, s/he manually introduces a constraining tag – "*regional*". The algebraic techniques in question (e.g. reduction, compounding, etc.) can now be used for identifying modules in the ontologies which cover this user-defined tag (see section 4). The result of this step was useful but there were many ontologies covering this term, such as economic zones, land areas, water bodies, statistical and reporting areas, etc.

At this point, rather than proposing such diverse exploration possibilities, the NeOn Explorer makes use of a simple piece of meta-data associated with the librarian's current job. The report arrived from ICCAT, a regional fisheries body responsible for monitoring Atlantic Tuna. Hence, the reductionist algebraic technique can now be re-invoked with the ICCAT's type – that is "reporting area" would be the most appropriate interpretation of regionality with respect to the underlying report. In other words, the librarian benefited from his/her association with a particular FAO division that is responsible for interacting with reporting bodies. In this simple way s/he basically instructed the module creation techniques to prefer a neighborhood typically used in his/her user group.

The processed and filtered tag cloud would then proceed to a Spotlight Browser (see section 3.3.2), which would position the different terms around the original central theme of "Albacore". The initial position of the "light beam" would reflect that exploratory path through the cloud that seems to be best covered by the existing FAO ontologies – see mock-up in Figure 6.1. The numbers in superscript in the figure refer to the internal formal resources in FAO referring to a particular theme (e.g. FIGIS, AgroVoc, etc.). In addition, the weight of the terms is given by their ontological reliability and provenance – where the librarian quickly sees that the fish species are particularly well conceptualized.

In the spotlight shape, the librarian can quickly appreciate the position of his/her core theme among other potentially related terms, concepts, schemas and ontologies. His or her attention is drawn to the items in red, which (a) are relevant to the notion of including regions s/he manually entered earlier; (b) are to some

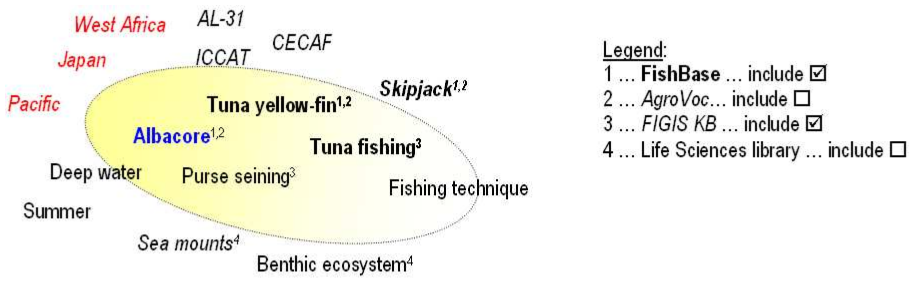


Figure 6.1: Mock-up of the initial summary of related terms and ontologies covering these terms. Typefaces reflect the trustworthiness of terms against ontologies with same typeface on the right.

extent contradicting the “Mid Atlantic” that s/he expected to be solely associated with ICCAT and the original query; and (c) do not fit together (are disjoint) according to NeOn ontology selection algorithm. Therefore, s/he decides to reposition the focus of the navigational beam to explore appropriate parts of the tag cloud. In the meantime the Tag Cloud Constructor was issuing ontology discovery queries to show examples of how different parts of the tag cloud could be covered. The outcome of this background investigation shows up as the librarian repositions the spotlight (see Figure 6.2).

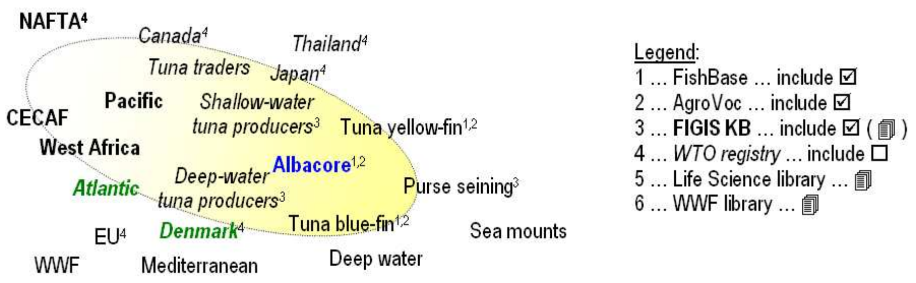

NAFTA4
Canada4
Thailand4
Tuna traders
Japan4
Pacific
Shallow-water tuna producers3
CECAF
Tuna yellow-fin1,2
West Africa
Albacore1,2
Deep-water tuna producers3
Atlantic
Purse seining3
EU4
Denmark4
Tuna blue-fin1,2
Sea mounts
WWF
Mediterranean
Deep water
Legend:
1 ... FishBase ... include ☑
2 ... AgroVoc ... include ☑
3 ... FIGIS KB ... include ☑ (🗐)
4 ... WTO registry ... include ☐
5 ... Life Science library ... 🗐
6 ... WWF library ... 🗐

Figure 6.2: Mock-up of the repositioned focus of related terms and ontologies covering these terms.

The first thing that the librarian sees is the reorganization of the conceptual space – tuna producing regions emerged in the picture together with differentiation of some fishing strategies. The librarian sees that new data sources appeared alongside the new information. See for instance, *WTO registry* and *WWF library* on the right-hand side of the figure. These new sources reflect the view shift and the librarian has a choice to include them alongside other (mostly internal) sources. At this point, our librarian is not sure whether WWF and Life Science would suit better the purpose. Also, the NeOn Explorer displays meta-information from the NeOn’s ontology repository indicating overlap of the two new sources (see the icon with overlapping sheets next to several repositories) with the already used FIGIS ontology. Since s/he is not an expert in evaluating ontologies in general, s/he launches NeOn’s OntoMapper tool to briefly appreciate how the discovered ontologies cover the terminological landscape of the FAO over-fishing alert system.

OntoMapper displays a pre-processed automatically mined map of the domain data in the over-fishing alert system and allows the user to zoom in or out to find the appropriate level of granularity. Once the landscape is set, the librarian activates the ontology layering mode, which attempts to pinpoint concepts (or concept neighborhoods) from each investigated ontology onto the alert landscape. While both new sources cover the “migration patterns”, the Life Science schema is not sufficiently covering the key implicit issues – over-fishing, alerts and other specific ecological observations (see Figure 6.3). However, bearing in mind that the resulting set of ontologies may be used by a variety of stakeholders, the librarian wants to double-check how these partially overlapping ontologies cover other landscapes (e.g. agriculture and regional economy). OntoMapper allows this simply by swapping one underlying landscape for another and stretching the ontological networks on the new area.

After selecting the ontologies that sufficiently cover the neighborhood of the new report the librarian proceeds with defining recommendations on typical navigational patterns. These correspond to different combinations of views or facets (see section 3.1), and reflect different relationships – within one ontology or between several ontologies. The purpose of this step is to enable the data manager to pre-compute those facets that are most likely to be visited by the visitors to the FAO Alert Portal. It is not necessary to define all such patterns, as there already are various patterns discovered for various user groups – thanks to NeOn Profiler (see section 2). The Profiler’s capability to mine large collections of seemingly ad-hoc data leads to several proposals of profile segments. For example, s/he can preview profiling the end user groups according to domain type and also role, like policymaker, fisheries manager, student, assessment expert.

User issues and case study requirements addressed in the test case include:

- reducing ontology complexity;
- making sense of links and relations within/between ontologies;

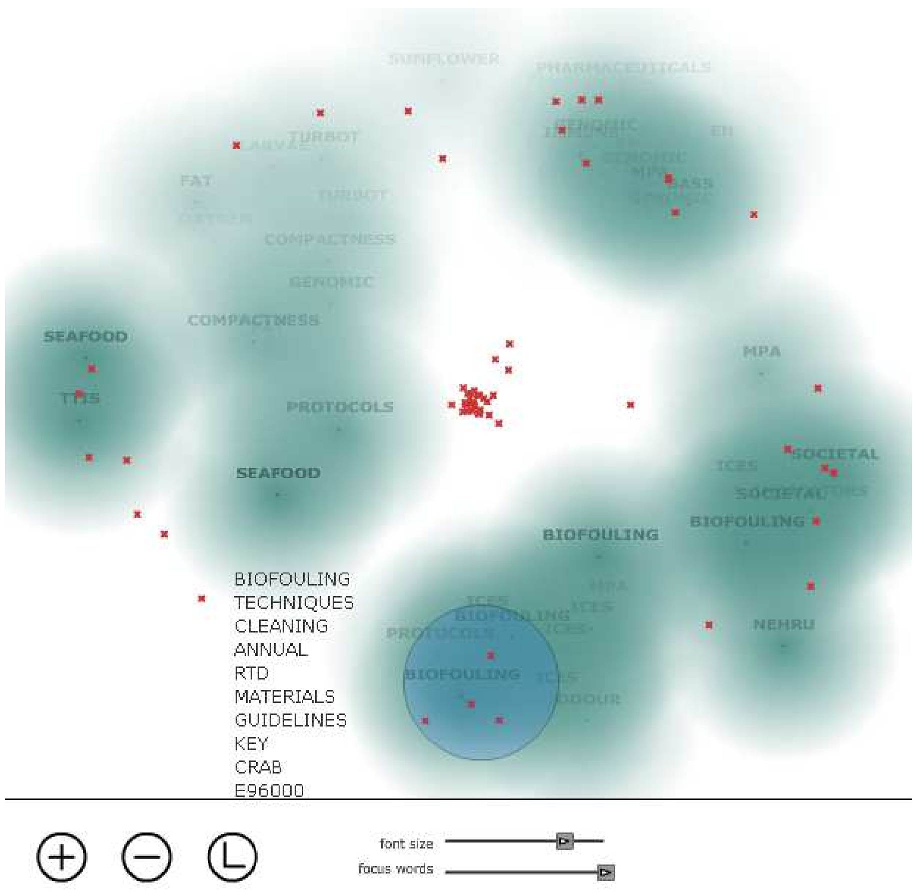


Figure 6.3: Mock-up of a 2D rendered landscape with two ontologies broadly covering and mapping different sections of it. Green areas roughly correspond to different ontologies and red crosses to selected terms whose distance/mutual positions depend on a particular corpus.

- modularization and view customization based on different user-related criteria;
- hiding the low-level aspects of several ontology engineering tasks

Tools or techniques reviewed in this report used in the test case:

- OntoGen and its future extensions (section 2.5)
- Ontology visualization on the level of domain coverage (section 2.5.2)
- Spotlight Browsing and its future extensions (section 3.3.2)
- Algebraic operations such as module reduction, compounding, differencing (section 4)
- to some extent also faceted views (section 3.1)

6.2 Customized Resolution of Redundant and Inconsistent Information in Integrated Knowledge Bases

As it has been mentioned in the use cases of [HHR+06], there is a possibility that inconsistent information may occur if several, consistent knowledge bases are integrated. A typical source of inconsistencies are possible redundancies in knowledge bases which should be merged (see 6.2.2). Techniques for exploring the integrated knowledge bases should be able to deal with such inconsistencies and redundancies. One way of resolving them is to take contextual information of a particular user into account – e.g. whether s/he prefers information coming from a certain (more trusted or more preferred) knowledge base. Then, redundant and contradicting information coming from other knowledge bases is removed step by step until those problems are resolved.

Exploration techniques like faceted browsing described in section 3.1 can be extended so that they allow presenting local views on potentially inconsistent knowledge bases. If the functionality for resolving inconsistencies based on a user's context is not available in a certain environment, then those exploratory techniques should point the user to the inconsistencies and allow an easy, manual resolution of the problem. While the actual resolution of the inconsistencies is out of the scope of this deliverable and work package, there are several ways how techniques reviewed in this deliverable can contribute in this scenario.

Having knowledge about the context of the user captured in a user profile is central to this customization strategy. The information about profiles can be gathered manually (e.g. by asking the user to fill in a questionnaire), or automatically (e.g. by collecting information about the user's previous browsing behavior) or by a combination of the two in a semi-automatic way (see 2). The profile can be expressed in different ways: For instance, it can consist of a set of rules assigning a score (level of the users preference) to each available knowledge base using the values of some parameters describing the user. The user profile can be used to support the user when combining different knowledge bases. The profile itself would be influenced by several parameters including the trust in certain information in the knowledge bases (trust issues will be handled in a separate deliverable), the users affinity, experience, expertise, trace record, problem severity, etc.

6.2.1 FAO Case Study

In addressing the FAO case studies, several (in the range of 5-20) knowledge bases would need to be integrated. Some of them are more relevant for specific type of users (e.g. fishery managers); here we expect semi-automatically assigning the initial user profile to the user by relating him or her to one of the pre-defined groups/roles.

For instance, in FAO a typical group of users are fishery managers, fishery engineers, biologists, ecologists, etc. The questionnaire needs to be designed to capture main characteristics of the knowledge bases related to the specific case study. Relevant databases in the FAO case study include the following:

- *FIGIS* includes fact sheets on species, vessel types, gear types, fishing techniques, fishing areas and time-series statistics (see `http://www.fao.org/fi/figis`). This database may for example be interesting for biologists (species), engineers (vessel and gear types) or assessment experts (statistics).
- *FISHBASE* is a comprehensive site on fish species available at `http://www.fishbase.org/`. It is primarily of interest to biologists, probably looking within certain species groups that are their specialty.
- *ASFA* contains fisheries abstracts, covering practically everything of interest to fisheries (see `http://www.fao.org/fi/asfa/asfa.asp`).

Further examples of databases are *EIMS* (`http://www.fao.org/figis/servlet/static?dom=root&xml=refer/pubsearch.xml`), *NEMS* (`http://www.fao.org/figis/servlet/static?dom=root&xml=refer/nemssearch.xml`) and *FAOLEX* (`http://faolex.fao.org/faolex/index.htm`).

It would be necessary to capture the parameters of the user's position/preferences related to the characteristics of those knowledge bases. For instance, a fishery manager would prefer some knowledge bases due to their relevance to the domain and maybe prefer other because of the credibility of their providers (e.g., data provided by regional bodies may be preferred over the data provided by international ones) or because of suitability of their measurement methods. A specific user like a fishery manager may have on top of that certain preferences for some parts of a knowledge base due to her/his own experience in the domain (e.g. having specific knowledge about a certain geographic region or species). A user profile should capture those preferences and enable their mapping onto the characteristics of the knowledge bases.

6.2.2 Semantic Nomenclature Case Study

In the Semantic Nomenclature case study, ontology editors have different types of ontologies that could be integrated in order to complete and merge information about drugs. As it is described in section 4.1 in [GDM$^+$06], different pharmaceutical models or nomenclatures (BOTPlus, Vademecum and Digitalis among others) are maintained by different stakeholders in Spain.

These models represent the same domain and contain complementary and sometimes redundant information about drugs. Especially redundancies are problematic as they may lead to inconsistencies when merging those models. The selection of the relevant information, the merge and cleaning of redundancies, and the automation of the mappings between the different models should be supported and facilitated by the NeOn platform (see also 6.4). This information may e.g. help workers at the General Spanish Council of Pharmacists (GSCoP) to facilitate the update of the BOTPlus database and to improve its content, which is one objective of the Semantic Nomenclature case study.

On the other hand, the Semantic Nomenclature aims to collect information from heterogeneous data sources and ontologies. So it is of great importance to this case study that the system helps to select ontologies and check their relevance and coherence with the previously selected ontologies.

6.3 Personalized View on a Network of Ontologies

User profiling for the purpose of Human Ontology Interaction can be seen as adjusting the user's view to the information contained in a profile. It can be understood as showing the data and the ontology through the semantic lenses of a particular user (see 2). One possibility is offering a personalized view on ontology by reducing the amount of information presented to the user. Namely, ontologies often contain much more detail than (i) is required for a specific task of the user, and (ii) the user is willing to parse. Having customized views to an ontology is an important feature for reducing their complexity to a manageable size, e.g. by abstracting or collapsing irrelevant elements using a selection of algebraic operators based on the preferred content or role of the user.

There are basically two ways how to create personalized views: On the one hand, a view may only show content which is relevant for the domain of interest of an individual user (in the abstract sense, as defined in section 2). On the other hand, one may restrict the view to content which is relevant for the current task currently performed by a user. Typically, the user profile may contain information about the domains of interest of a user while the task or role based filtering will usually be predefined and be influenced by e.g. certain navigational patterns that are "imposed by" (or associated with) a specific role of a user.

It is also possible to combine these two kinds of views – here again, an algebraic operator supporting a guided merger of modules/views may be applicable (see 4). For example, the content-based filtering might be used to reduce an ontology for a domain like "fishing techniques" and the role-based filtering would further adapt the "fishing techniques" to suit the (typical) needs of a report annotator. The intersection of both views can then be used by an annotator specializing in reports about the fishing techniques etc. The creation of views on an ontology will often be ad-hoc and customized to a particular problem, task or situation. This implies, that the view creation has to be sufficiently easy and lightweight in order to be feasible for non-power users. It shall be reasonably simple as not to distract users from their primary goals in the given situation.

One possible direction for providing a personalized view on ontologies is to assume that the user is constructing several ontologies and the user profile is based on the past ontologies constructed by the same user, user group or by users with the same role (see 2). There are several research questions to be addressed, including construction and usage of the profile and its potential changes over time. In order to make the profile operational and usable for adjusting the user's view on the data while constructing an ontology, we would assume that the ontologies used for constructing the profile are populated. We then may investigate possible ways of incrementally changing the user profile over time, as more data about the user is collected thanks to new ontologies the user has constructed.

We may also consider constructing the user profile in a semi-automatic manner that would enable the user to explicitly select a sub-set of ontologies to be used for constructing the profile. This strategy may be particularly useful if the user is active in several roles while constructing different ontologies and wants to have several independent user profiles – each connected to some but not all of the ontologies that s/he constructed in the past. Advantage of the semi-automatic approach to user profiling can also come from handling new users with no ontologies constructed to date; in such a case we may use active learning to build an initial user profile for the novices.

For filtering ontologies based on preferred content, input from WP3 can be taken here, as there will be a method developed in that work package for constructing user profiles based on the previously constructed ontologies. This can be further enhanced to include user interaction techniques and interfaces enabling the user to adjust their views on the ontology under construction. In this sense, the past user's activities can be seen as providing context for the user in his or her current ontology engineering task. Such an approach can be implemented in collaboration with WP3 inside the existing tool for semi-automatic ontology construction OntoGen (mentioned earlier in section 2).

For filtering ontologies based on the role of a user, input from WP2 about navigational patterns is relevant. Navigational patterns can be seen as a sub-set of ontology design patterns. These navigational patters are implied by the roles a user may take with respect to a particular ontology. Whilst the actual formal representations and compositions of design and navigational patterns are subject of research in WP2, it is important to translate the patterns to e.g. a sequence of typical facets that are most suitable to achieve a particular task (e.g. the identification of regions where a particular fish species is under threat). The facets may be annotated themselves – e.g. in terms of what tasks or roles they relate to, or in terms of how trusted (or "effective") they are for a particular type of users (see 3).

In consequence, such patterns may act as seeds to combining different facets in an exploratory user interface. Similarly, the notion of patterns may be extended to include the presentation of ontologies against certain "domain landscapes" (see the hypothetical case described in section 6.1 and in Figure 6.3). In particular, different user types (and hence profiles) may be associated with certain preferred landscapes that would be used to visualize or depict the ontologies recommended by the system or discovered by the user (cf. the Document Atlas system described in 2.5.2). For example, we can (in principle) imagine a landscape of fishing techniques being almost orthogonal to the landscape of fish species. So perhaps for a marine biologist it makes more sense to use the latter landscape as a basis for visualization; whereas for a fisheries engineer it seems to be more relevant to follow the former pattern in selecting the default landscape.

Obviously, having tied the default "landscape" to a particular role or group does not mean this is the only view available. What other views are at the disposal, may depend (i) on the user's access rights in the first instance, but also (ii) on the *reframing* strategies preferred or endorsed by a particular group. This is particularly relevant since with an increasing number of contextual landscapes and (in principle) allowing any transition between them, the task of exploring and re-interpreting the ontologies may rapidly become cognitively daunting. Thus, user profiles and algebraic operators may help to reduce the space of user choices – in a kind of meta-filtering fashion[8].

6.3.1 FAO Case Study

In the FAO case study we could imagine several groups of users, each having specific interests and expertises with some relevant underlying domains. These expertises and interests also influence which parts of an ontology are relevant for the user group and thus which of them should be shown. Examples of user groups at FAO include:

- *Fisheries Managers* have as domains of interest the exclusive economic zones, major fishing areas or political land areas (where fishes are landed). Thus a typical facet along which a fisheries manager may navigate is the spatial information contained in a knowledge base. But besides those spatial

[8]The *meta-filtering* is meant here as not filtering the actual content of the ontology but rather filtering the user interaction techniques or views applicable to a particular ontology and user type.

aspects s/he may also be interested in information about the commercial classification of species or the legislation and regulations of fisheries.

- *Fisheries Engineers* have as domain of interest the different vessel and gear types in use as well as a species taxonomy. Furthermore, they are interested in fishing equipment. Opposed to the fisheries managers, they will usually not navigate the information along a spatial facet.

The above mentioned two user groups demonstrate how this information may influence the preferred view on an ontology. Further relevant user groups in the FAO case study are the fisheries economist, the stock/resource assessment expert and the marine biologist each coming with other preferences.

6.3.2 Invoicing Case Study

In the Invoicing case study, we may also apply personalization to different sets of users. There are two main types, namely the developers and the users of invoice ontologies. The developers will have to deal with different kind of data. This data will be business processes, legal requirements or technical requirements such as the specific technical issues regarding an application that a given company uses for managing invoices. The profile of such a user may be pre-populated according to the role. Once the initial profile based on the specific role is defined, the NeOn toolkit may start learning from the previously developed ontologies. This may lead to filtering and offering a subset of related resources. Also, based on the NeOn Editor for developing ontologies, the system may learn which features are used most frequently, and possibly show them in the first instance (as a “default” option).

Secondly, there exist those users which actually use the invoice models created by the developers. This user group can be further divided into sub-groups like the emitters and the receivers of an invoice. But even the receivers of an invoice may have different needs: For example, some receivers need to validate that all goods have arrived, that the goods provided match with the delivery note previously generated, etc. This user group should therefore be able to access the invoices that have arrived to the company and also access unstructured data that may contain the information about the products that were originally ordered.

Some other receivers are furthermore enabled to decide whether to accept or refuse invoices. These decision makers cross-checks the invoices with the data maintained by the organization’s main administration system, to which they have direct access. This can be used for justifying the acceptance (or rejection) decision with more complete information. Both kinds of users would benefit from NeOn techniques which help to identify the most likely information and which help to process this additional information.

6.3.3 Semantic Nomenclature Case Study

In the Semantic Nomenclature case study there are three main types of users: editors of ontologies, users with full access to the information and users with restricted access (non-associated). The editors will have to deal with multiple sets of data, both from internal and external sources. An example for an internal source is the nomenclature contained in the BOTPlus database managed by the General Spanish Council of Pharmacists (GSCoP) while the pharmaceutical ontologies of the United States are an example for external sources. The users with full access to all the information are typically the pharmacists associated to the GSCoP. The rest of the users will have restricted access, and the system must hide the private set of data stored in the BOTPlus database and showing the public part.

On the other hand, depending on the past behavior of the user the system should react in different ways. For instance, if the user is reluctant in accessing some of the sources, such as information in ontologies in other languages than Spanish, or ontologies and data coming from non-official sources then the system should learn from this behavior and exclude in principle these types of information and related applications from their first choices.

6.4 Ontology Integration

This scenario deals with the customization of ontologies during the design-time. In this scenario, customization means integration of already existing ontologies into a network of ontologies and adapting them according to the needs of a user. Usually, this task is manually performed like in the user study described in [DMG+06] and [DMA+06]. This study and other similar experiments (e.g. [MFRW00]) showed that the integration task is rather vague and is not well supported by standard ontology editors like Protegé [NSD+01]. It would be ideal, if this important task can be supported by some kind of “ontology re-use wizard”, which might be based e.g. on group- or role-profiles or on navigational patterns (see 6.3 and 2). This wizard would guide the user through the process and offer her/him a pre-selected set of techniques that proved to be useful in a given role or for a specific task.

Several steps have to be performed in order to achieve this task. First of all, the user has to select those ontologies that should be reused. Techniques for selecting ontologies out of a larger repository are described in WP2. But also techniques reviewed in this deliverable may contribute to making sense of the retrieved or selected ontologies. For example, the user may be guided in formulating the actual selection query by taking information from the user profile into account (see 2). Then the selection procedure may compare the available ontologies in how far they cover the user’s current focus of interest.

Alternatively, the selection query formulation may be transformed into a constrained exploration of an initial user-defined term cloud arranged into particular “spotlights” (see mock-ups in section 6.1 and the systems described in 2.5.2 and 3.3.2). In this case, the query is not restricted to logical formulas. Instead, the relationships among concepts might be expressed more in spatial terms. This strategy is frequently used by various tagging systems (e.g. `http://www.citeulike.org/`), where the contributors do not use a formal structure but merely choose keywords they deem sufficient to describe a given resource. Techniques reviewed in section 3 may serve this purpose rather well – they take familiar metaphors of terms, links, clicks, etc. and cloak cognitively more complicated activities (such as a formulation of a more complicated ontology selection query).

The outcome of the ontology selection will usually be several eligible ontologies and the user has to finally decide which of them s/he wants to reuse. For making this decision, it is important that the user is supported in getting a quick overview of their contents. Here the exploration techniques described in section 3, personalized views on the ontologies or a summarization of their content will be useful (see 4). This strand of work has been elaborated in our broader mock-up test case in section 6.1, where we suggested alternative ways of previewing and manipulating concepts and/or ontologies.

Once the ontologies are selected, they must be integrated into a network with other ontologies. This includes reducing the ontology to its most relevant parts and aligning these parts with other ontologies in the network (e.g. ones the user committed to earlier). The step of reducing an ontology to its relevant parts may be done by deploying specialized algebraic operators in parallel to the personalized views on an ontology (see 4 and 6.3).

The mapping and aligning of ontologies is covered in WP3, but techniques described in this review are also relevant for these activities. For example, one can also infer mappings between ontologies if they were used for annotating the same set of instances (cf. [DMDH04]), using an extensional notion of concept similarity. Usually, a set of instances will only be annotated with the concepts taken from one ontology. Here, the annotation tools reviewed in section 5 may be used for annotating this set of instances with the concepts from a second ontology, for which a mapping to the first ontology should be inferred.

The actual integration of an ontology into a network with other ontologies is the primary scenario of the binary algebraic operators described in section 4. Those binary algebraic operators may be used to combine two ontologies with each other; in particular, operators like module union or module intersection. But also the unary operators may be useful for selecting relevant parts of an ontology or module.

6.4.1 Invoicing Case Study

In the Invoice case study there exist users who are the creators of invoice models and who are interested in ontologies themselves. Invoice models can be created by adapting existing models to the particular needs of a given company or a new participant. Thus, this kind of users is interested in detecting similarities with the existing resources. Similarities can be detected, for example, with the help of a concept cloud where related resources would be grouped together. The notion of clouds was also mentioned in section 6.1.

Moreover, it may be important for ontology developers or validators that the eventual "NeOn Suggest" technique automatically offers them those ontologies that are more closely related to their corresponding user profile. These users are interested in ontologies or ontology modules describing any of the three main layers of an ontology in the Invoicing case study. Each of these layers corresponds to one of the domains relevant for handling and describing invoices. These three main layers are the legal, the business and the application layer (see Figure 6.4).

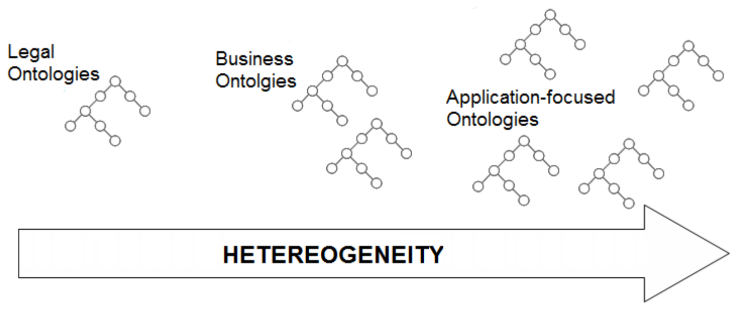


Figure 6.4: Parts of an invoice model and their level of heterogeneity between the models of different partners

The fact that the Legal layer must be compliant with a specific EU directive on electronic invoicing reduces the search space significantly. Technically this might be achieved by applying an operator, but in any case, it needs to be communicated to the user. For example, such a communication may be realized by constraining the choice of facets or "graying out" some terms in the concept cloud. However, lots of possibilities regarding the Application layer can exist.

The e-invoicing application, e.g. SAP or Oracle, determines the language adopted for the invoices. Also, even if the stakeholders use the same invoice interchange language, each of them may structure their invoice information in many different ways. On the other hand, the Business layer may reflect sector-specific, format and language-free information about invoicing practices, like types of discounts applied, purchase dates, emitters, receivers, etc. In this case, it might be sensible to express organization-specific modifications and/or application specifics on top of this more generic layer.

Thus, typically, a creator of invoice models would expect to use NeOn components to assist with searching for existing ontologies related to his/her profile. The profile might be described in terms of Legal and Business layers that the user preliminarily characterized by means of a questionnaire. Of course, this profile would be personal and would extend the group- or role-based profile. Finally, the profile can be enriched by monitoring the user actions during his/her interaction with navigation through the space of suitable ontologies. The navigation through such a space might be expressed e.g. in terms of facet patterns or prevailing sequences of click actions in a faceted browser.

This example from the Invoicing case study demonstrates that integrating and customizing existing ontologies may help during the initial creation phase of an ontology. But it may also help to reduce the effort needed for making the ontologies of different partners interoperable, e.g. of a pharmacy and a wholesaler. In the example above, the different parts of the invoice models will have different levels of heterogeneity (see Figure 6.4). For example, it may be easier for the partners to agree on a common legal layer of the shared ontology, and this should be integrated in their customized invoice model. On the contrary, this is unlikely to be the case for other parts like the business processes as they are significantly different between e.g. a pharmacy, pharmaceutical lab, regulatory body or a wholesaler.

This kind of partial agreement would lead to a reduced effort in creating the mappings between the models of the partners. The effort may be further reduced if only those parts of the models are mapped where interoperability is needed. For example, in most cases it would be sufficient to map only parts of the business processes on each other.

Chapter 7

Conclusion

This deliverable described several techniques for personalization and customization which can be used during navigating and exploring existing ontologies and datasets as well as during the ontology engineering process. For all of those reviewed techniques there also exist conclusions on how the state-of-the-art can be further improved. An integral part of this deliverable are the scenarios described in section 6 which align the possibilities of the reviewed techniques with actual needs of the case study partners.

It is the goal of WP4 to achieve improvements over the current state-of-the-art amongst others in the support of these scenarios and use cases. In most of the identified scenarios more than one of the techniques mentioned in this deliverable are useful. In fact, most of the possible benefits achievable through ontology personalization will be realized by combining and integrating the different techniques (e.g. integration of user profiling and ontology exploration).

Nevertheless, there will be two main directions of research: On the one hand we will work on finding appropriate user interfaces and navigation paradigms for exploring ontologies. Amongst others we will work on techniques like faceted or spotlight navigation. On the other hand we will research how to support the user during ontology engineering in identifying and selecting relevant parts of existing ontologies which may then be re-used in another context and in a network with other ontologies. An important success criterion for this work is in how far we were able to improve the user experience with ontology visualization and engineering tools and whether we were able to cover the scenarios taken from the case studies.

Bibliography

[BR02] P. Brusilovsky and R. Rizzo. Map-based horizontal navigation in educational hypertext. *Journal of Digital Information*, 3(1):156, 2002.

[CBB+04] J. Contreras, R. Benjamins, M. Blázquez, S. Losada, R. Salla, J. Sevilla, D. Navarro, J. Casillas, A. Mompó, D. Patón, Ó. Corcho, P. Tena, and I. Martos. A semantic portal for the international affairs sector. In *Proceedings of EKAW*, pages 203–215, 2004.

[CCDW04] F. Ciravegna, S. Chapman, A. Dingli, and Y. Wilks. Learning to harvest information for the semantic web. In *Proceedings of ESWS*, pages 312–326, 2004.

[CMZ05] T. Collins, P. Mulholland, and Z. Zdrahal. Semantic browsing of digital collections. In *Proceedings of the 4th International Semantic Web Conference*, 2005.

[DDM04] J. Domingue, M. Dzbor, and E. Motta. Magpie: supporting browsing and navigation on the semantic web. In *IUI '04: Proceedings of the 9th international conference on Intelligent user interface*, pages 191–197. ACM Press, 2004.

[DF04] P. Demian and R. Fruchter. CoMem: Evaluating interaction metaphors for knowledge reuse from a corporate memory. Technical Report 158, Center for Integrated Facility Engineering, Stanford University, Stanford, CA, 2004.

[DHG06] C. Dolbear, G. Hart, and J. Goodwin. What owl has done for geography and why we don't need it to map read. In *Online Proceedings of the Workshop on OWL: Experiences and Directions 2006*, 2006.

[DMA+06] M. Dzbor, E. Motta, C. Buil Aranda, J.M. Gomez, O. Goerlitz, and H. Lewen. Developing ontologies in OWL: An observational study. In *Online Proceedings of the Workshop on OWL: Experiences and Directions 2006*, 2006.

[DMDH04] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Ontology matching: A machine learning approach. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 385–404. Springer, 2004.

[DMG+06] M. Dzbor, E. Motta, J. M. Gomez, C. Buil Aranda, K. Dellschaft, O. Görlitz, and H. Lewen. Analysis of user needs, behaviours & requirements wrt. user interfaces for ontology engineering. Deliverable D4.1.1, NeOn Project, 2006.

[ESAM03] N.A. Ernst, M.A. Storey, P. Allen, and M.A. Musen. Addressing cognitive issues in knowledge engineering with jambalaya. In *Proceedings of the Knowledge Capture Conference (K-Cap)*, 2003.

[FMG05] B. Fortuna, D. Mladenic, and M. Grobelnik. Visualization of text document corpus. *Informatica journal*, 29(4):497–502, 2005.

[FMG06] B. Fortuna, D. Mladenic, and M. Grobelnik. Semi-automatic data-driven ontology construction system. In *Proceedings of the 9th Multiconference on Information Society*, pages 223–226, 2006.

[Gan05] A. Gangemi. Ontology design patterns for semantic web content. In *Proceedings of the 4th International Semantic Web Conference*, 2005.

[GDM+06] J. M. Gomez, C. Daviaud, B. Morera, R. Benjamins, T. Pariente Lobo, and G. Herrero Cárcel. Analysis of pharma domain and requirements on the case study. Deliverable D8.1.1, NeOn Project, 2006.

[GMG05] M. Grcar, D. Mladenic, and M. Grobelnik. User profiling for interest-focused browsing history. In *Proceedings of the 8th Multiconference on Information Society*, pages 223–226, 2005.

[GPdBvG94] M. Gysenns, J. Paredaens, J. Van den Bussche, and D. van Gucht. A graph-oriented object database model. *IEEE Transactions on Knowledge and Data Engineering (T-KDE)*, 6(4):572–586, 1994.

[Hea00] M. Hearst. Next generation web search: Setting our sites. *IEEE Data Engineering Bulletin*, Special issue on Next Generation Web Search, Luis Gravano (Ed.), 2000.

[HHR+06] P. Haase, P. Hitzler, S. Rudolph, G. Qi, M. Grobelnik, I. Mozetic, and D. Bojadziev. Context languages - state of the art. Deliverable D3.1.1, NeOn Project, 2006.

[HMK05] D. Huynh, S. Mazzocchi, and D. Karger. Piggy bank: Experience the semantic web inside your web browser. In *Proceedings of the 4th International Semantic Web Conference*, pages 413–430, 2005.

[HS02] S. Handschuh and S. Staab. Authoring and annotation of web pages in CREAM. In *WWW '02: Proceedings of the 11th international World Wide Web conference*, pages 462–473, New York, NY, USA, 2002. ACM Press.

[HSS03] S. Handschuh, S. Staab, and R. Studer. Leveraging metadata creation for the semantic web with CREAM. In *Proceedings of KI*, pages 19–33, 2003.

[HvOH06] M. Hildebrand, J. van Ossenbruggen, and L. Hardman. /facet: A browser for heterogeneous semantic web repositories. In *Proceedings of the 5th International Semantic Web Conference*, 2006.

[JMN+99] J. Jannink, P. Mitra, E. Neuhold, S. Pichai, R. Studer, and G. Wiederhold. An algebra for semantic interoperation of semistructured data. In *IEEE Knowledge and Data Engineering Exchange Workshop (KDEX'99), Chicago*, 1999.

[JS91] B. Johnson and B. Shneiderman. Treemaps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd International Conference on Visualization*, 1991.

[KFWA06] S. Kaushik, C. Farkas, D. Wijesekera, and P. Ammann. An algebra for composing ontologies. Technical Report ISE-TR-06-07, George Mason University, Fairfax, VA, 2006.

[KK01] J. Kahan and M.-R. Koivunen. Annotea: an open RDF infrastructure for shared web annotations. In *WWW '01: Proceedings of the 10th international World Wide Web conference*, pages 623–632. ACM Press, 2001.

[Kle01] M. Klein. Combining and relating ontologies: an analysis of problems and solutions. In *Proceedings of Workshop on Ontologies and Information Sharing,IJCAI*, 2001.

[Kru97] B. Krulwich. Lifestyle finder. *AI magazine*, 18(2):37–46, 1997.

[KS03] J. Komzak and P. Slavik. Scaleable GIS data transmission and visualisation. In *Proceedings of the International Conference on Information Visualization (IV)*, 2003.

[KWA06] S. Kaushik, D. Wijesekera, and P. Ammann. An algebra for composing ontologies. In *Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS 2006)*, 2006.

[Lan95] K. Lang. News weeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning ICML95*, 1995.

[LRP95] J. Lamping, R. Rao, and P. Pirolli. A focus-context technique based on hyperboli geometry for visualizing large hierarchies. In *Proceedings of the Conference on Human factors in computing systems*, 1995.

[Man05] C. Mancini. *Cinematic Hypertext: Investigating a new paradigm*, volume 122 of *Frontiers in AI and Applications*. IOS Press, The Netherlands, 2005.

[MC$^+$94] T. Mitchell, , R. Caruana, D. Freitag, J. McDermott, and D. Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7):81–91, 1994.

[MFRW00] D. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning*, 2000.

[MG03] P. Mutton and J. Golbeck. Visualization of semantic metadata and ontologies. In *Proceedings of the Conference on Information Visualization*, pages 300–305, 2003.

[Mik05] P. Mika. Flink: Semantic web technology for the extraction and analysis of social networks. *Journal of Web Semantics*, 3(2):20, 2005.

[MKW00] P. Mitra, M. Kersten, and G. Wiederhold. Graph-oriented model for articulation of ontology interdependencies. In *Proceedings of the 7th Extending Database Technology*, 2000.

[Mla96] D. Mladenić. Personal WebWatcher: Implementation and design. Technical Report IJS-DP-7472, Jožef Stefan Institute, 1996.

[MS97] C.C. Marshall and F.M. Shipman. Spatial hypertext and the practice of information triage. In *Proc. of the 8th ACM Conf. on Hypertext*, 1997.

[MW02] P. Mitra and G. Wiederhold. Resolving terminological heterogenity in ontologies. In *Proceedings of Workshop on Ontologies and Semantic Interoperability at the 15th European Conference on Artificial Intelligence (ECAI)*, 2002.

[MW04] P. Mitra and G. Wiederhold. An ontology composition algebra. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, pages 93–113. Springer, 2004.

[MWJ99] P. Mitra, G. Wiederhold, and J. Jannink. Semi-automatic integration of knowledge sources. In *Proceedings of Fusion*, 1999.

[NM99] N. Noy and M. Musen. SMART: Automated support for ontology merging and alignment. In *Proceedings of the 12th Banff Workshop on Knowledge Acquisition, Modeling and Management*, 1999.

[NM00] N. Noy and M. Musen. PROMPT: Algorithm and tool for automated ontology merging and alignment. In *Proceedings of the 17th National Conference on Artificial Intelligence*, 2000.

[NM03] N. Noy and M. Musen. The PROMPT suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.

[NSD+01] N. Noy, M. Sintek, S. Decker, M. Crubezy, R. Fergerson, and M. Musen. Creating semantic web contents with protegé 2000. *IEEE Intelligent Systems*, 16(2):60–71, 2001.

[ODD06] E. Oren, R. Delbru, and S. Decker. Extending faceted navigation for RDF data. In *Proceedings of the 5th International Semantic Web Conference*, 2006.

[PBKL06] E. Pietriga, Ch. Bizer, D. Karger, and R. Lee. Fresnel: A browser-independent presentation vocabulary for RDF. In *Proceedings of the 5th International Semantic Web Conference*, 2006.

[PGB02] C. Plaisant, J. Grosjean, and B.B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *Proceedings of the International Symposium on Information Visualization (IV)*, 2002.

[PH03] J. Pan and I. Horrocks. RDFS(FA) and RDF MT: Two semantics for RDFS. In *Proceedings of the 2003 International Semantic Web Conference (ISWC 2003)*, 2003.

[PWG05] B. Parsia, T. Wang, and J. Golbeck. Visualizing web ontologies with CropCircles. In *Proceedings of the ISWC 2005 Workshop on End User Semantic Web Interaction*, 2005.

[SCRH04] M.C. Schraefel, L. Carr, D. De Roure, and W. Hall. You've got hypertext. *Journal of Digital Information*, 5(1):253, 2004.

[SGG+04] N.R. Shadbolt, N. Gibbins, H. Glaser, S. Harris, and M.C. Schraefel. CS AKTive Space: or how we learned to stop worrying and love the semantic web. *IEEE Intelligent Systems*, 19(3):41–47, 2004.

[SS03] J. Santos and S. Staab. FONTE - Factorizing ONTology Engineering complexity. In *Proceedings of ACM K-Cap 2003 - International Conference on Knowledge Capture*, 2003.

[SSO+05] M.C. Schraefel, D.A. Smith, A. Owens, A. Russel, C. Harris, and M.L. Wilson. The evolving mSpace platform: leveraging the semantic web on the trail of the memex. In *Proceedings of the International Conference on Hypertext*, 2005.

[UCI+06] V. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, and F. Ciravegna. Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(1):14–28, 2006.

[VVMD+02] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. MnM: Ontology driven semi-automatic and automatic support for semantic markup. In *Proceedings of the 13th European Conference on Knowledge Management (EKAW)*, 2002.

[Wie94] G. Wiederhold. An algebra for ontology composition. In *Proceedings of the Workshop on Formal Methods*, 1994.

[WP06] T. Wang and B. Parsia. CropCircles: Topology sensitive visualization of owl class hierarchies. In *Proceedings of the 5th International Semantic Web Conference*, 2006.

[YSLH03] P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *Proceedings of the ACM Conference on Computer-Human Interaction (CHI)*, 2003.