NeOn: Lifecycle Support for Networked Ontologies

Integrated Project (IST-2005-027595)

Priority: IST-2004-2.4.7 – "Semantic-based knowledge and content systems"

D5.7.1 Plan for evaluating support for the NeOn methodology in the NeOn Toolkit

Deliverable Co-ordinator: **Martin Dzbor**

Deliverable Co-ordinating Institution: **OU**

Other Authors: **Mari Carmen Suárez-Figueroa (UPM), Asunción Gómez-Pérez (UPM)**

The primary purpose of this deliverable is to propose a coherent plan for assessing the NeOn methodology as such; i.e., to complement the partial studies of the individual methods reported in D5.6.1 and D5.6.2. Another objective is to propose test settings for the *applied* NeOn methodology; i.e., the methodology used in the context of designing and maintaining ontologies using the NeOn Toolkit.

Document Identifier:	NEON/2009/D5.7.1/v1.1	Date due:	February 28, 2009
Class Deliverable:	NEON EU-IST-2005-027595	Submission date:	February 28, 2009
Project start date:	March 1, 2006	Version:	v1.1
Project duration:	4 years	State:	Final
		Distribution:	Public

NeOn Consortium

This document is a part of the NeOn research project funded by the IST Programme of the Commission of the European Communities by the 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 Englerstrasse 28 D-76128 Karlsruhe, Germany Contact person: Peter Haase E-mail address: pha@aifb.uni-karlsruhe.de
Universidad Politécnica de Madrid (UPM) Campus de Montegancedo 28660 Boadilla del Monte Spain Contact person: Asunción Gómez Pérez E-mail address: asun@fi.upm.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: Jesús Contreras E-mail address: jcontreras@isoco.com	**Institut 'Jožef Stefan' (JSI)** Jamova 39 SI-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 – 655 avenue de l'Europe Montbonnot Saint Martin 38334 Saint-Ismier France Contact person: Jérôme Euzenat E-mail address: jerome.euzenat@inrialpes.fr	**University of Sheffield (USFD)** Dept. of Computer Science Regent Court 211 Portobello street S14DP Sheffield United Kingdom Contact person: Hamish Cunningham E-mail address: hamish@dcs.shef.ac.uk
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. Martino 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	**Food and Agriculture Organization of the United Nations (FAO)** Viale delle Terme di Caracalla 1 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	**Laboratorios KIN, S.A. (KIN)** C/Ciudad de Granada, 123 08018 Barcelona Spain Contact person: Antonio López E-mail address: alopez@kin.es

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 to the writing of this document:

OU and UPM

Change Log

Version	Date	Amended by	Changes
0.1	25-01-2009	Martin Dzbor	Outline & first draft public
0.2	13-02-2009	Martin Dzbor	Initial drafts of ch. 4 and 8
0.3	25-02-2009	Mari Carmen Suarez Figueroa	Extensions to assumptions about experiments, generic designs
0.4	03-03-2009	Mari Carmen Suarez Figueroa	Input to ch. 5 and 6
0.5	10-03-2009	Martin Dzbor	Input to ch. 9 + conclusions + references
0.6	15-03-2009	Martin Dzbor, Asun Gomez-Perez	Plans w.r.t. users, tools, timelines,… sec.3.4
0.7	20-03-2009	Mari Carmen Suarez Figueroa	Extension of ch.7
0.8	11-04-2009	Michael Erdman	Review, quality assurance
1.0	15-04-2009	Martin Dzbor	Attending to review comments, minor updates to content, corrections, clarifications
1.1	16-04-2009	Mari Carmen Suarez Figueroa	Finalization, reference clarifications

Executive Summary

The NeOn methodology, co-ordinated in WP5, has been designed in a structured and principled fashion by classifying a range of methods according to their position and place within an ontology design and use lifecycle. Due to this principled approach, the NeOn methodology goes beyond other, more narrowly focused methodologies, and as a consequence is being developed in steps.

The NeOn methodology is grounded in existing methodologies and methods, existing practices and previous experiences, and existing NeOn tools and has been designed taking into account the following criteria: generality, completeness, effectiveness, efficiency, consistency, finiteness, discernment, environment, transparency, essential questions, domain or scope, perspectives, understanding, usability, grounded on existing practices, flexibility, and tool independency. The first draft of the NeOn methodology was made available in D5.4.1, which was substantially revised and extended in D5.4.2.

The primary purpose of this deliverable is to propose a coherent plan for assessing the NeOn methodology as such; i.e., to complement the partial studies of the individual methods reported in D5.6.1 and D5.6.2.

Another objective of this deliverable is to propose a test of the *applied* NeOn methodology; i.e., the methodology used in the context of designing and maintaining ontologies using the NeOn Toolkit. In other words, in addition to assess the level of depth of the NeOn methodology, it is important, in our opinion, to also assess the degree of alignment between the methodology and the engineering environment implementing techniques that are abstracted and explained by the methods together forming the NeOn methodology.

Table of Contents

List of Figures

1. Introduction

One of the objectives of WP5 is to collate a coherent NeOn Methodology for building ontology networks comprising a range of concrete methods and guidelines to support activities involved in the typical ontology lifecycle and design process. In addition to the explicit objective of producing "a book of methods", WP5 also needs to provide qualitative and (where possible) quantitative evidence that using the NeOn Methodology leads to users being able to design ontologies faster and/or to better quality standards – in other words, to assess how effective NeOn Methodology is for the target user.

In the previous deliverables D5.6.1 [7] and D5.6.2 [6] we analyzed a range of methods that were at our disposal within the first three years and proposed several user studies and experiments with a subset of the methods, and reported on the execution and the results found. In particular, we focused on testing a rather limited sub-set of methods forming the NeOn Methodology: working with ontology design patterns, of a user study drawing upon the available support for ontology localization, ontology specification and for the use of ontology lifecycles. Furthermore, the previous studies were not necessarily coupled with testing the respective methods as a part of a longer process chain, as a part of a comprehensive methodology. That is, such studies on different activities and processes were performed independently.

To respond to the evolving understanding of the project (and user) priorities, as well as the evolving nature of the NeOn methodology and support for it in the NeOn Toolkit, in this deliverable we are formulating our proposals for experimenting with larger chunks of the overall methodology. The main reason why this deliverable presents *a proposal for a study* rather than a comprehensive study is in the fact that many techniques only emerged in the second half of period M25-M36. Methods associated with the majority of developed techniques thus started appearing around M34-M36, which would allow very little time to run the actual study. Hence, when planning the activity for WP5, we decided to split the planning and the execution stage into two reporting periods. First, in M25-M36 we consider possibilities and draw some commitments, then in M37-M48 we will work on putting the plan in action.

1.1. Overview of the document

We start the report by providing a brief overview of other works done in the project in the area of assessing parts of the NeOn Methodology; in particular, section 1.2 sums up the content of D5.6.2 [6], a deliverable that reports findings from testing individual methods. Next, in chapter 2 we present the principles of the NeOn Methodology, its structure, motivating scenarios and different faces. In the same chapter we also present a view of a "methodology applied within a tool" by means of Eclipse cheat sheets and our gOntt plugin [21], which is an ongoing work and subject to further investigation in M37-M48.

Chapter 3 is devoted to presenting the scope of assessment and evaluation; we discuss a range of appropriate approaches and strategies for evaluating both the effectiveness of the methodology and the behavioural efficiency of its inclusion in the NeOn Toolkit. In the same chapter we introduce specific test cases, and continue elaborating each of them in the subsequent chapters. We start with chapter 4, where initial studies with some aspects of the NeOn Methodology (mainly the ontology requirement specification activities) are briefly reported and summarized.

Thereafter chapter 5 to 9 propose partial segments, sequences of ontology development activities that can be carried out as standalone evaluation studies or can be included as a part in larger studies. The main reason we opted for this modular description of the study settings is to reflect different constraints in particular cohorts of participants (e.g., the tutorial explaining the NeOn Methodology range from 6-10 hours all way up to 120 hours).

In particular, chapter 5 covers the process around reusing and reengineering non-ontological resources; that is, XML schema data and RDBMS data by means of finding, reverse engineering, retrieving reuse patterns, transforming and possibly importing data from 'legacy' sources. Next, chapter 6 offers a similar perspective of reusing and reengineering, but for ontological resources, i.e., whole ontologies, ontology statements, and similarly. The primary activities focused on here include ontology resource discovery, comparison, reuse or reengineering, integration, consistency check, and possibly visualization and querying.

In chapter 7 we look at the activity of reusing ontological resources but in the evolving, changing setting. Hence, additional activities become more prominent, including ontology change capture, ontology evolution, workflow management, collaborative development, etc. Chapter 8 takes the ontological resources and focuses on the creation of modules from large ontologies, module composition and integration into consistent (or paraconsistent) ontology networks. Finally, in chapter 9 we raise a few propositions with respect to possibly testing some ontology contextualization activities, which include ontology mapping, alignment visualization and validation, and ontology customization. Chapter 10 wraps up the plans and discusses some implications of carrying out studies in the way proposed in the deliverable.

1.2. Summary of other studies with NeOn Methodology

Parts of the NeOn Methodology for building ontology networks have been evaluated in the past. For example, deliverable D5.6.2 [6] contains several reports on user studies carried out between months M25 and M36. Unlike this document, the studies reported in D5.6.2 looked at one concrete activity each, and one can argue they assessed the individual methods rather than methodology as a whole. Nevertheless, such partial studies are useful, and in this section their outcomes are briefly summarized.

First, the notion of ontology patterns (also called modelling components) has been described e.g., in NeOn deliverable D5.1.1 [19]. The concern in the user studies was to show the effects and benefits of using patterns in ontology engineering in a scientifically rigorous manner. There are several different aspects of patterns that may be studied and several types of pattern usage effects that may be explored and analyzed. Patterns in design and thus in ontology engineering are suggested to give three kinds of benefits: reuse benefits, guidance benefits and communication benefits. Reuse is concerned with constructing "better" ontologies due to the use of proved modelling chunks, fragments; the guidance is referring to the assistance and learning opportunities a structured pattern offers to the user, and finally, the communication is concerned with patterns as a tool for describing existing ontologies, modelling situations in a standard, shareable manner.

Initially two preliminary experiments on patterns were carried out in D5.6.1 [7]. The experiments gave some very interesting initial results and insights on how the users perceive, understand, and use ontology design patterns. Additionally, these experiments were an excellent opportunity to test the settings and discover good ways to experiment with ontology design patterns. In this sense, the experiments have also been used as a starting point, providing valuable experiences for designing a more complete set of experiments on ontology design patterns.

From the proposed experiment options, the fundamental one was carried out first – how people perceive the role of ontology design patterns and their potential benefits in developing higher-quality ontologies. Thus, the study took the depth-first approach in order to ensure the main expectations about the benefits of patterns are indeed explicitly identifiable in the design process. The approach focused on the primary role of patterns is important and carried out a similar set of studies with several user groups to ensure broader validity of our conclusions. Further particulars and detailed findings from the user studies with ontology design patterns are in D5.6.2 [6].

The second study concerned the support for ontology specification. The goal of this experiment was to test the benefits of using the proposed methodological guidelines for obtaining the ontology requirement specification document (ORSD) as an output of the ontology (requirement)

specification activity and as an input to the ontology development itself. The main motivation for this study was to learn about and assess the ease of comprehension and the usability of the proposed methodological guidelines for carrying out the ontology (requirement) specification activity. The study focused on the roles of software developers, analysts and various knowledge engineering practitioners who capture requirements that the ontology should fulfil, and the expected outcome of the study was the improvement of the guidelines. Further particulars from the user studies with ontology specification methodical guidelines are presented in D5.6.2 [6].

The third study reported in D5.6.2 concerned ontology localization. The ontology localization activity comprised the adaptation of an ontology designed in a particular natural language (e.g., English) to another concrete language and culture community (e.g., Spanish), as defined in [20]. The primary objective of this study was to evaluate the aspects related to the translation ranking techniques, where the task was to select the most appropriate translation for each ontology label. In particular, the study looked into two concrete ranking methods: the obtained output using manual and automatic operation, and the quality of translation. Further particulars and detailed findings from the user studies with ontology localization are in D5.6.2 [6].

In the parallel deliverable D5.6.2 [6] a range of additional studies with individual methods applied to a specific ontology development activity is presented and discussed. Studies were carried out with ontology lifecycle selection and specification. Also two studies were carried out that attempted to integrate preliminary methodological guidance with concrete tools – LabelTranslator was tested for usability (and alignment with the functional specification provided as a methodological guidance) and also Collaboration Server was deployed as a backend to testing the viability of developing ontologies in a collaborative setting, with managed editorial workflows. Deliverable D5.6.2 goes in further depth – both at the level of individual studies as well as trying to generalize some of the most important similarities in partial findings. The deliverable aims to offer a balanced view onto the components of the NeOn Methodology and highlights its strengths as well as weaknesses, and links the individual studies to this, larger-scale integrative proposal.

2. NeOn Methodology in a nutshell

In the NeOn project, one methodology and different methods are being defined to support the different activities of the networked ontology lifecycle. One of the goals of WP5 is to provide not only a collection of such methods at an abstract level, but also concrete guidelines, together with qualitative and quantitative evidence (where possible) that using the NeOn Methodology for building ontology networks, the individual methods, and techniques, ontologies can be built more effectively, efficiently, and generally, with higher-quality outputs.

As we already stated in [22], since mid 1990s there is a growing interest of practitioners in approaches that support the creation and management as well as the population of single ontologies built from scratch. Some well-recognized methodological approaches include e.g., METHONTOLOGY [8], On-To-Knowledge [18], and DILIGENT [15]) that provided guidelines to develop ontologies. All of them are a step forward, as they try to transform ontology design from an art to a principled engineering activity. However, they have at least four important limitations (see [22] for further discussion):

1. They lack guidelines for building ontologies by *reusing and reengineering the existing ontologies* and existing knowledge resources available in a particular domain.
2. They lack guidelines for *contextualizing an existing ontology* and plugging it in with existing ontologies that might be in continuous evolution.
3. They do not explain the ontology building process with the same style and granularity as available software engineering. That is, methodologies are not targeted to software developers or ontology practitioners[1] beyond ontology researchers.
4. The degree of maturity of the ontological engineering field is lower compared to the Software Engineering field. The long-term goal of the Ontology Engineering field is thus to reach a similar degree of maturity as Software Engineering.

Thus in the NeOn project, the NeOn methodology for building ontology networks has been proposed with a scenario-based approach in mind. A set of nine individual scenarios has been identified for building ontology networks. These can be combined between them to form more complex development scenarios. Each scenario is decomposed into processes and activities (see [20], [21], [22] and [23]). For each process or activity detailed and structured methodological guidelines are provided. In this sense, the NeOn Methodology was written with a process or an activity-centric approach in mind, and in a more descriptive way than prescriptive. Before going deeper into setting up the framework for assessing the utility of the NeOn Methodology, let us briefly summarize the motivating scenarios and use situations for the methodology.

2.1. Overview of the motivating scenarios

Based on the analysis of the three NeOn use cases and on the experience with building ontologies in different international and national projects, we have detected alternative ways to build ontologies and ontology networks. These can be seen as different scenarios in the NeOn methodology for building ontology networks. While details of the scenarios can be found in the previous deliverables [20, 22], the broad scope for the NeOn Methodology is as follows:

- Ontology (network) development from scratch, no reuse of existing knowledge resources;

[1] For ontology practitioners we mean ontology developers who are not necessarily ontology researchers

- Ontology (network) development by means of reusing non-ontological resources;
    - Ontology (network) development involving schema reengineering;
    - Ontology (network) development involving data adaptation and localization;
- Ontology (network) development by means of reusing existing ontological resources;
    - Ontology (network) development involving integration and reuse of resources;
    - Ontology (network) development involving mapping and/or merging of resources;
    - Ontology (network) development involving restructuring and design patterns;
    - Ontology (network) development involving data adaptation and localization

As argued in the previous deliverables, there is a substantial support for the activities covered by our first bullet point above. Hence, let us focus on understanding the latter two categories – development by *re-using non-ontological resources* and development by *re-using existing ontologies*. One may notice that the above list of scenarios does not contain some obvious knowledge engineering activities, such as knowledge acquisition or project documentation. These are somewhat orthogonal to the distinctive feature of our methodology (focusing on knowledge reuse), and can be seen as carried out regardless of which scenario, which specific reuse style one applies in ontology development.

This section tries to show the nine motivating scenarios in 'clusters' and focusing on the core, distinctive characteristic of each scenario. That is, we do not plan to go much further into the description of these generic scenarios. It is sufficient for the subsequent parts of this deliverable to realize that the NeOn Methodology is largely about choosing a sub-set of the paths through a fairly complex 'graph' of ontology development *activities* (cf. D5.3.1 [20] and D5.3.2 [21] where NeOn Glossary has been elaborated in terms of 60 distinguishable processes and activities).

In fact, this choice between different potential pathways through the ontology development process is so strongly visible that we proposed to dedicate it a standalone activity in our glossary – *ontology (development) scheduling*. While this is one of the meta-activities, i.e., an activity that does not directly manipulates an ontological model; it is precisely at this level of "meta planning" of the ontology design process, where the NeOn Methodology is at its strongest. For this reason, we will look at this relationship between the actual realization of the ontology development and our methodology in the next section, in more depth.

2.2. Realizing the value of the NeOn Methodology

There are in principle two complementary ways of how NeOn Methodology may be 'activated'. First, one may see it as a theoretical, at best conceptual, collection of the individual methods that can be collated in a book, studied, learned and/or otherwise trained. Secondly, one may perceive the NeOn Methodology as a referential framework comprising a collection of concrete guidelines for achieving concrete activities and a planning/scheduling tool enabling the user to compose and follow process chains most suitable for a specific ontology project.

The two views upon a methodology are in no way contrarian; in fact, they easily complement each other and one may see the theoretical collection of methods as a foundation for the referential framework, or the referential framework being the realization, implementation of the pool of theoretical guidelines. In this deliverable (D5.7.1 and subsequently in D5.7.2) we are pursuing and focusing more on the *referential utility* of our methodology, rather than its conceptual value. Hence, we propose to assess the benefits of the NeOn Methodology not only in its "taught form", i.e., as it could have been presented to a group of students and/or practitioners, in an artificial setting of a classroom, but mainly look at the NeOn Methodology in its "applied form".

When talking about applying NeOn Methodology, we may distinguish two levels at which one may come in an interaction with the aspects of methodology whilst developing ontologies and ontology networks: (i) using *generalized methods* advising the user on how best to carry out an activity (e.g., re-use of data from RDBMS repositories); and (ii) drawing upon *toolkit-specific guidelines/steps* allowing the user to accomplish a particular activity in a particular tool or toolkit. The importance of both levels comes from the following key requirements onto NeOn Methodology [22]:

- **Generality**. The methodology should be general enough and should not be driven to solve ad-hoc cases or problems. In our case, the NeOn methodology treats the development of ontology networks in general, and thus, should be expressible in the shape of generalized, tool-independent methods – sequences of primitive design activities.
- **Effectiveness**. The methodology should solve adequately the proposed cases that have a solution, regardless of its user. So, it should contain strong prescriptive, not merely descriptive elements. In our case, the NeOn methodology should be expressed in a way that any person (being a software developer or an ontology practitioner) could understand it. Furthermore, the methodology should be "translatable" to a set of tool-specific guidelines and to achieve this objective, it needs to be supported by a tool.
- **Discernment**. The methodology must be composed of a small set of structural, functional and representational components.
    - Regarding structural components, the methodology provides a collection of methods prescribing how to carry out activities involved in ontology design lifecycle.
    - Regarding functional components, the methodology includes processes, activities, tasks, inputs, outputs and restrictions, and should allow for their scheduling.
    - Finally, with respect to representational components, this methodology provides a graphical representation for the scenarios and for describing each process or activity.
- **Transparency**. The methodology must act like a white box, allowing the user to know at any time, what are the active processes or activities, what is being performed, who is performing them, etc. In our case, we explicitly define the actors, inputs, and outputs of each activity covered by the methodology, and in addition to that bind the individual methods using the notion of (ontology development) scheduling.

One such development in WP5 that follows the above rationale on embedding the NeOn Methodology into a specific software development environment is the feature of gOntt, a plugin for the NeOn Toolkit enabling the user to draft their own ontology project in terms of pathways through a sequence of processes and activities s/he identifies as relevant. Obviously, process planning and scheduling is valuable in its own right, yet the value of gOntt is actually in acting as a gateway to the NeOn Methodology (in the NeOn Toolkit, at the moment): In addition to understanding when and where a development process or activity should be carried out, gOntt is intended to allow the user to explore methodical advice and also to refer to the specific guidelines and steps detailing how such a process or activity can be realized. More on gOntt and some of its "methodological" features are provided in the next section.

2.3. Coupling NeOn Methodology into NeOn Toolkit

As described earlier, in order to achieve transparency, generality and discernment, it is beneficial to describe the NeOn Methodology in a standardized manner, for example, in the form of filling cards, workflows and guidelines for each identified process and activity, as used in D5.4.1 [22]. However, in order to maximize the effectiveness and the efficiency of the methodology, it was equally important to find a way to deliver the guidance to the user directly (or as closely to as possible) in the integrated engineering environment. To this extent, the design choice to base

NeOn Toolkit on a widely used Eclipse platform proved to be a very advantageous decision – also from the methodological point of view.

One common way of expressing (parts of) a methodology in software engineering is the use of so-called reference manuals. These are usually short or shortened accounts summing up the core aspects of "how to" achieve a particular outcome. Reference manuals are usually structured by problem(s) the individual entries help to solve, and they exist for a range of software development but also office and production environments.

In Eclipse, the functionality of providing reference guidance at the user's fingertips is provided by means of so-called "cheat sheets". Cheat sheets in Eclipse [24] can help guide the user through a series of steps in order to achieve an overall goal. Some steps can be performed by the cheat sheet directly, others are formulated sufficiently prescriptively so that the user can manually complete the step, and yet others are described at the level of principles. In addition, composite cheat sheets can guide the user through larger problems by breaking that problem into tasks. Each task may then be represented by a simple cheat sheet.

Figure 1, cut-out A shows the overall idea of tying the methodology with the toolkit on a generic example of an Eclipse Cheat Sheet for creating an Eclipse plug-in. Note that this example is taken from the Java integrated development environment, with respective Java cheat sheets. It is intended as an illustration of functional possibilities, rather than report on NeOn Toolkit status. In cut-out B we can see how a more complex design task is decomposed into a set of simpler tasks. When a task is selected, it is further decomposed into a sequence of activities, see cut-out C. In this sequence of activities there is some prescriptive guidance (e.g., pointer D) and some automated execution in the Eclipse GUI (e.g., pointer E).

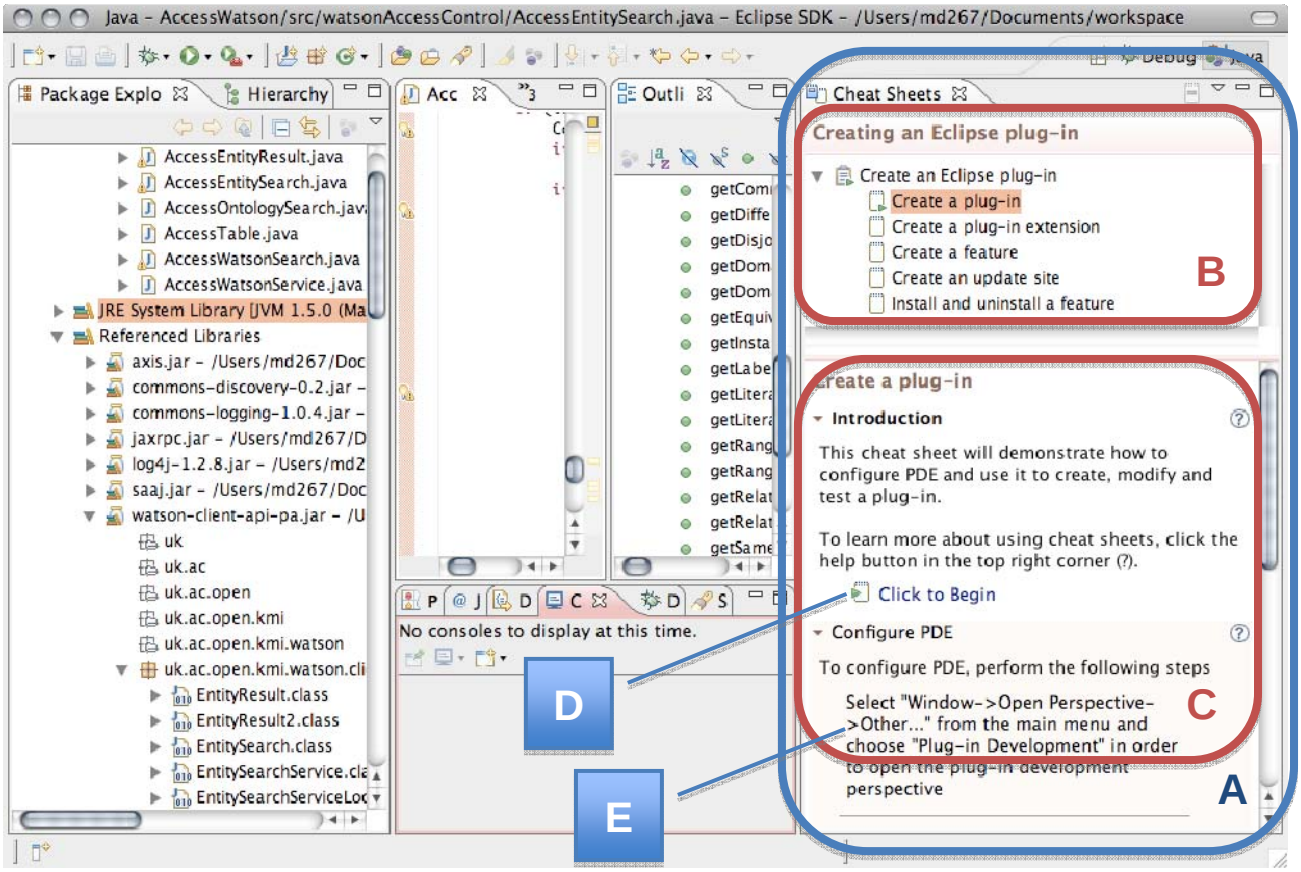

Figure 1. Example of using a cheat sheet in Eclipse as a reference to carry out a complex software engineering activity (here, "How to develop Eclipse plugin")

Cheat sheets in Eclipse, and thus in the NeOn Toolkit, may be launched using the dedicated menu item “Help > Cheat Sheets...”. Cheat sheets can also be launched from an introductory page or in response to a user’s click in the contextual menu. The main advantage of using cheat sheets as a way of rendering NeOn Methodology is that they are essentially an XML wrapped content (in our case, content coming from the existing structured descriptions of the methods). The work sketched in this section on merging the NeOn Methodology into a NeOn Toolkit plug-in is an ongoing work, subject to further investigation, specific design decision, etc. in M37-M48. Hence, this section is indented as a sketch of possibilities rather than a ‘fait acompli’ report.

As stated in deliverable D5.4.1 [22], the NeOn methodology for building ontology networks is a scenario-based methodology, in which each scenario is composed by a set of processes and activities that are defined in the NeOn Glossary. For each process and activity, the following methodological information is provided in the framework/context of the NeOn methodology:

- A **filling card** including process or activity definition, goal, input, output, who, and when. The use of such filling cards allows us to explain the information of each process and activity in the NeOn methodology in a practical and easy way. Each filling card follows the *filling card* template shown in Figure 2 and described in [22].

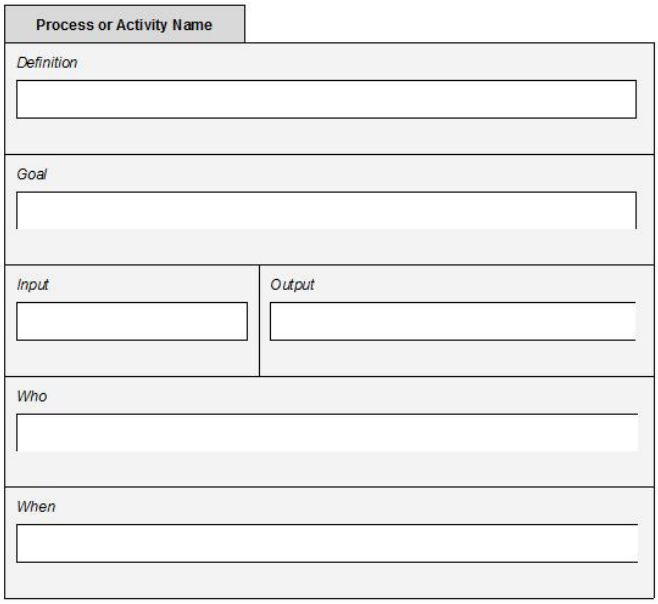

Figure 2. Filling Card Template [22]

- **Methodological guidelines** on how to perform the process or activity. A graphical **workflow** on how the process or the activity should be carried out is also included, with inputs, outputs and actors involved. Additionally, methods, techniques and tools supporting the process or the activity are proposed.

One of the ways to have the NeOn methodology integrated in the NeOn Toolkit is by means of the **gOntt plugin** [21], which is at the same time:

- a plug-in to help in scheduling an ontology network development, based on the methodological guidelines for establishing the ontology network lifecycle provided by the NeOn methodology; and
- a NeOn meta-tool with the goal of providing methodological information for each process and activity and information about the existing NeOn plug-ins for each process and activity.

In the latter role, gOntt intends to provide all necessary (methodological) information to perform the different processes and activities involved in an ontology project. That is, gOntt should include filling cards, workflows, and methodological guidelines in a way that is clear, easy and accessible for the user. Based on such objectives, the following possibilities have been analyzed to include in gOntt the methodological information presented in the NeOn methodology:

- **Including all methodological information in a configuration file**. All information would be store in a file, with XML format, that would be read to get the information.
- **Create a user-friendly interface**. The idea would be to present to the user an intuitive interface to show all necessary information.
- **Just show a PDF with all information**. The user could read all information about a methodological guideline from a PDF file.
- **Eclipse Help**. The user would get all information as Eclipse Help.
- **Cheat Sheets**. All information is showed to the user as in eclipse help, but this time, the user can interact with the data.

All these possibilities have different advantages and disadvantages that have been studied in order to select the best option for the different methodological information to be included in gOntt. Such pros and cons are the following:

- **Including all methodological information in a configuration file.**
  - **Advantages.** Easy to implement and to adapt in gOntt plug-in.
  - **Disadvantages.** The text format is limited. That is a problem taking in count the long texts that explain the different tasks of a workflow.
- **Create a user-friendly interface**.
  - **Advantages.** Very intuitive for the user.
  - **Disadvantages.** The text format is again limited. For long texts, it would be a problem.
- **Just show a PDF with all information**.
  - **Advantages.** Users are used to this kind of documents. The user could print or save the information very easy.
  - **Disadvantages.** The information will be not really integrated in the tool (in gOntt). In addition, the user may have a PDF Reader installed.
- **Eclipse Help**.
  - **Advantages.** Formatted text and integrated in the user interface. Eclipse Help system supports full HTML, incl. graphics, hyperlinks, etc.
  - **Disadvantages.** Limited interaction with the user.
- **Cheat Sheets**.
  - **Advantages.** Formatted text and integrated in the user interface. Possibility of interaction and automatic execution of different actions.
  - **Disadvantages.** Text Format is still limited. It recognizes just
, and similar tags in ASCII code. Besides, no images are allowed.

Taking into account the different possibilities and the study about their pros and cons, the decision was the following:

- To create an ad-hoc interface to show the filling card recreating its style as shown in Figure 2. It is usually a small amount of static information for each field that can be shown in an ad-hoc user interface. Figure 3 shows an example of a filling card included in gOntt plug-in.

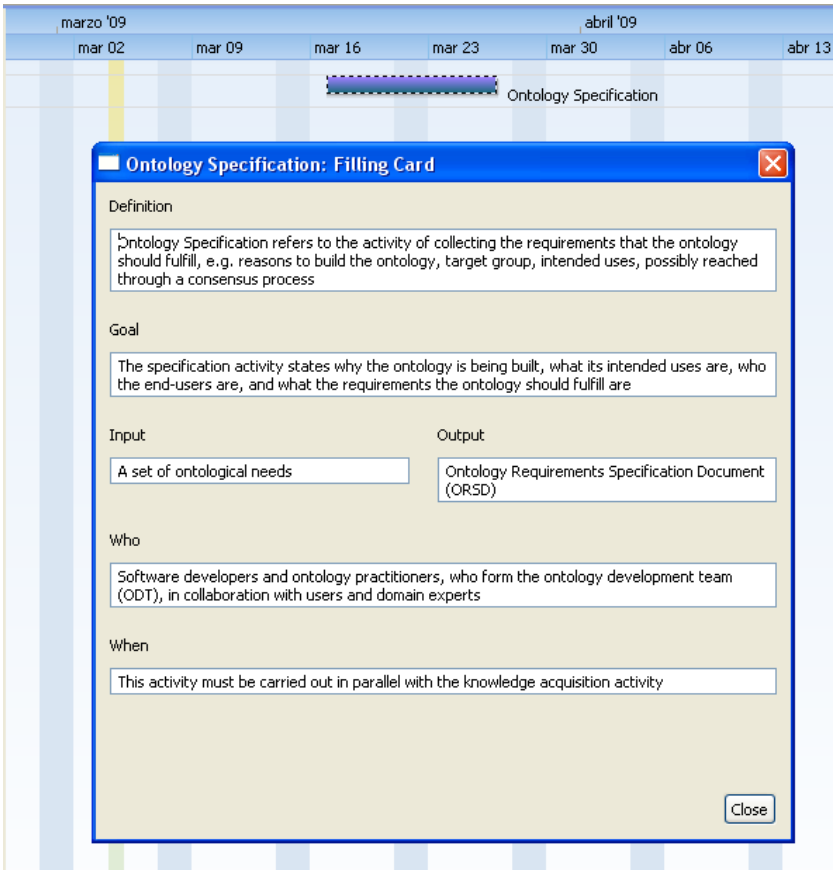

Figure 3. Example of the Ontology Specification Filling Card in gOntt

- To use cheat sheets to show workflows and methodological guidelines. Eclipse Cheat Sheets offer a very friendly interface to show information and, in addition, there is the possibility of creating a guide “step by step” to achieve a goal. That was one of the most important reasons of choosing that option. A workflow is an amount of tasks to do in a given order. The cheat sheets provide us with the necessary functionalities to create a guided workflow, helping the user to achieve the different tasks. Figure 4 shows an example of how cheat sheets are used in gOntt.

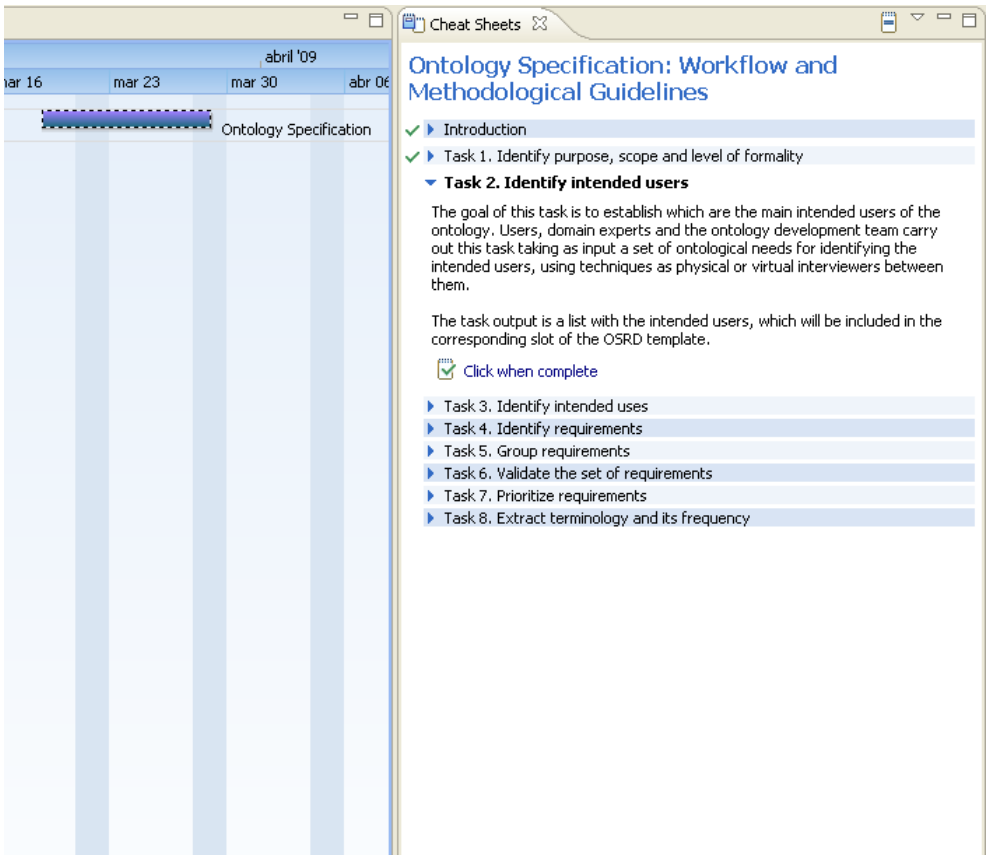

Figure 4. Example of workflow and methodological guidelines for the ontology specification activity packaged as an Eclipse Cheat Sheet.

One important requirement that needs to be resolved with respect to gOntt is its integration with other plug-ins. Here, there is a need to provide the gOntt plug-in with an extension point with which other plug-ins can easily 'advertise' (that is, expose) their supported activities to gOntt, which can then be automatically linked to the respective methodological 'hooks'.

3. Assessing NeOn Methodology

There are, in principle two different approaches how we can assess the utility of the NeOn Methodology. The first and most usual strategy can be described as looking at and comparing the differential effectiveness of the methodology. In other words, the main principle of this approach is to assess the *content* aspects of the methodology. That is, to compare the outcomes produced by a group that uses the methodology against the outcomes of a control group, the one not given any methodological guidance. However, this form of assessing impact may be difficult to realize in the practical setting. We will explain this in more depth in section 3.1.

The second approach to assessing a methodology is to see it in action. In this case, one does not necessarily need a control group "without any knowledge of the methodology" vs. test group "with knowledge of methodology". Instead we could be assessing the effectiveness of the *guidance* aspects of the methodology. That is, to compare the outcomes when following specific guidelines as opposed to merely having knowledge of basic methodical points. In this way, this latter style is assessing the applicability and the application of the methodology in a concrete setting, and is explained in more detail in section 3.2.

3.1. Preliminaries

Let us consider the above-mentioned two approaches to analyze the utility value of a methodology in more depth. We start by giving some rationale to the opinion that a standard comparison of "with vs. without methodology" is not a very practical way to obtain a valid set of results – at least in the context of teaching/training an ontology engineering methodology.

Whilst one may look at differential effectiveness of different tools in achieving a task (e.g., Protégé vs. NeOn Toolkit, in our case), one has to assume that both groups possess a minimal degree of knowledge allowing them to carry out the task. This is precisely the approach taken in the earlier studies conducted in the NeOn's WP4 [3, 4, 5], and this approach is also visible, e.g., in the study of using ontology design patterns [6]. Although the approach worked then to compare pairs of tools, transferring this approach to a comprehensive methodology test is not straightforward. In our case, since we are targeting ontology development, people have to know what this actually comprises, including details like creating and assessing alignments, reengineering database schemas, etc. Hence, there is the need for being trained in some foundational aspects of the methodology.

In our situation, it would be difficult to have a group totally without any guidance on methods, yet still proficient in their domain and willing to spend significant amount of time developing ontologies. Even more difficult would be to realize such an assessment as a part of a training course – whether academic or commercial. One can hardly imagine 'depriving' a subset of trainees of any methodological guidance to have them as a control group and only training the test group.

Similarly, since we are talking about rather advanced ontology engineering activities, it is not very informative – from the analytic point of view – to assess the differential using the technique of Pre- and Post-testing. Having explored this technique with training facilitators, we ran into difficulties with people simply not having sufficient amount of prerequisite knowledge allowing them to attempt a more complex reuse or mapping activity. Once we start introducing elements of the theory, there is a fine line where general vocabulary ends and methodological perspective starts. Hence, the outcomes would in many cases be statistically rather questionable. In many situations, however, one may simulate the pre- vs. post-testing; e.g., by looking at results of similar activities in the past ('pre-methodology test') and compare them with methodology-supported execution of those activities ('post-methodology test'). Hence, one part of our assessment is intended to exploit this

opportunity – comparing a facilitator's view of the qualitative performance of a similar group in the past with a group undergoing methodology training.

On the other hand, we mentioned the opportunity to assess whether and how people actually follow the methodology when designing ontologies and ontology networks. Admittedly, the question on *how* people follow a methodology is not the same as asking whether the methodology is any good. However, there is a certain degree of connectivity between the clarity and usefulness of the methodology on one hand side, and the analysis of whether, how and how frequently people draw upon methodical support available to them, on the other hand. Hence, one may construe a proto-hypothesis along the lines:

> "NeOn Methodology can be considered effectively delivered if people use the tools embedding it in their respective ontology development task. On the contrary, the methodology would be less effective if people consciously avoid following the method guidelines and method sequences offered in the ontology development toolkit."

Another argument towards assessing the methodology in practice reflects the primary target audience of the NeOn pursuits. From this perspective if we were to target students, "paper-based", theoretical and prescriptive methodology is often enough (and preferred) means how to internalize certain skills with ontology engineering. If however, one decides to better target practitioners form the disciplines ranging form software engineering, process modelling to fishery data analysis, one needs to look into different ways of engraining the methodology in their daily practice. For example, the notion of "pushed" methodology based on just-in-time available guidelines comes to the fore, and thus, it makes sense to assess to what extent people draw upon and follow such pushed methodical overviews and tool-specific guidelines.

3.2. Systemic description of proposed studies

In order to collect the descriptions of the planned experiments, we defined a guiding template to unify the format of the individual studies and to ensure that no relevant topic about the experimentation was left out. The topics we will touch for the respective user studies include the following:

1. *User study/experiment overview*:
    - *Understanding experiment motivation*: There is always a motivation either at a personal or at an organizational level for carrying out a user study. Questions to be answered are as follows:
        - Why are we performing this particular experiment?
        - What activity sequences do we see as relevant to a given experiment?
        - What do we want to prove or learn at a more generic level?
    - *Identifying user study target*: Each study targets a particular user group, from which we select our participants:
        - Who is going to benefit from this experiment and in which way?
        - Who is expected to participate in the study and in which role?
    - *Defining experiments goals*: In this part of the description we aim to identify the beneficiaries of the specific studies and the benefits that should be observed:
        - What are the goals for the experiment?
        - What methods, techniques, and their sequences are to be studied in a given user study?
        - What level of tool support we expect in the individual studies?

2. *User study/experiment preparation:*
    - *Identifying relevant characteristics*: The subject of each study has many different characteristics, but for the experiment only some of them are relevant.
        - Which attributes of the participant or methodology are going to be studied?
        - What type of experiment are we looking at (cf. section 3.1)?
    - *Defining metrics and criteria*: Metrics are used to describe some attributes of the situation when a user applies a methodology. On the other hand, criteria are needed to interpret the acquired data:
        - What metrics are suitable to assess the characteristics in focus?
        - Which criteria will be used to interpret these observations?
    - *Identifying the variables* (time, effort, cost, etc.):
        - Which variables affect a given situation and can be controlled?
        - Which variables affect a given situation but cannot be controlled?
    - *Defining the requirements*: Each experiment to perform may have some unique requirements to carry it out; hence we need to be aware of them:
        - How many times is each elementary experiment going to be repeated?
        - Are there a minimal number of people needed to run the study?

3. *User study/experiment design:*
    - *Deciding on the analysis procedure*: Once we have the data, we have to analyse it according to some procedure. Questions to be answered should be:
        - What data (e.g., ontologies) are necessary for a given study?
        - How will the observations be validated analysed?
        - Will we compare the results against a baseline? If so, what is the baseline?
    - *Defining the experimentation plan*: The plan of a user study includes the time and resources that will be used. Questions to be answered should be:
        - Who will perform the experiments and where?
        - When will the experiments be performed and what prerequisites are to be satisfied in order to realize the study?
        - How will the observations be analysed and summarized?

3.3. General assumptions

Section 3.2 lists quite a number of questions that need to be understood before an experiment or user study yields some findings. However, we are able to make several general assumptions that would apply across the collection of our studies with the NeOn Methodology. Hence, we list these assumptions here, rather than repeating them for each proposed scenario:

1. Related to experiment overview questions
    a. In terms of participant setup, we will constrain our scope to people whom we called in previous studies [3, 4] “non-power users”; i.e., people familiar with problem

domains but not necessarily with details of knowledge and ontology engineering. In this category, we are looking at two types of participants: (i) university students (usually at the postgraduate level) and (ii) software engineers, information analysts and other similar practitioners. In order to obtain a sufficient sample, it is likely that our populations would draw upon both types of participants.

b. The generic objective of each experiment is to provide hard evidence that participants (i) are aware of methodological guidance when using NeOn Toolkit, (ii) are able to translate a set of domain descriptive paragraphs in an ontology requirements specification document[2] (ORSD) and to translate a particular ORSD in an ontology project scheduling using gOntt, and (iii) are using the methodological cheat sheets within the NeOn Toolkit instead of asking the facilitator or blindly poking around. Hypotheses related to specific activity sequences are presented later, in the actual descriptions of the experiments.

c. In terms of tool support, our experiments rely on the extended version of NeOn Toolkit – the toolkit may not be equipped with all the relevant plug-ins though. Hence, a part of the user study is also the methodological and tool support for bringing in appropriate plug-ins into the engineering environment. Since only descriptive documents are given to the participants as an input to creating ORSD, participants should be allowed to use tools they are familiar with (e.g., MS Excel for capturing responses to filling cards or Notepad for taking notes) – since there is a good support for these auxiliary activities, no integrated support will be provided as a part of NTK.

2. Related to experiment preparation questions

a. With respect to types of experiments proposed, we will focus on testing the “pushed” methodology comprising referential guidelines, as discussed in section 3.1. While this will be the repeatable core of the experiment, we will also bring in (usually, in the discussion section) our experience with ‘pre-methodology’ performance of similar participants, but this would be more at a qualitative level.

b. In terms of what can be observed and analyzed, we can refer to [12] and in particular, to the following of his proposed levels of analysis: (i) user’s general satisfaction with (or reaction to) the studied object, (ii) effectiveness of the studied object in achieving goals, and (iii) behavioural efficiency of [presenting] the studied object. The studied object in our case comprises chunks of NeOn Methodology forming longer but systemic chains of development activities. Goals in this case reflect the quality of the (re-)developed ontologies.

3. Related to experiment design questions

a. With respect to input data, these may vary depending on the actual experiment, however, an effort will be made to use to the greatest possible extent ontologies and other content used in the experiments available freely and/or online.

b. In terms of time plans and constraints, the proposed experiment plans are directly dependent on two key aspects: (i) the availability of a stable version of NeOn Toolkit and respective plugins, and (ii) the implementation of the NeOn Training Plan, where practitioners are to be inducted into the core aspects of engineering ontologies using the NeOn products (toolkit, plugins and methodology).

c. As discussed in section 3.1, NeOn Methodology should be tested in conjunction with the NeOn Toolkit, in a series of comprehensive stories involving several ontology development activities and processes.

[2] Examples of particular ORSDs are shown in deliverable D5.4.1 [22].

d. Since we are looking at evaluating the NeOn Methodology in practice, we will generally consider the users who will obtain training in using the NeOn Toolkit and the NeOn Methodology in the dedicated tutorials.

e. The comparison will be generally based on the two groups of participants: (i) those who draw upon taught methodology (including referring to respective deliverables), and (ii) those who will have direct access to the snippets of the methodology from the NeOn Toolkit and its gOntt plug-in.

f. Both groups should be provided with the same input:

- In some cases with a set of paragraphs about a concrete domain and task to be modelled, and with such a set the users will write the ORSD. This should be the control group with the methodological guidelines in paper, and the test group with the help provided by gOntt in the ontology (requirement) specification activity.
- In other cases with an ontology requirements specification document[3] (ORSD) for a concrete domain and task.
- And in yet another case with the ORSD and with the ontology project scheduling scheme.

If only provided the ORSD, the users will plan their respective ontology project (the control group manually, the test group with the help of gOntt and cheat sheet guidance).

Once the plan is in place, both groups will continue using the NeOn Toolkit to accomplish the tasks; again, the test group with only 'textbook and slides' support, the test group equipped with suitable guidelines embedded into the toolkit.

g. We distinguish two broad approaches to segmenting users:

- Users who plan their ontology development project as a subset of the activities described in the subsequent chapters and
- Users who will be provided a plan including a subset of activities we want them to carry out.

With this approach we aim to observe how people use ORSD to plan and if the scheduling guidelines are useful in this process. The downside is that we could obtain many different plans and they would be hard to judge for being right or wrong. For this reason we propose to fix the plan in a subset of experiments in order to have the same ontology development activity sequences to evaluate and compare.

3.4. Target groups of participants

As we mentioned before, there are two broad types of user audiences we plan to target and inviting them to participate in the studies with the NeOn Methodology. The first category comprises students and junior researchers undertaking a series of course in ontology engineering; the second group is comprised of non-academic professionals who undertake various training sessions and tutorials of the selected aspects of the NeOn Methodology. Because these groups work to different schedules, and even within one category, different sessions follow different time constraints, the experiments in the subsequent chapters are presenting in a modular fashion. That is, we offer a range of experimental setups along a sequence of ontology development activities, which can be

[3] Examples of particular ORSDs are shown in deliverable D5.4.1 [22].

seen either as mini-experiments or as building blocks of a larger experiment – depending on the time constraints and the content of methodology tuition tutorials.

Thus, in the first category, we have already planned the tuition of the NeOn Methodology into the curriculum of the students at UPM. In particular, there will be two sets of PhD students and one set of Master-level students. We plan to attract around 20 participants for each of the PhD-level cohorts – first being scheduled in to receive the tuition between 21 April and 15 May 2009 (let us call it 'PhD-1'), the second group will have its courses between October 2009 and January 2010 (let us label it as 'PhD-2'). Altogether, each cohort will receive 8 hours of theoretical lectures that would be accompanied by 32 hours of hands-on practice and additional 2 hours for presenting the results of their mini-project.

We are planning to use the cohorts for both in-group analysis and between-group comparison. In-group analysis will focus on measures like comprehension of the guidelines, usefulness of the methodological guidelines, the style of using methodological guidelines, difficulties with the methodology, and overall quality of the developed ontologies. On the hand, between-group analysis enables us to test hypotheses arguing that a tighter integration of the methodological guidelines into an engineering toolkit is likely to produce better results and makes the following of the methodology easier for the users – all compared to users receiving verbal tuition only.

Due to very short time scale for these two courses, the preliminary plan is to start by giving people hands-on activities in the first half, which would then be followed by two more systemic uses of the methodology: (i) to specify the formal ontology requirement document and break it into a schedule of ontology development activities, and (ii) to choose the appropriate techniques and use the appropriate guidelines to carry out scheduled activities. The first use is estimated to take place at the end of week 2 of the course and will conclude with a presentation of 'mini project proposal' and justification of ontology development choices. The second use will taka place around week 3 and will conclude with a submission of a report and a developed ontology.

The third cohort that is already planned involves part-time students at the Master's level. The main difference between this and other two groups is their background – these are mostly professionals seeking to increase their qualifications rather than researchers or academics. Again, in-group analysis would give us clues as to how these users interact with the methodology; this time presented both verbally and integrated into the NeOn Toolkit. Between-group analysis enables us to compare how users with different background, academic degree and modelling skills differ in their use of the methodology.

In addition to these cohorts comprising students with different degrees and experiences, there are plans to deliver several training activities in April-May 2009 to the professionals likely to interact with the outputs of the use cases in NeOn. These cohorts are expected to attract between 10 and 25 participants from different FAO divisions, and a similar number of participants from the pharma industry. Since the training sessions will be much shorter for these groups, the scope for extensive testing is limited. Nevertheless, it is our intention to gather user feedback on the methodology as a part of hands-on activities and practices accompanying the theoretical tutorials.

All groups are expected to use NeOn Toolkit as the primary means to realize and implement their ontologies. In order to ensure validity of the studies, we need to constrain ourselves to using NeOn Toolkit v1.2.x branch with the plug-ins available for this version. Later in the year, the project plans to roll out NeOn Toolkit v2.x branch with the major difference being the shift from KAON2 to OWL API as the core datamodel. This shift is likely to reduce potential issues with interoperability of ontologies; however, since the final releases of the NeOn Toolkit v2.4 with a fully migrated API is planned for October 2009, this generates the risk of delays to experiments. Since our intention is to test the use of the toolkit and the integration of the methodology with the toolkit, we believe there is no loss in data generality if we retain the currently stable branch (v1.2.x) of the toolkit for all studies.

In terms of using methodology, two primary documents support the tuition – deliverables D5.4.1 ('NeOn Methodology v1' giving guidance on reuse and reengineering of non-ontological data) and

D5.4.2 ('NeOn Methodology v2' extending the guidance to ontology modularization, evolution, localization and design pattern application). Since these are both rather extensive deliverables, we intend to summarize the core points of the individual guidelines in the form of 'reference cards' to give people a simple way to access the content of the methodology tuition. Obviously, once the gOntt and its support of cheat sheets containing aspects of the NeOn Methodology pertinent to the NeOn Toolkit are implemented, the cheat sheets will effective take over as a 'user reference guide'.

4. Preliminary experiments with the NeOn Methodology

This chapter describes an experiment carried out at the UPM that consists in the development of an ontology network for the digital home using the NeOn Methodology. The experiment was performed in the Master's program, within the course "Ontologies and the Semantic Web".

The main objective of the course is to provide practical and theoretical foundations of the scientific, methodological and technological principles of the Semantic Web that may be used in the building of applications that integrate, combine and deduce distributed and heterogeneous information.

Students attending this course had different backgrounds: some held a five-year degree either in Computer Engineering or another Science field, whereas others held a three-year degree in Computer Science. Most of them had majored in databases, software engineering and artificial intelligence but none in ontology engineering. For a better understanding of the course topics, the teachers in charge taught the following lessons:

1. **Introduction to the Semantic Web**
2. **Computational Linguistics**
3. **Ontologies.** In this lesson, the theoretical foundation of ontologies is explained; also explained in detail is the NeOn Methodology for building ontologies (lifecycle, requirement specifications, reusing and re-engineering knowledge resources, ontology localization, etc.)
4. **The Semantic Web**

With the aim of studying the contents of the subject in depth, teachers established a set of mini-projects, which the students had to develop once the theoretical contents of each theme had been dealt with. The purpose here was to make the students build an ontology network in the domain of the digital home. Since this was largely an experiment with the methodology, the use of the Protégé integrated environment was compulsory in some mini-projects. In others, students were free to choose any ontology development tool they wished. Thus, the emphasis in this particular preliminary study was to see the NeOn Methodology in action, without any explicit tool support or integration, as we are now proposing in chapter 3.

The projects could be carried out in pairs or individually. The mini-projects related to the NeOn Methodology included the following activities:

- To establish with a maximum level of detail the requirements onto the domain ontology to be built.
- To identify the lifecycle model to be used for building the ontology; here students were asked to reuse existing knowledge resources.
- To plan the ontology development, tasks to carry out, resources to use, etc.
- To reuse non-ontological resources (lexica, thesauri, etc.) and, by means of re-engineering process, transform them into ontological resources.
- To reuse ontological resources and re-engineer them if necessary.
- To localize the outcome ontology in, at least, one additional language.

To pass the course, students were expected to have (a) carried out practical assignments related to each of the themes dealt with; and (b) written a final project compiling and improving all the practical assignments. This final project was to be presented in the classroom.

Once all the practical assignments and the final project had been done and submitted, we drew two kinds of conclusions: those related to the NeOn Methodology, and those related to the technology used.

4.1. Initial observations related to the NeOn methodology

- Students commented that the NeOn methodology was generally helpful.
- Students expressed that the NeOn methodological guidelines used during the mini-projects had enough detail and were well explained.
- Different examples included in the methodological guidelines were very useful for the students to inspire the decisions in their particular mini-project.
- Students agreed that performing the ontology (requirement) specification activity to obtain the ontology (requirement) specification document (ORSD) was essential to focus the search on different knowledge resources that were to be reused and to develop the required ontology.
- Students were very interested in the existence and use of ontology networks, and also in the reuse of existing knowledge resources.
- Students reused a lot of consensual knowledge (taken from ontologies and non-ontological resources).
- Ontologies obtained from this experiment were in general more systematic and had a higher degree of quality than ontologies obtained in previous years (when students did not use the NeOn methodology). Note that the level of quality was assessed by the lecturers using their established means of assessment.
- Another important conclusion was that the NeOn methodology could be used successfully with widely accepted ontology tools (such as Protégé, in some cases).

4.2. Initial observations related to the technology used

- As we suggested earlier, in some mini-projects Protégé was the tool of choice to implement the ontology. The primary reason was that many students had been exposed to this tool in the past, and the use of the same technology was deemed as a ‘gentle introduction’ to the principled, methodology-driven ontology engineering.
- When given a choice of tools, some students still decided to use Protégé for developing the ontologies because they had hands-on experience with this tool during the course. Other students, however, tried to use both Protégé and NeOn Toolkit (NTK) for developing the ontologies. The uptake of NTK was enthusiastic, yet NTK at the time of experiment was not stable yet, as it lacked full support for OWL ontologies that are available as reusable ontological resources. Hence, there is a motivation now to re-visit similar experiments with a full-fledged NTK in a stable state.
- Similar rationale was observed with respect to Topbraid Composer, the main reason for using this tool was its quick import of the (wrong and correct) OWL ontologies and the capability to work with several ontologies at once – both points observed in more generic users studies carried out in NeOn before [3, 4].
- Students used and liked the functionalities provided by Watson, both as a web interface and as a NTK plug-in, which helped them to locate an ontological resource quickly and easily, and subsequently re-use it in their own ontology at a very low cost (2-3 mouse clicks).

- Students commented that having in the NTK tools, plug-ins or other support means for extracting the entire ontology modules and also tools for analysing the reasoner steps would be very useful.
- Students suggested having a tutorial similar to the “Pizza ontology” tutorial in Protégé for the NTK – something that would introduce the unique elements of NTK to the user, as otherwise the user may not be aware of some of the toolkit functions.

4.3. Towards the planned experiments

Having briefly summarized the content of the previous evaluation work of the NeOn Methodology, we now move to describing the evaluation framework for the subsequent experiments and user studies. These are formulated in a modular rather than monolithic style because of different target groups being subject to different time constraints. Thus, in different user groups experiments with a varied degree of complexity and length can be realistically expected, and the proposals of respective studies need to be sufficiently flexible to allow for such a dynamic ‘experiment composition’ from partial sub-sets.

Chapters 5 to 9 propose partial segments, sequences of ontology development activities that can be carried out as standalone evaluation studies or can be included as a part in larger studies. The main reason we opted for this modular description of the study settings is to reflect different constraints in particular cohorts of participants (e.g., the tutorial explaining the NeOn Methodology range from 6-10 hours all way up to 120 hours).

In particular, chapter 5 covers the process around reusing and reengineering non-ontological resources; that is, XML schema data and RDBMS data by means of finding, reverse engineering, retrieving reuse patterns, transforming and possibly importing data from ‘legacy’ sources. Next, chapter 6 offers a similar perspective of reusing and reengineering, but for ontological resources, i.e., whole ontologies, ontology statements, and similarly. The primary activities focused on here include ontology resource discovery, comparison, reuse or reengineering, integration, consistency check, and possibly visualization and querying.

In chapter 7 we look at the activity of reusing ontological resources but in the evolving, changing setting. Hence, additional activities become more prominent, including ontology change capture, ontology evolution, workflow management, collaborative development, etc. Chapter 8 takes the ontological resources and focuses on the creation of modules from large ontologies, module composition and integration into consistent (or paraconsistent) ontology networks. Finally, in chapter 9 we raise a few propositions with respect to possibly testing some ontology contextualization activities, which include ontology mapping, alignment visualization and validation, and ontology customization. Chapter 10 wraps up the plans and discusses some implications of carrying out studies in the way proposed in the deliverable.

5. Supporting reengineering of XML and RDBMS data

This part of the proposed experiment with the NeOn Methodology concerns one of the activity sequences that is by many practitioners considered as vital in their deployment of semantic web technologies and ontologies. In particular, in this chapter we consider the experiment with testing methodological support for developing ontologies from the legacy data sources and with the legacy data in mind. This is indeed one of the main concerns, as indicated, e.g., in the descriptions of user requirements for fisheries [10, 11] and for pharmaceuticals [9].

For example, the Fisheries use case distinguishes between three broad user groups, ontology engineers, editors and validators, where the role of the first group is primarily concerned with various reuse and reengineering activities. In particular, their requirements 2.4.1 2, 2a, 2b, 2c, and 3 are directly expecting support for creating ontologies from non-ontological data, for interfacing RDBMS data via ontologies, for transforming, integrating and re-using existing models.

For example, there are number of non-ontological fisheries-related resources already available in FAO and suitable for conversion into ontologies. Examples of these are: thesauri (e.g., AGROVOC, ASFA thesaurus), classification schemes (e.g., the Fishery International Standard Statistical Classification of Aquatic Animals and Plants [11]), and existing FAO Knowledge Organization Systems (KOS), such as FAOTERM, Fisheries Glossary, etc. These resources are stored in relational databases or in XML documents [11].

A similar motivation can be found in [9] where the authors expect the support for 'lifting' the existing RDBMS, EDI-FACT, and similar schemas to the ontological level. Similarly as in the Fisheries domain, there is an expectation to provide support to the IT engineers and data managers with creating their organization specific invoicing models. Thus, the evaluation of how the NeOn toolkit and the NeOn Methodology support the fundamental needs of lifting, transforming, converting and otherwise reusing non-ontological data sources is a good motivator for the first part of our methodology evaluation proposal.

5.1. Overview and objectives

Formally, as explained in deliverable D5.3.2 [21] the problems motivating this chapter can be assigned to several different ontology development activities. For example, our methodology and the glossary account for reuse of non-ontological resources (NOR), for reengineering of NOR, for reverse engineering of NOR, and for transformation of NOR. Furthermore, there is a very rich classification of techniques listed in [22] with respect to working with NOR: methods and techniques for transforming RDBMS data and schemas, for transforming lexicons and thesauri, for transforming various generic XML represented resources, etc. The engineer must be therefore aware of a range of conceptual principles and techniques that would eventually drive his or her choice. S/he must be aware of a type of NOR s/he wants to reuse, its representational formalism, the selection of techniques applicable to a given problem; then s/he must choose the right technique, set it up appropriately (including connecting and initializing the data source), and finally s/he must be able to check and adjust the settings during the reuse process.

The first proposal therefore focuses on the methodological support for the process of reusing NOR, which normally includes such activities as the initial creation of an ontology project in the NeOn Toolkit, import of various supportive ontologies (e.g., SKOS, Dublin Core, etc.), reverse engineering and analysis of NOR, the actual transformation of NOR (which may be repeated several times with slight variations), and the forward engineering of the new ontology, as shown in Figure 5. It is therefore a reasonably self-contained set of activities that is likely to be carried out by one type of user – a data schema manager or a knowledge engineer.

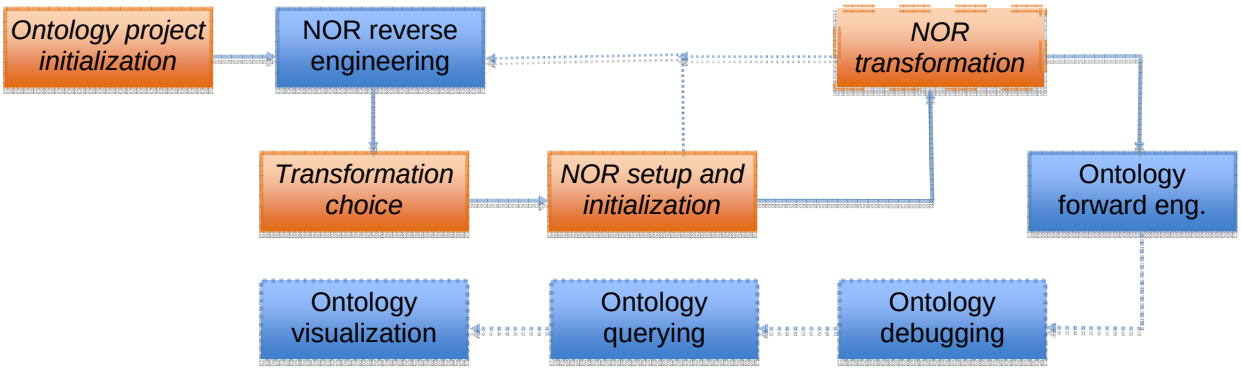

Figure 5. A process of ontology development steps[4] relevant to reusing NOR and including selected ontology development activities (in blue) and other auxiliary activities (in orange).

From the activities listed in Figure 5 this experimental proposal should be concerned primarily with the methodological support for the initial configuration of an ontology project in the NeOn Toolkit, for the planning and selection of the right NOR transformation technique, for the setup of the chosen NOR transformation technique and for any adjustments (if necessary).

In the above sequence box labelled as "NOR transformation" is actually a process and may include several ontology development activities applicable to different resources – for example, some resources may need ontology (label) translations as a part of transformation process.

5.2. Assumptions and experiment preparation

First and foremost, this part of the proposal depends on the availability of methodological guidelines for the above-mentioned ontology development activities. Second, the proposal depends on the availability of a stable release of NeOn Toolkit supporting OWL as the main ontology representation formalism and the availability of the respective plug-ins realizing at least some of the NOR transformation techniques. Third, this setup is dependent on the provision of detailed guidelines via the gOntt plug-in and the appropriate cheat sheets. Let us look at each of the categories in more detail.

Reuse, reengineering and transformation of NOR is one of the most detailed areas covered by the NeOn Methodology. The methods and guidelines supporting NOR reuse have been present in the first release of the NeOn Methodology [22], and as such were partially assessed with real user data in the past. Hence, we can safely assume that methods and guidelines to support the activity sequence as sketched in Figure 5 are in a sufficiently stable state to bootstrap a more comprehensive experiment.

Second, with the release of NeOn Toolkit v1.2, there is a full editing support for OWL available. One can configure the toolkit to start a new OWL project (in addition, one can also setup a FLogic project). OWL editor and other utilities (such as basic visualizations) are also in a stable state. In terms of plug-ins needed for this experiment, we expect the following to be involved:

- ***ODEMapster*** [available in plug-in store] … this allows users to create mappings between NOR schemas and ontological elements (expressed in R2O language), execute and query the mappings.

[4] Note that the steps shown in the figure in orange are not included in the NeOn Glossary. Informal choice boxes (orange) depict the point where one or another ontology development activity (blue) needs to be selected.

- ***ODEMapster Wizard*** [available in plug-in store] … this plug-in adds to the NeOn Toolkit the functionality to execute existing R2O mappings using the ODEMapster processor, and supports the full set of R2O primitives.
- ***XML Mapping*** [available for v1.1] … this allows users of the Neon Toolkit to translate XML schema constructs into ontology classes, attributes and properties, with the order of element being either ignored or preserved
- ***RaDON*** [available in plug-in store] … this plug-in helps the user to deal with inconsistence and incoherence occurring in networked ontologies. It is capable of diagnosing (spotting) the source(s) of inconsistence and incoherence in single or networked ontologies, as well as repairing the identified issue either fully automatically or manually by the user.
- ***LabelTranslator*** [available in plug-in store] … this plug-in helps the user to obtain the most probable translation for each ontology label, to capture the linguistic information associated with concepts using a linguistic model repository (LIR), and to maintain the ontology and linguistic information synchronized.

In addition to the core NOR reuse plug-ins, other auxiliary plug-ins may be involved; e.g., the plug-in supporting initial ontology schema modelling, the plug-in for visualizing resulting ontology relationships and/or summaries, possibly SPARQL for ontology querying.

The third assumption for successfully running this part of the experiment is the availability of the gOntt plug-in with respective cheat sheets. As mentioned in section 2.3, this work is at the time of writing this deliverable in progress, thus, the experiment needs to be planned after the cheat sheets and gOntt reach the stable release.

5.3. Specifics of the experiment design

Since legacy data tend to be organization specific, it is hard to recommend one specific NOR to be used throughout the experimental sessions targeting the objectives described in this chapter. We propose to exploit one of the standard NOR types used by the participants to our experiment. Since the methodological process needs to be explained to the users, it is likely the experiment will take place as a part of a training programme. Such a setup proved to be successful in running experiments with applying simple ontology design patterns (cf. chapter 2 in [6]), and thus the choice of NOR data to be used as a testing sample can be easily adjusted to the group taking part in the training session – e.g., one of the ASFA or AGROVOC glossaries for FAO trainees and EDI-FACT / legacy invoicing RDBMS schemas for KIN trainees. For the purpose of running this experiment on a larger sample of participants, e.g., PhD students, the choice of NOR data needs no additional adaptation, as most NOR in both Fisheries and Invoicing data sets are in the public domain, and hence, they may be exploited beyond FAO or KIN.

5.4. Experiment analysis and discussion

In terms of evaluating the results of reusing, reengineering and transforming NOR to ontologies, one can use different measures of success. First, we can measure the *behavioural equivalence*; i.e., requesting the users execute several pre-defined information retrieval queries on both the original NOR data set and on the resulting ontology. The hypotheses about the success of the NeOn Methodology can be then formulated as follows:

- H1.1: The results of data queries performed on the reengineered ontologies are equivalent to the results of data queries executed directly over the original NOR data sets.

- H1.2: The results of data queries performed on the ontologies that were created with a direct help of the methodological cheat sheets and toolkit guidance will show a higher degree of equivalence ('precision') with the data queries in the original NOR data sets, when compared to the data queries performed on ontologies developed without any direct methodological support embedded in the toolkit.
- H1.3: Selection and transformation of the non-ontological resources to be reused will be faster and better justified (explained) for the user group using built-in methodological guidelines compared to the test group without such methodological guidance.
- H1.4: Selection and transformation of the non-ontological resources to be reused will be faster and better justified (explained) for the user group using built-in methodological guidelines compared to the test group that receives no tuition of the NeOn Methodology.
- H1.5: The use of built-in guidance for executing NeOn Methodology will lead to more detailed documentation of modelling decisions compared to test groups (without the built-in support and without any tuition of the NeOn Methodology).
- H1.6: The use of built-in guidance for applying NeOn Methodology will lead to better modelling practices – that is, to the correct choice between classes vs. individuals, singular vs. plural, property definition vs. restriction definition, etc.

Second, we can measure the *overall quality and coverage* of the resulting ontologies. Unlike the previous measure, this one would require an existence of a 'gold standard'; i.e., the ideal outcome of the NOR reengineering process. Within this study, the quality evaluation would be primarily on the functional and usability level (see [16]), although some of the methods used additionally cover the structural level to some extent:

- Coverage of the problem set out by ORSD (functional measure)
- Usability of the outcome (usability measure)
- Modelling issues/unresolved problems (structural and functional measure)

The latter measure assumes that we can identify a minimal set of concepts and properties the resulting ontology should contain, and look for those to calculate the degree of "coverage"; i.e., to what extent does the ontology produced in the experiment cover the minimal set of expected terms and their relationships. With respect to usability, the reengineered ontologies can be analyzed regarding their clarity, whether the concepts are formally defined and axioms included, whether the local names are clear and unambiguous as a result of setting up R2O mappings, and whether the participants followed some naming convention, including the ontology containing labels and comments. The third aspect is concerned with problems in the ontologies. Even an ontology that exactly covers the competency questions and the ORSD might have some inherent problems that would for example hamper its usage in a system or make it less flexible and not so easy to extend or update. Such issues are not easily discovered, but are often closely related to slightly strange modelling choices. These problems can be on both the structural and functional level.

With respect to the second measure, the hypotheses about the success of the NeOn Methodology can be then formulated as follows:

- H1.7: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling mistakes) will be higher if participants follow the in-built step-by-step guidance implementing the NeOn Methodology in the NeOn Toolkit, when compared with the users who do not have such a tight methodological guidance.

- H1.8: The use of built-in guidance for applying NeOn Methodology will lead to better modelling practices – that is, to the correct representation of the reused entities in a given ontology formalism (presumably OWL).
- H1.9: Ontologies produced by reengineering non-ontological resources and following the tight methodological guidance will show a higher overall quality ranking as judged by peers (e.g., in systems like Cupboard).

6. Supporting ontological resource reuse and ontology localization

One of the nine scenarios identified in the NeOn Methodology, and motivating it, is Building Ontology Networks by Reusing Ontological Resources. The underlying principle of such a scenario is that reusing existing ontological resources reduces time and costs associated to the ontology development. Moreover, ontologies developed by reuse are also expected to spread good practices (from well-developed ontologies) and increase the overall quality of ontological models. Besides, the reuse of ontological resources is encouraged by a recent increase in the number of online available ontologies, ontology libraries and repositories.

In the opinion of the experts participating in NeOn it is very important for ontology practitioners and software developers to take an intermediate approach, i.e., an approach in which a solution to the modelling problem lies between not reusing any knowledge resource at all and reusing too many knowledge resources. Where no reuse basically points to the outdated scenario of developing ontologies 'from scratch', the latter challenge is equally dangerous. In particular, if developers are overwhelmed and feel under pressure to incorporate into their developments any and every resource that looks remotely relevant, resulting ontologies may get far too complex and thus less likely to be reused in their own right. This would be clearly not desirable from the NeOn's point of view and general focus.

Another identified scenario in the NeOn Methodology is one that involves the ontology localization activity, in which the translation of all ontology terms into another natural language (Spanish, French, German, etc.) different from the language used in the conceptualization, using multilingual thesauri and electronic dictionaries (e.g. EuroWordNet). This scenario comes up due to the access to top-quality ontologies (e.g., Galen, CYC, or AKT) is in many cases free and unlimited for users all around the world, yet most of these ontologies are available only in English. Furthermore, many ontologies resulting from the principled development process are also in English, which makes it harder (if not impossible) to deploy them in new ontology development projects running in non-English environments.

Thus, a vicious circle arises, with two forces acting against each other – on the one side we want to facilitate as wide reuse of ontologies as possible, on the other, reuse is often hampered by the lack of availability of ontological resources in a particular language. Hence, this experimental setup covers these two forces, and attempts to gain some insight into how one can improve on the chances to reuse an ontology if principled support for ontology localization is given.

6.1. Overview and objectives

Formally, as defined in [21] the problems motivating this chapter can be assigned to several different ontology development activities. The NeOn Methodology and the NeOn Glossary define ontological resource reuse as using available ontological resources (ontologies, modules, statements, or ontology design patterns) in the solution of different problems.

The ontological reuse process is often influenced by the type of ontological resource to be reused. On the one hand, general or common ontologies provide conceptualization of generic topics such as time and space. On the other hand, domain ontologies provide knowledge of a concrete domain such as medicine, pharmacy, fisheries, etc. Such ontologies can be helpful in cases when a domain ontology in the same domain is being built. Unlike in the case of general or common ontologies, which are reused as a whole, we distinguish here different levels of granularity in the reuse of domain ontologies: (1) ontologies can be reused as a whole; (2) one part or module of a domain ontology is relevant for reuse; and (3) the reuse of ontological knowledge at the level of a single statement or axiom.

The reuse process, as we see it for the purposes of this experiment, normally involves the following activities:

1. Searching for candidate ontological resources that satisfy (fully or partially) the ontology (network) requirements established in the ORSD.
2. Assessing found ontological resources by inspecting their content and granularity of the resources to find out if they satisfy their needs.
3. Comparing ontological resources taking into account a set of criteria identified by the software developers and ontology practitioners (e.g., ontology language, ontology coverage, existing ontology documentation, etc.) and the ontology (network) requirements.
4. Selecting the most appropriate ontological resource(s) based on the comparisons.
5. Defining the reuse mode: at this point, software developers and ontology practitioners need to decide how they will reuse the selected ontological resources. There are three modes: (1) the selected ontological resources will be reused as they are; (2) the ontology reengineering activity should be carried out with the selected ontological resources; or (3) ontological resources will be merged to obtain a new ontological resource.

Hence, as can be seen from this sketch, the set of activities mentioned is sufficiently rich and reasonably self-contained that is likely to be carried out following the methodological guidelines provided by the NeOn Methodology [22, 23] and using the NeOn Toolkit.

Additionally, the next activity where methodological input is likely to be critical is supporting the localization of ontologies in order to build multilingual ontologies. Such ontologies would facilitate the access to ontological knowledge by non-English users.

6.2. Assumptions and user study setup

First and foremost, this part of the proposal depends on the availability of methodological guidelines for the above-mentioned ontology development activities. Second, it also depends on the availability of a stable release of NeOn Toolkit supporting OWL as the main ontology representation formalism and the availability of the respective plug-ins supporting at least part of the reuse process. Third, this setup is dependent on the provision of detailed guidelines via the gOntt plug-in and the appropriate cheat sheets. Let us look at each of the categories in more detail.

Regarding the first aforementioned part, prescriptive methodological guidelines for reusing ontological resources, focused on general or common ontologies, domain ontologies as a whole, and ontology statements are provided in the NeOn methodology [22]. Additionally, prescriptive methodological guidelines for localizing ontologies are included in D5.4.2 [23].

Second, with the release of NeOn Toolkit v1.2, there is a full editing support for OWL available. One can configure the toolkit to start a new OWL project (in addition, one can also setup a FLogic project). OWL editor and other utilities (such as basic visualizations) are also in a stable state. In terms of plug-ins needed for this experiment, we expect the following to be involved:

- ***Watson*** [available in plug-in store] … this supports knowledge reuse and allows the ontology developer to query Watson to find existing descriptions of selected entities in the edited ontology, and to reuse (i.e., integrate) these descriptions in the edited ontology.
- ***Oyster API/GUI*** [available in plug-in store] … this plug-in provides services for storage, cataloguing, discovery, management, and retrieval of ontologies (and related entities) from a network of peers based on the ontology metadata definitions.
- ***OWLDiff*** [available from developers, third-party] … this supports ontology comparison and merging; OWLdiff serves the same purpose for ontologies as *diff* does for textual files.

- ***RaDON*** [available in plug-in store] … this plug-in helps the user to deal with inconsistence and incoherence occurring in networked ontologies. It is capable of diagnosing (spotting) the source(s) of inconsistence and incoherence in single or networked ontologies, as well as repairing the identified issue either fully automatically or manually by the user.
- ***LabelTranslator*** [available in plug-in store] … this plug-in helps the user to obtain the most probable translation for each ontology label, to capture the linguistic information associated with concepts using a linguistic model repository (LIR), and to maintain the ontology and linguistic information synchronized.

In addition to the core plug-ins supporting ontology resource reuse and integration, other auxiliary plug-ins may be involved; e.g., the plug-in for visualizing summaries of the discovered ontologies and visualizing the developed ontological relationships, possibly SPARQL for ontology querying and OWL documentation plug-in may be included to improve the ontology quality standard and the its 'chances' to be reused. Overall, all core plug-ins are available and most of the other mentioned plug-ins in auxiliary roles are also available, at least as advanced working prototypes. Obviously, there is likely to be further development in all the plug-ins, most notably with Watson and ontology summary visualization, which are planned to be integrated and cross-linked to better support previews for the discovered ontologies. Nonetheless, this makes the setup for this part of the experiment proposal aligned with the technological developments and implementation, and thus the study as motivated in this chapter is realistic to be accomplished during M37-M48.

The third assumption for successfully running this part of the experiment is the availability of the gOntt plug-in with respective cheat sheets. As mentioned in section 2.3, this work is at the time of writing this deliverable in progress, thus, the experiment needs to be planned after the cheat sheets and gOntt reach the stable release.

6.3. Specifics of the experiment design

In this experiment the idea is to provide participants with a particular ORSD including requirements for building an ontology in a selected domain. Additionally, we provide participants with a concrete schedule they should follow to develop the requested ontology. Such scheduling will include the whole set of processes and activities to be performed during the ontology project; and at least, such a plan will include the ontological resource reuse process, the ontology localization activity, and the ontology evaluation activity. In the case of the reuse process we will involved all the aforementioned types of reuse.

This experiment is following a screenplay that has been tested several times in NeOn, e.g., in the user studies carried out in WP4 [3, 4, 5]. In each case, a set of existing ontologies was identified, as available on the Web, and a sequence of tasks with a growing degree of complexity was given to the user. For each task one can easily observe whether or not the participant follows the methodological guidance.

In order to bootstrap this experiment, the main challenge is in identifying an appropriate sub-set of ontologies in the public domain that would include ontologies of different quality, size, and coverage. Yet, this set needs to be 'controlled', in the sense of being comprehensible to the study participants, being sufficiently informative to allow someone who is not subject expert in a given field to discover, reverse engineer and integrate chunks of ontologies provided in the set.

6.4. Experiment analysis and discussion

In terms of evaluating the results of this experiment, one can again use different measures. As in section 4.4, we can measure the overall quality and coverage of the resulting ontologies. As before

this would require an existence of a 'gold standard'; i.e., the ideal outcome. Within this study, the quality evaluation would be primarily on the functional and usability level (see [16]), although some of the methods used additionally cover the structural level to some extent:

- *Coverage of the problem set out by ORSD* (functional measure). This measure assumes that we can identify a minimal set of concepts and properties the resulting ontology should contain, and look for those to calculate the degree of "coverage"; i.e., to what extent does the ontology produced in the experiment cover the minimal set of expected terms and their relationships. Such set of terms to be covered by the ontology is already included in the ORSD.
- *Usability of the outcome* (usability measure). The integrated ontologies can be analyzed regarding their clarity, whether the concepts are formally defined and axioms included, whether the local names are clear and unambiguous as a result of including the statements and/or ontology components found on the (Semantic) Web, and whether the participants followed some naming convention, including the ontology containing labels and comments.
- *Modelling issues/unresolved problems* (structural and functional measure). Even an ontology that exactly covers the competency questions included in the ORSD might have some inherent problems that would for example hamper its usage in a system or make it less flexible and not so easy to extend or update. Such issues are not easily discovered, but are often closely related to slightly strange modelling choices. These problems can be on both the structural and functional level.
- *Use or absence of ontology design patterns* (structural measure). In this case, one can also observe whether or not some ontology design patterns were applied in the development of the ontological resources.

Specific hypotheses we can pursue with these measures may include the following:

- H2.1: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling problems) for ontologies developed by reusing existing ontological resources will be higher in comparison with ad-hoc development of the ontology from scratch.
- H2.2: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling problems) for ontologies will be higher if participants follow the in-built step-by-step guidance implementing the NeOn Methodology in the NeOn Toolkit, when compared with the users who do not have such a tight methodological guidance.
- H2.3: The reuse ratio (measure of how many time a chunk of a particular ontology is found in another ontology by means of a module or import) will increase for languages other than English compared to the current state when no explicit localization is supported in the ontology engineering toolkits.
- H2.4: The use of built-in methodological guidance will lead to a more structured and transparent representation of differences between concepts, terms, labels, linguistic alternatives, etc.

7. Supporting ontological resource reuse and evolution

The idea of this experiment is to see what impact on the developed ontology does the reuse of different existing ontological resources have, and how evolution of ontologies developed by (full or partial) reuse can have impact on the reused ontological resources. Ontologies are fundamental building blocks of the Semantic Web and are often used as the knowledge backbones of advanced information systems. As such, they need to be kept up to date in order to reflect the changes that affect the lifecycle of such systems (e.g., changes in the underlying data sets, need for new functionalities, etc). Additionally, the reuse of ontological resources is encouraged by a recent increase in the number of online available ontologies, ontology libraries and repositories.

As already mentioned in the previous chapters, one of the nine scenarios identified in the NeOn Methodology, and motivating it, is Building Ontology Networks by Reusing Ontological Resources. The underlying principle of such a scenario is that reusing existing ontological resources reduces time and costs associated to the ontology development. Moreover, ontologies developed by reuse are also expected to spread good practices (from well-developed ontologies) and increase the overall quality of ontological models.

Within their lifetime, ontologies undergo changes. They evolve for example to correct errors or adapt to new knowledge about the world, or changed circumstances. Moreover, during the ontology development process sometimes errors are not spotted and have to be corrected later, i.e. after initial deployment. However, as ontologies may depend on several others and may also be related to other elements (e.g., instances, mappings, applications, metadata, etc.), one has to be careful with making changes. Dealing with ontology changes usually involves the execution of many related tasks identified in the context of the ontology evolution activity.

Thus, this experimental setup covers these two forces (reusing ontological resources and evolving ontology networks), and attempts to gain some insight into how one can improve on the chances to reuse an ontology if support for ontology evolution is given.

7.1 Overview and objectives

Formally, as defined in [21] the problems motivating this chapter can be assigned to several different ontology development activities. The NeOn Methodology and the NeOn Glossary define: (a) ontological resource reuse as using available ontological resources (ontologies, modules, statements, or ontology design patterns) in the solution of different problems; and (b) ontology evolution as the activity of facilitating the modification of an ontology by preserving its consistency. Ontology evolution can be seen as a consequence of different activities during the development of the ontology.

The process of reusing ontological resources is often influenced by the type of ontological resource to be reused. Such a process normally involves the following activities:

1. Searching for candidate ontological resources that satisfy (fully or partially) the ontology (network) requirements established in the ORSD.
2. Assessing found ontological resources by inspecting their content and granularity of the resources to find out if they satisfy their needs.
3. Comparing ontological resources taking into account a set of criteria identified by the software developers and ontology practitioners (e.g., ontology language, ontology coverage, existing ontology documentation, etc.) and the ontology (network) requirements.
4. Selecting the most appropriate ontological resource(s) based on the comparisons.

5. Defining the reuse mode: at this point, software developers and ontology practitioners need to decide how they will reuse the selected ontological resources. There are three modes:
    a. the selected ontological resources will be reused as they are;
    b. the ontology reengineering activity should be carried out with the selected ontological resources; and
    c. ontological resources will be merged to obtain a new ontological resource.

Relating with ontology evolution, the following tasks should be performed:

1. Requesting a change in the ontology or ontology network.
2. Planning the change, where the change request is analyzed and it is determined why the change needs to be made and which part of the ontology is affected by the change.
3. Implementing the change. Implementing the changes is of varying difficulty, depending on the impact of the requested change. While some change can be as easy as adding or removing a subclass, other changes can require complex operations and restructuring of the ontology.
4. Verifying and validating. Before the ontology is considered evolved completely, the last step deals with assessing questions whether the right ontology is built, and whether it is built in the right way. During this assessment, usually not only the ontology originally modified is verified in isolation, but in general this activity can include the verification of other artefacts related to the ontology to ensure they were not changed in a wrong way or they have an unexpected behaviour.

Finally, there are several aspects with respect to ontology evolution that are interesting for the purposes of evaluating the methodology. For instance, in practice the evolution of ontologies is no longer driven by single users; often an entire team of ontology editors is in charge of the development and maintenance of the ontology. Each member of the team may have different role and associated permissions and usually they follow a well-defined process to control when and how the ontology can change (evolve). Furthermore, the members of the team are often geographically distributed (as is the case with the AGROVOC maintenance).

7.2 Assumptions and user study setup

First and foremost, this part of the proposal depends on the availability of methodological guidelines for the above-mentioned ontology development activities. Second, it also depends on the availability of a stable release of NeOn Toolkit supporting OWL as the main ontology representation formalism and the availability of the respective plug-ins supporting at least part of the reuse process. Third, this setup is dependent on the provision of detailed guidelines via the gOntt plug-in and the appropriate cheat sheets. Let us look at each of the categories in more detail.

Regarding the first aforementioned part, prescriptive methodological guidelines for reusing ontological resources, focused on general or common ontologies, domain ontologies as a whole, and ontology statements are provided in the NeOn methodology [22]. Additionally, prescriptive methodological guidelines for evolving ontologies are included in D5.4.2 [23].

Second, with the release of NeOn Toolkit v1.2, there is a full editing support for OWL available. One can configure the toolkit to start a new OWL project (in addition, one can also setup a FLogic project). OWL editor and other utilities (such as basic visualizations) are also in a stable state. In terms of plug-ins needed for this experiment, we expect the following to be involved:

- ***Change Capturing*** … [available in plug-in store] the purpose of this plug-in is to capture ontology changes from the NeOn Toolkit and log them into Oyster distributed registry. This plug-in allows users to visualize the history of ontology changes, is in charge of applying changes received from other clients to the same ontology, and supports the communication with Oyster to carry out the synchronization.
- ***Workflow Support*** … [available in plug-in store, 80% finished] the purpose of the plug-in is to track the ontology changes and then manipulate these changes according to the role of a user, where the user could log in / out or register in a preference page. Specifically, the roles of a Viewer, Subject Expert and Validator are supported.
- ***Collaboration Server*** … [only available temporarily for v1.2 and only internally to the partners] this is a proprietary component pluggable to the NeOn infrastructure and NeOn Toolkit v1.2 supporting the management of ontology changes, and the development and maintenance of ontologies in a collaborative scenario.
- ***Evolva*** … [available for download as early prototype] an ontology evolution tool that handles information discovery from external data sources (e.g., text documents), data validation, ontology changes, evolution validation and evolution management. One of the advantages of Evolva is that it relies on multiple background knowledge sources (including online ontologies and lexical databases) for knowledge integration.
- ***Watson*** [available in plug-in store] … this supports knowledge reuse and allows the ontology developer to query Watson to find existing descriptions of selected entities in the edited ontology, and to reuse (i.e., integrate) these descriptions in the edited ontology.
- ***Oyster API/GUI*** [available in plug-in store] … this plug-in provides services for storage, cataloguing, discovery, management, and retrieval of ontologies (and related entities) from a network of peers based on the ontology metadata definitions.
- ***OWLDiff*** [available from its developer, third-party] … this supports ontology comparison and merging; OWLdiff serves the same purpose for ontologies as *diff* does for textual files.

In addition to the core plug-ins supporting ontology resource reuse and integration, other auxiliary plug-ins may be involved; e.g., the plug-in for visualizing summaries of the discovered ontologies and visualizing the developed ontological relationships, possibly SPARQL for ontology querying and OWL documentation plug-in may be included to improve the ontology quality standard and the its 'chances' to be reused. Overall, all core plug-ins are available and most of the other mentioned plug-ins in auxiliary roles are also available, at least as advanced working prototypes. Obviously, there is likely to be further development in all the plug-ins, most notably with Watson and ontology summary visualization, which are planned to be integrated and cross-linked to better support previews for the discovered ontologies. Nonetheless, this makes the setup for this part of the experiment proposal aligned with the technological developments and implementation, and thus the study as motivated in this chapter is realistic to be accomplished during M37-M48.

The third assumption for successfully running this part of the experiment is the availability of the gOntt plug-in with respective cheat sheets. As mentioned in section 2.3, this work is at the time of writing this deliverable in progress, thus, the experiment needs to be planned after the cheat sheets and gOntt reach the stable release.

7.3 Specifics of the experiment design

In this experiment we provide participants with a set of a few informal paragraphs about the domain to be modelled. Such paragraphs will include issues related to ontological resource reuse

and ontology evolution to force participants to plan the reuse process and the evolution activity, and thus to use Watson and Change Capturing plug-ins.

Using the set of paragraphs provided by us, participants would be expected to:

1. Write the ORSD using the methodological guidelines for the ontology (requirement) specification activity. Such guidelines will be used by one group directly from D5.4.1 [22], and by the other group directly by using the methodological help provided by gOntt.

   It is important to mention that at this moment there is no NTK plug-in associated with the ontology (requirement) specification activity. Thus, the idea behind this experiment is to evaluate how people follow methodological guidelines that have no associated plug-in, and how people work with the NeOn Toolkit in which some activities resulting from the ORSD should then be carried out manually (or with other external tools). With this experiment we can also learn if a plug-in for the specification activity is needed or recommended.

2. Using the ORSD they wrote, participants should create the scheduling for their ontology project, depending on the group based on the methodological guidelines in documents or based on the guidelines included in gOntt plug-in.

3. Develop the ontology by means of reusing existing ontological resources (this partly overlaps with the experiment described in the previous chapter, but as the reader can see, here the reuse only forms a small part of the ontology development 'value chain'.

4. Evolve the developed ontology and formally capture the evolutionary changes to the ontology so that further reuse is encouraged and facilitated.

7.4 Experiment analysis and discussion

In terms of evaluating the results of this experiment, one can again use different measures. As in section 4.4, we can measure the overall quality and coverage of the resulting ontologies. As before this would require an existence of a 'gold standard'; i.e., the ideal outcome. Within this study, the quality evaluation would be primarily on the functional and usability level (see [16]), although some of the methods used additionally cover the structural level to some extent:

- *Coverage of the problem set out by each ORSD* (functional measure). This measure assumes that we can identify a minimal set of concepts and properties the resulting ontology should contain, and look for those to calculate the degree of "coverage"; i.e., to what extent does the ontology produced in the experiment cover the minimal set of expected terms and their relationships. Such set of terms to be covered by the ontology is already included in the ORSD.

- *Usability of the outcome* (usability measure). The integrated ontologies can be analyzed regarding their clarity, whether the concepts are formally defined and axioms included, whether the local names are clear and unambiguous as a result of including the statements and/or ontology components found on the (Semantic) Web, and whether the participants followed some naming convention, including the ontology containing labels and comments.

- *Modelling issues/unresolved problems* (structural and functional measure). Even an ontology that exactly covers the competency questions included in each ORSD might have some inherent problems that would for example hamper its usage in a system or make it less flexible and not so easy to extend or update. Such issues are not easily discovered, but are often closely related to slightly strange modelling choices. These problems can be on both the structural and functional level.

- *Use or absence of ontology design patterns* (structural measure). In this case, one can also observe whether or not some ontology design patterns were applied in the development of the ontological resources.

Specific hypotheses we can pursue with these measures may include the following:

- H3.1: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling mistakes) for ontologies developed by reusing existing ontological resources will be higher in comparison with ad-hoc development of the ontology from scratch.
- H3.2: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling mistakes) for ontologies will be higher if participants follow the in-built step-by-step guidance implementing the NeOn Methodology in the NeOn Toolkit, when compared with the users who do not have such a tight methodological guidance.
- H3.3: The reuse ratio (measure of how many time a chunk of a particular ontology is found in another ontology by means of a module or import) will increase for languages other than English compared to the current state when no explicit localization is supported in the ontology engineering toolkits.
- H3.4: The use of built-in methodological guidance will lead to a more structured and transparent representation of differences between concepts, terms, labels, linguistic alternatives, etc.

8. Supporting ontology discovery, integration and modularization

To some extent, this part of the proposal extends and complements the part described in the previous chapter. While the concern in chapter 4 was on reusing non-ontological resources, the same motivation applies to the reuse, reengineering and integration of existing ontologies. Indeed, according to both sets of user requirements ([9, 11]), there is an explicit request to support situations whereby new ontologies are developed on the basis of existing ones, either by transforming the conceptual model of an existing and implemented ontology into a new ontology (reengineering) or by including the existing ontology into the newly developed one (integration). The ontology engineer shall be able to open and visualize any ontology and use it as a basis to create a new one. This implies that the engineer be able to select and copy any ontology element or a part of an existing ontology, and paste it into the newly developed (edited) ontology.

Especially in case of large ontologies, it may be impractical to reuse the entire ontologies, i.e., all its axioms – whether because of their low relevance to the modelled domain or because of their inconsistency with already made modelling choices. It is therefore important to support users in defining and selecting “fragments” of ontologies, be it self-contained axioms or modules covering a subset of axioms. Common reasons for these facilities are: to improve efficiency (by selecting only the part of the ontology most frequently used), sharing editorial duties, visualization purposes and user rights. Modules may be manually created by ontology engineers and editors/validators, or created by means of (semi)automatic methods (e.g., entire branches of a hierarchy, subclass skeleton). More discussion of the mentioned user needs can be found e.g., in requirements 2.4.1 3 and 2.4.4 4a, and their respective rationale, in [11].

Ontology reuse and reengineering is tightly coupled with the capability to discover the entire ontologies, partial modules or the individual axioms satisfying some given user need. Since ontology development is to a great extent about formalizing a particular domain, one can formulate the ontology discovery as a bridge between an informal problem specification (often by means of key words) and a formal solution to the problem by means of a representation that can be directly integrated into the developed ontology. As suggested in [6], the ontology discovery activity may be enriched and supported by various ontology quality assessment techniques, but also social ranking and reviewing functionalities, which help to make the choice what to integrate easier, more efficient, and generally less expensive for the end user.

8.1. Overview and objectives

Formally, as defined in [21] the problems motivating this chapter can be assigned to several different ontology development activities. Our methodology and the glossary define ontology resource reuse as using available ontological resources (ontologies, modules, statements, or ontology design patterns) in the solution of different problems. We distinguish several specialized activities in this larger process, e.g., reusing ontology statements (axioms), reusing ontology modules (self-contained subsets of axioms), reusing full ontologies, and reusing ontology design patterns (i.e., generalized solutions to some common modelling problems). In addition to these reuse-centric activities, there are several distinguishable support activities relevant to the needs captured in this part of the proposal, e.g., searching for ontologies, comparing found ontologies, selecting ontologies for further reuse, and obviously ontology (or component) integration.

Furthermore, the problem of ontology selection may be seen as a wrapper for several finer-grained activities, including ontology reverse engineering (for establishing the purpose of the original ontology, module or axiom), and ontology forward engineering (for deciding on how the discovered resource may be exploited in the new ontology). One can also formally distinguish mere ontology restructuring (where verbatim components are reused, only in a different place) from a more

generic ontology integration (where axioms may acquire different roles and purposes than originally modelled in the reused source). An example of the former activity is the inclusion of axiom setting the newly defined “Researcher” concept as a specialization of the reused “Employee” role. An example of the latter might be the refinement of a restriction on what makes a “Collective” by means of referring to entities like “Person” and “participatesIn” reused from a third-party model.

Hence, as can be seen from this sketch, the set of activities mentioned (and illustrated in) is sufficiently rich and reasonably self-contained that is likely to be carried out by one type of user – in this case, an ontology repository manager or an ontology editor (including domain experts).

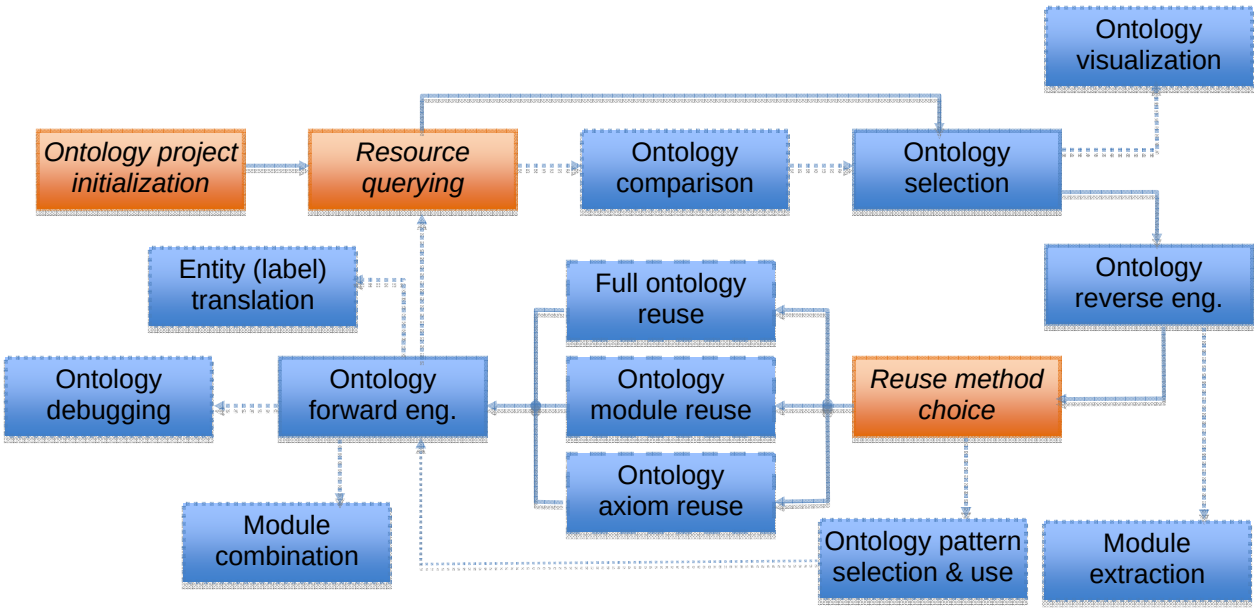

Figure 6. A process of ontology development steps[5] relevant to reusing ontologies and including ontology development activities (in blue) and auxiliary activities (in orange).

An ontology developer targeted in this part of the evaluation proposal needs to be aware of how his or her needs may be expressed to query an appropriate repository – this could be an ontology search engine (such as Watson), p2p infrastructure (such as Oyster), a combination of both (as e.g., realized in the Cupboard framework), or, in principle, other ontology sources. Query formulation is one area where methodological support makes a difference. Once the initial need is formulated as a keyword query, the user needs methodological support to select the right ontology; here auxiliary activities like ontology comparison or visualization may be helpful.

The next activity where methodological input is likely to be critical is supporting the choice of what to reuse from an existing ontological resource, and how to realize the restructuring or integration in a particular case. Once decided on the reuse style, this needs to be correctly engineered into a new ontology, and here methodology is likely to provide useful guidance. Finally, as in the previous chapter, additional auxiliary activities may be invoked, including debugging and consistency check for the newly developed ontology network, ontology querying, visualization, publication to the online repositories, etc.

[5] Note that the steps shown in the figure in orange are not included in the NeOn Glossary. Informal choice boxes (orange) depict the point where one or another ontology development activity (blue) needs to be selected.

8.2. Assumptions and user study setup

First and foremost, this part of the proposal depends on the availability of methodological guidelines for the above-mentioned ontology development activities. Second, the proposal depends on the availability of a stable release of NeOn Toolkit supporting OWL as the main ontology representation formalism and the availability of the respective plug-ins realizing at least some of the ontology resource integration/reuse techniques. Third, this setup is dependent on the provision of detailed guidelines via the gOntt plug-in and the appropriate cheat sheets. Let us look at each of the categories in more detail.

Methodological support for formulating queries towards ontology repositories is fully covered in formal terms, however, the intention was to draw upon the familiar metaphor of web search (Google, Yahoo!), which suggests little extra overheads on this front to be supported. Initial methodological support for choosing the concrete reuse activity has been formulated in the first draft of the NeOn Methodology [22]. Furthermore, existing plug-in for Watson removes the need to formulate any keyword query, as it supports the interaction with Watson on the basis of menus and contextual (right) clicks.

Module manipulation support is present in the updated NeOn Methodology. The guidelines for carrying out module extraction, and subsequently for applying various module operations in the NeOn Toolkit on the (discovered and reused) modules is now included in [23]. Existing guidelines may need some adjustment to take in account more recent features implemented for the ontology discovery activity – including, the reliance on social ranking and rating of reusable ontologies and ontology components, the use ontology quality criteria and measures in selecting resources most likely to be reused, and also the inclusion of publishing, sharing mechanisms that would feed the result of the process involving discovery → selection → integration back to the communal repositories.

In addition to these, another activity may need to be considered; namely, how to feed back to the community the rationale for reusing or not reusing a particular resource. In other words, some extensions to the methodological support encouraging the users to rate and review the reused ontologies may be needed. However, this does not seem to be a critical issue at the moment, as this kind of input is to be provided as a part of Cupboard development and deployment in the final year of the project.

Second, with the release of NeOn Toolkit v1.2, there is a full editing support for OWL available. One can configure the toolkit to start a new OWL project (in addition, one can also setup a FLogic project). OWL editor and other utilities (such as basic visualizations) are also in a stable state. In terms of plug-ins needed for this experiment, we expect the following to be involved:

- ***Watson*** [available in plug-in store] … this supports knowledge reuse and allows the ontology developer to query Watson to find existing descriptions of selected entities in the edited ontology, and to reuse (i.e., integrate) these descriptions in the edited ontology.
- ***Oyster API/GUI*** [available in plug-in store] … this plug-in provides services for storage, cataloguing, discovery, management, and retrieval of ontologies (and related entities) from a network of peers based on the ontology metadata definitions.
- ***Modules*** [available in plug-in store] … the plug-in provides the functionality to extract modules and the operators to manipulate and combine modules; in particular, it supports export/import features, and the calculations of unions, intersections and alignments among modules.
- ***OWLDiff*** [available from its developer, third-party] … this supports ontology comparison and merging; OWLdiff serves the same purpose for ontologies as *diff* does for textual files.
- ***RaDON*** [available in plug-in store] … this plug-in helps the user to deal with inconsistence and incoherence occurring in networked ontologies. It is capable of diagnosing (spotting)

the source(s) of inconsistence and incoherence in single or networked ontologies, as well as repairing the identified issue either fully automatically or manually by the user.

- ***OntoSumViz*** [available as prototype download] … this plug-in assists with the summarization of an ontology into a set of concepts with the highest information value ('key concepts') and a topological overview of the ontology structure, and enables navigation in the visualized summaries in a 'mid-out' manner (as opposed to the usual 'top-down' style).
- ***LabelTranslator*** [available in plug-in store] … this plug-in helps the user to obtain the most probable translation for each ontology label, to capture the linguistic information associated with concepts using a linguistic model repository (LIR), and to maintain the ontology and linguistic information synchronized.

In addition to the core plug-ins supporting ontology resource reuse and integration, other auxiliary plug-ins may be involved; e.g., the plug-in supporting initial ontology schema modelling, the plug-in for visualizing resulting ontology relationships and/or summaries, possibly SPARQL for ontology querying. Also, OWL documentation plug-in may be included to improve the ontology quality standard. Overall, all core plug-ins are available and most of the other mentioned plug-ins in auxiliary roles are also available, at least as working prototypes. This makes the setup for this part of the experiment proposal aligned with the technological developments and implementation, and thus the study as motivated in this chapter is realistic to be accomplished during M37-M48.

The third assumption for successfully running this part of the experiment is the availability of the gOntt plug-in with respective cheat sheets. As mentioned in section 2.3, this work is at the time of writing this deliverable in progress, thus, the experiment needs to be planned after the cheat sheets and gOntt reach the stable release.

8.3. Specifics of the experiment design

This experiment is following a screenplay that has been tested several times in NeOn, e.g., in the user studies carried out in WP4 [3, 4, 5]. In each case, a set of existing ontologies was identified, as available on the Web, and a sequence of tasks with a growing degree of complexity was given to the user. Each task can then be concerned with a particular type of integration (e.g., axiom restructuring vs. module import), and one can easily observe whether or not the participant follows the methodological guidance or not.

In order to bootstrap this experiment, the main challenge is in identifying an appropriate sub-set of ontologies in the public domain that would include ontologies of different quality, size, and coverage. Yet, this set needs to be 'controlled', in the sense of being comprehensible to the study participants, being sufficiently informative to allow someone who is not subject expert in a given field to discover, reverse engineer and integrate chunks of ontologies provided in the set.

If support for ontology selection is to include the analysis of ontology ratings and/or of ontology quality measures, these need to be provided prior to the experiment itself. The precise set of ontologies to be used cannot be specified at the moment of writing, as some open questions exist, e.g., with respect to domain focus, target audience subject expertise, and the ease of obtaining auxiliary information (such as ontology quality, rating and/or reviews).

8.4. Experiment analysis and discussion

In terms of evaluating the results of ontology resource reuse and integration, one can again use different measures. As in section 4.4, we can measure the *overall quality and coverage* of the resulting ontologies. As before this would require an existence of a 'gold standard'; i.e., the ideal outcome of the reengineering and integration process. Within this study, the quality evaluation

would be primarily on the functional and usability level (see [16]), although some of the methods used additionally cover the structural level to some extent:

- Coverage of the problem set out by ORSD (functional measure)
- Usability of the outcome (usability measure)
- Modelling issues/unresolved problems (structural and functional measure)
- Use or absence of ontology design patterns

This measure assumes that we can identify a minimal set of concepts and properties the resulting ontology should contain, and look for those to calculate the degree of "coverage"; i.e., to what extent does the ontology produced in the experiment cover the minimal set of expected terms and their relationships. With respect to usability, the integrated ontologies can be analyzed regarding their clarity, whether the concepts are formally defined and axioms included, whether the local names are clear and unambiguous as a result of including the statements and/or ontology components found on the (Semantic) Web, and whether the participants followed some naming convention, including the ontology containing labels and comments. The third aspect is concerned with problems in the ontologies. Even an ontology that exactly covers the competency questions and the ORSD might have some inherent problems that would for example hamper its usage in a system or make it less flexible and not so easy to extend or update. Such issues are not easily discovered, but are often closely related to slightly strange modelling choices. These problems can be on both the structural and functional level. One can also observe whether or not some ontology design patterns were applied in the integration or restructuring of the ontological resources.

Specific hypotheses we can pursue with these measures may include the following:

- H4.1: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling mistakes) for ontologies involving reused, integrated and restructured resources will be higher in comparison with ad-hoc development of the ontology from scratch.
- H4.2: The overall quality measure (comprising ontology coverage, usability and lack of occurrence of modelling problems) for ontologies involving reused, integrated and restructured resources will be higher if participants follow the in-built step-by-step guidance implementing the NeOn Methodology in the NeOn Toolkit, when compared with the users who do not have such a tight methodological guidance coupled with the toolkit.
- H4.3: Selection of the ontological resources to be reused will be faster and better justified (explained) for the user group using built-in methodological guidelines compared to the test group without such methodological guidance.
- H4.4: Selection of the ontological resources to be reused will be faster and better justified (explained) for the user group using built-in methodological guidelines compared to the test group that receives no tuition of the NeOn Methodology.
- H4.5: The use of built-in guidance for executing NeOn Methodology will lead to more detailed documentation of modelling decisions compared to test groups (without the built-in support and without any tuition of the NeOn Methodology).

9. Supporting ontology contextualization

The primary motivation for looking into experiments in ontology contextualization is the management of conceptual and structural heterogeneity in a corpus comprising a number of ontologies coming from different sources and being designed following different methodologies. Since ontologies, in principle, formally represent a certain viewpoint of a certain group of users pertaining to a certain part of the world, it may not always be appropriate to ask which of the two ontologies covering, say, agricultural thesaurus (i.e., AGROVOC vs. EUROVOC) is 'better'. More common question among professionals is how the two ontologies relate and whether there are any opportunities to interpret their alignment, mapping, or any other form of contextualization.

Different styles of ontology contextualization may be proposed. One of those better elaborated and evaluated sees contextualization as ontology matching and alignment. This view aims to find correspondences between semantically related entities of the aligned ontologies. Once found, these correspondences can be used for various tasks, such as ontology merging, query answering, data translation, or for the navigation on the semantic web. Thus, contextualizing ontologies in the sense of matching and aligning them enables the knowledge and data expressed in the respective ontologies to interoperate. A full-scale ontology matching initiative[6] has emerged over the years leading to annual evaluation exercises of the proposed competitive techniques.

A slightly different take-on of the ontology contextualization has been pursued from user's point of view. Here, some of the applicable techniques include various forms of ontology customization, such as hiding or extraction of a sub-set of concepts satisfying a given condition/query [1]. Along similar lines, much effort has been put into a range of ontology partitioning activities, all broadly inspired by the same objective – giving the user an opportunity to adapt a large and multi-topical ontology to their narrower needs and requirements. Many existing techniques are reviewed e.g., in [17]. Similarly as with ontology matching, there are ongoing initiatives in evaluating ontology partitioning and customization techniques.

Thus, ontology contextualization is to a great extent a 'backend' activity, i.e., there is a growing motivation to produce techniques capable of learning and requiring little or only very simple involvement of the user for (i) bootstrapping or (ii) correctness checking. Increasingly, the results of applying a certain contextualization technique to the ontology are stored persistently for future reuse or reference. For example, the Alignment Server[7] is one such framework that is capable of automatically executing several techniques and storing results for further analysis and retrieval by means of a dedicated API residing on top a relational database.

Hence, due to the nature of ontology contextualization efforts and the push towards more automated techniques, much of the methodological guidelines for ontology contextualization relates to the advice regarding how to adjust the results of an automated technique by setting its thresholds and other parameters to obtain a contextual correspondence that is deemed sufficient by the user for their specific task or problem.

9.1. Overview and objectives

Since the area of ontology contextualization is in flux and is one of the hot research topics, it is not surprising that detailed guidelines for carrying out such contextualization activities as ontology mapping, ontology matching and alignment, ontology partitioning, or ontology customization are

[6] For further details and a number of publications visit http://www.ontologymatching.org, the ontology matching portal partially supported by the recent KnowledgeWeb project (http://knowledgeweb.semanticweb.org).

[7] The actual server and API for connecting to it are available from http://alignapi.gforge.inria.fr.

not fully developed and formally captured yet. As we mentioned above, where such guidelines are developed, they usually refer to processing and interpreting the outcomes of applying a certain ontology contextualization technique. The lack of guidelines for supporting ontology contextualization throughout the ontology lifecycle makes it difficult to propose one specific experiment that would be more methodology-centric rather than technique-centric. There is little need to replicate the successful evaluation initiatives e.g., in ontology matching, at the level of tools and techniques…

However, the issues related to ontology customization are too important to be ignored in our effort to assess the degree to which NeOn Methodology as a whole is helpful and the degree to which it is supported by the ontology engineering toolkit. Thus, rather than testing and evaluating different tools and techniques, we intend to evaluate how methodological guidance may help the users in (i) setting up and executing an appropriate contextualization technique, and (ii) post-processing and interpreting the outcomes of such an activity.

At the former level, it makes sense to assess the level of methodological support, e.g., in a tool like gOntt, for directing the user towards the right set of ontology development activities by helping the user to distinguish different forms of contextualization (say, customizing ontology vs. aligning it with another ontology). At the latter level, we can observe how supportive are the existing plug-ins in terms of guiding the user through a series of choices, query reformulations, parameter adjustments and similarly, once the outcomes of the contextualization are available.

9.2. Assumptions and user study setup

As mentioned earlier in section 9.1, this part of the experimentation critically depends on the availability of methodological guidelines for the different styles of ontology contextualization. Second, the proposal depends on the availability of a stable release of NeOn Toolkit supporting OWL as the main ontology representation formalism and the availability of the respective plug-ins realizing at least some of the ontology contextualization techniques. Third, this setup is dependent on the provision of detailed guidelines via the gOntt plug-in and the appropriate cheat sheets. Let us look at each of the categories in more detail.

As is shown below, the current situation with NeOn Toolkit plug-ins relevant to this part of the experiment is rather mixed. There are several plug-ins reaching maturity (e.g., the Alignment plug-in), but there are also advanced prototypes (e.g., OntoConto plug-in) and early prototypes (Ontology Customization plug-in).

The Alignment plug-in [13] allows ontology users to match existing ontologies, i.e., by computing alignments between them to retrieve existing alignments and store available alignments on the Alignment Server for future reuse and sharing. The NeOn Alignment plug-in embeds the Alignment API and can use the web service interface to connect to any alignment server. We are currently in the process of formulating methodological guidelines for using it and interpreting its outcomes. OntoConto [14] is a graphical frontend to the alignment plug-in/server that gives the user an opportunity to interact with the contextual mappings at a less technical level – that is, instead of working with OWL/XML constructs, the correspondences are visually depicted as links between two graphs side by side.

In terms of supporting ontology customization, early prototypes based on SAIQL, OWL query engine [1] are being realized at the time of writing in the form of plug-in. The approach proposed and piloted in D4.2.3 [2] is based on the idea of identifying repetitive patterns in the customization queries, transforming these patterns into ‘customization templates’ and bundling the process of filling in a template into a Customization Wizard GUI. The approach assumes step-by-step extension of the customization query by the user, who may refer to the standard topological relations among the concepts but also to various meta-data (e.g., language of labels, access right, provenance or trustworthiness assertion). While the plug-in is being implemented, it is premature at the time of writing to forecast the availability of the methodological guidelines for this particular

ontology development activity. Nevertheless, this is an interesting area to experiment with, especially, in the context of usability and user friendliness.

Second, with the release of NeOn Toolkit v1.2, there is a full editing support for OWL available. One can configure the toolkit to start a new OWL project (in addition, one can also setup a FLogic project). OWL editor and other utilities (such as basic visualizations) are also in a stable state. In terms of plug-ins needed for this experiment, we expect the following to be involved:

- ***Alignment plug-in*** [available for download as prototype] … the plug-in embeds INRIA's Alignment API and can use the web service interface to connect to any alignment server; it allows ontology users to match existing ontologies by computing alignments between them, retrieving existing alignments, and storing new alignments on the Alignment Server.
- ***OntoConto*** [available for download as prototype] … this plug-in enables contextualized visualization of one ontology against another ontology, thus delivering the notion of ontology network; it enables the user to browse through an ontology in the context of other, formally aligned and thus related networked ontologies.
- ***Ontology Customization*** [under development and testing] … the plug-in addresses the cognitive overload while working with large ontologies or ontology networks by enabling the user to 'shrink' the ontology so that it contains the 'right amount' of information. This is achieved by the user defining a view over the ontology, which only contains those ontology segments that are needed by the user (akin to database views).
- ***RaDON*** [available in plug-in store] … this plug-in helps the user to deal with inconsistence and incoherence occurring in networked ontologies. It is capable of diagnosing (spotting) the source(s) of inconsistence and incoherence in single or networked ontologies, as well as repairing the identified issue either fully automatically or manually by the user.
- ***OntoSumViz*** [available for download as prototype] … this plug-in assists with the summarization of an ontology into a set of concepts with the highest information value ('key concepts') and a topological overview of the ontology structure, and enables navigation in the visualized summaries in a 'mid-out' manner (as opposed to the usual 'top-down' style).

9.3. Specifics of experiment design

Since much in ontology contextualization depends on how the tools and techniques are used, this suggests that experiments relevant to this chapter are likely to take a standard usability testing direction. Informally, we may want to investigate how well does the tool perform in helping the user to translate their need into a formal artefact (e.g., alignment parameter or customization query). Furthermore, to what extent is the formal complexity hidden from the user by means of breaking the process into smaller decisions, guiding the user through the choices, making recommendations and similarly.

9.4. Experiment analysis and discussion

In terms of evaluating the results of ontology resource reuse and integration, one can again use different measures. As in the previous sections, we may consider measuring the *overall quality and coverage* of the resulting ontologies, but this may not be the most informative measure in this case. This is because the main principle of ontology contextualization is *approximation* and *estimation*. In other words, while the overall quality tends to be measuring how good the result is, in the case of ontology contextualization, it is more typical to consider whether or not the proposed mapping is sufficient, good enough or acceptable. While some gold standards; i.e., the ideal outcome of the process, exist for alignment, these are well evaluated (as described earlier) and there is low impact for us to achieve in repeating these studies. Hence, within this study, the quality evaluation would

be primarily on the usability level (see [16]), although some of the methods used additionally cover the structural level to some extent:

- Coverage of the problem set out by ORSD (functional measure)
- Usability of the procedure to reach a sufficient outcome (usability measure)

The coverage assumes that we can identify a minimal set of concepts and properties the resulting ontology should contain, and look for those to calculate the degree of "coverage"; i.e., to what extent does the ontology alignment or custom view produced in the experiment cover the minimal set of expected terms and their relationships. With respect to usability, both the results and the procedures can be analyzed regarding their clarity, whether the concepts are formally defined and axioms included, whether the local names are clear and unambiguous as a result of including the statements and/or ontology components found on the (Semantic) Web, and whether the participants followed some naming convention, including the ontology containing labels and comments. Generally, it is likely we will move towards this latter measure.

For this purpose we may drive the experiment along the lines of the Software Usability Measurement Inventory (SUMI) technique [25] that focuses on getting users' responses to a range of questions and statements, from which we would select the ones focusing on how helpful the guidance is, how comprehensible it is, how likely it is to lead to mistakes or backtracking, how consistent the guidance is, etc.

Some specific hypotheses we can pursue with these measures may include the following:

- H5.1: Positive attitudes in categories affect and helpfulness would be quantitatively greater for built-in support for methodological guidelines compared to similar attitudes observed for user who only receive verbal or no methodological tuition.
- H5.2: Users will feel more in control (more positive attitude to respective statements) with the built-in guidance if compared to verbal tuition of the methodology.
- H5.3: Users will perceive the task of ontology contextualization as easier to learn and master when using the built-in wizards compared to the verbal tuition of the methodology.

10. Conclusions and Wrap-Up

In this deliverable we presented a series of plans of the experiments to be performed at the level of NeOn methodology as a whole, as opposed to only testing one partial method or guideline. The proposals are clustered into sub-scenarios and cover the areas that were at the beginning of the project considered as critical to the success; that is to the realization of support for engineering and developing networked ontologies. The hallmarks of the process include the requirements to:

- support development of new ontologies by means of reusing existing resources, both ontological and non-ontological;
- support development of smaller ontological artefacts by means of extracting modules from a large ontology and combining modules into a large ontology;
- support development of ontologies in collaborative and multilingual setting;
- support development of ontologies with the knowledge artefacts being distributed over the web and/or peer-to-peer networks; and
- support contextualization of ontologies and the development of ontology networks in which a concrete contextual relationship (e.g., alignment) is the main driver of networking;

The experiments discussed in this report have been planned and defined to address the above requirements. They were deliberately described in a 'modular' fashion rather than one, large, monolithic experiment, as this description allowed us to go more in depth in justified and rationalizing the core principle of the respective experimental focus. The individual experimental strands are likely to be re-combined into coherent setups of specific experimental sessions, at the time of executing a particular session. This leaves us with sufficient flexibility with respect to various constraints a particular user group may have. For example, some of the studies may follow up on the two-day tutorial and training sessions, where time may be significantly shorter for the participants to complete a task, as compared to, say, a training course lasting several months. The implication is that studies accompanying the tutorials need to be shorter and smaller in scope, yet we need to take advantage of as many training sessions as possible to feed us some, even if partial, evaluation data for the NeOn Methodology.

Plans described here will be realized in year 2009 and their results will appear in the next version of this deliverable. Specific decisions what experiments may be carried out where and at what level of complexity remains to be decided at a later point. This is primarily to ensure that all relevant technological and methodological inputs are in place, are sufficiently robust and reasonable familiar to the facilitators of the experiments. Hence, to make a realistic estimate, we believe the first large-scale tests are likely to take place in July/August 2009 – that is, M41-42 of the project. The bulk of experimentation is likely to coincide with the new academic year – that is M44-M46 of the project.

Note that the purpose of this deliverable is to sketch how the NeOn Toolkit supports the methods and methodology as a whole defined in NeOn. This support is one of the key points in the adoption of both the NeOn Toolkit and the NeOn methodologies and methods, hence, the proposed studies and experiments are likely to provide further insight into viability and market competitiveness of the project outcomes. Although analyzing market impact is not the main task and the main driver for this experimental work, we will liaise with WP6 and WP9, and share our observations and findings with them, e.g., for the purposes of making market-related predictions.

11. References

[1] N. Bercovici, G. Groener, S. Schenk, A. Kubias, and M. Dzbor: *Ontology customization and module creation: Query-based customization operators and model*. Deliverable D4.2.2, NeOn Consortium (http://www.neon-project.org), March 2008.

[2] N. Bercovici, M. Dzbor, R. Melero, and J. Candini: *Ontology customization prototype presentation*. Deliverable D4.2.3, NeOn Consortium (http://www.neon-project.org), February 2009.

[3] M. Dzbor, E. Motta, C. Buil Aranda, J.M. Gomez-Perez, O. Goerlitz, and H. Lewen: *Developing ontologies in OWL: An observational study*. Workshop on OWL: Experiences and Directions, November 2006, Georgia, US. (http://owl-workshop.man.ac.uk/acceptedLong/submission_30.pdf)

[4] M. Dzbor, E. Motta, C. Buil Aranda, J.M. Gomez-Perez, O. Goerlitz, and H. Lewen: *Analysis of user needs, behaviours & requirements wrt interfaces and navigation of ontologies*. Deliverable report D4.1.1, NeOn Project Consortium (http://www.neon-project.org), August 2006.

[5] M. Dzbor, C. Buil Aranda, E. Motta, and J.M. Gomez-Perez: *Analysis of user needs, behaviours and requirements on ontology engineering tools*. Deliverable report D4.1.2, NeOn Project Consortium (http://www.neon-project.org), January 2008.

[6] M. Dzbor, M.C. Suárez-Figueroa, E. Blomqvist, H. Lewen, M. Espinoza, A. Gómez-Pérez, and R. Palma: *Experimentation and Evaluation of the NeOn Methodology*. NeOn Deliverable D5.6.2, NeOn Project (http://www.neon-project.org). February 2009.

[7] R. García-Castro, M. C. Suárez-Figueroa, M. Espinoza, M. Sini, H. Lewen and Eva Blomqvist. *Experimentation with the NeOn methodologies and methods*. NeOn Deliverable D5.6.1, NeOn Project (http://www.neon-project.org). March 2008.

[8] A. Gómez-Pérez, M. Fernández-López, O. Corcho. *Ontological Engineering*. November 2003. Springer Verlag. Advanced Information and Knowledge Processing series. ISBN 1-85233-551-3.

[9] J.M. Gómez-Perez, C. Daviaud, B. Morera, R.V. Benjamins, T. Pariente Lobo, G. Herrero Cárcel, and G. Tort: *Analysis of the pharma domain and requirements on the case studies*. NeOn deliverable D8.1.1, NeOn Consortium (http://www.neon-project.org), August 2006.

[10] M. Iglesias, C. Caracciolo, Y. Jaques, M. Sini, F. Calderini, J. Keizer, F. Le Hunte Ward, M. Nissim, and Aldo Gangemi: *User requirements*. NeOn deliverable 7.1.1, NeOn Project Consortium (http://www-neon-project.org), August 2006.

[11] M. Iglesias Sucasas, C. Caracciolo, C. Baldassarre, and Y. Jaques: *Revised specifications of user requirements for the Fisheries case study*. NeOn deliverable 7.1.2, NeOn Project Consortium (http://www.neon-project.org), February 2008.

[12] D.L. Kirkpatrick: *Evaluating Training Programs: the Four Levels*. 1994, San Francisco: Berrett-Koehler Publishers. p.289.

[13] Ch. Le Duc, M. d'Aquin, J. Barrasa, J. David, J. Euzenat, R. Palma, R. Plaza, M. Sabou, B. Villazón-Terrazas: *Matching ontologies for context: The NeOn Alignment plug-in*. NeOn deliverable D3.3.2, NeOn Project Consortium (http://www.neon-project.org), February 2008.

[14] B. Pajntar and D. Mladenic: *NeOn Toolkit plugin realizing revised and extended techniques for contextualized visualization of ontologies and ontology networks*. Deliverable report D4.5.2, NeOn Project Consortium, November 2008.

[15] H. S. Pinto, C. Tempich, S. Staab. *DILIGENT: Towards a fine-grained methodology for DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies*. In Ramón López de Mantaras and Lorenza Saitta, Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004), August 22nd - 27th, pp. 393--397. IOS Press, Valencia, Spain, August 2004. ISBN: 1-58603-452-9. ISSN: 0922-6389.

[16] R. M. Sabou, S. Angeletou, M. d'Aquin, J. Barrasa, K. Dellschaft, A. Gangemi, J. Lehmann, H. Lewen, D. Maynard, D. Mladenic, and M. Nissim. *Methods for Selection and Integration of Reusable*

Components from Formal or Informal User Specifications. NeOn Deliverable D2.2.1, NeOn Project (http://www.neon-project.org). May 2007.

[17] A. Schlicht and H. Stuckenschmidt: *Criteria-Based Partitioning of Large Ontologies*. Proc. of the Knowledge Capture (K-Cap) Conference, British Columbia, Canada, 2007.

[18] S. Staab, H.P. Schnurr, R. Studer, Y. Sure. *Knowledge Processes and Ontologies*. IEEE Intelligent Systems 16(1):26–34. (2001).

[19] M. C. Suárez-Figueroa, S. Brockmans, A. Gangemi, A. Gómez-Pérez, J. Lehmann, and H. Lewen. *NeOn Modelling Components*. NeOn Deliverable D5.1.1, NeOn Project (http://www.neon-project.org). April 2007.

[20] M. C. Suárez-Figueroa, G. Aguado de Cea, C. Buil, C. Caracciolo, M. Dzbor, A. Gómez-Pérez, G. Herrero, H. Lewen, E. Montiel-Ponsoda, V. Presutti. *NeOn Development Process and Ontology Life Cycle*. NeOn Deliverable D5.3.1, NeOn Project (http://www.neon-project.org). August 2007.

[21] M. C. Suárez-Figueroa, M. Fernández-López, A. Gómez-Pérez, K. Dellschaft, H. Lewen, M. Dzbor. *Revision and Extension of the NeOn Development Process and Ontology Life Cycle*. NeOn Deliverable D5.3.2, NeOn Project (http://www.neon-project.org). November 2008.

[22] M. C. Suárez-Figueroa, G. Aguado de Cea, C. Buil, K. Dellschaft, M. Fernández-López, A. Garcia, A. Gómez-Pérez, G. Herrero, E. Montiel-Ponsoda, R. M. Sabou, B. Villazon-Terrazas, and Z. Yufei. *NeOn Methodology for Building Contextualized Ontology Networks*. NeOn Deliverable D5.4.1, NeOn Project (http://www.neon-project.org). February 2008.

[23] M. C. Suárez-Figueroa *et al*. *Revision and Extension of the NeOn Methodology for Building Contextualized Ontology Networks*. NeOn Deliverable D5.4.2, NeOn Project (http://www.neon-project.org). February 2009.

[24] P. Tiedt: *Building cheat sheets in Eclipse v3.2*. IBM developerWorks, 8 Aug 2006 (see http://www.ibm.com/developerworks/opensource/library/os-ecl-cheatsheets).

[25] E.P.W.M van Veenendaal. *Questionnaire based usability testing*. In Proc. of the European Software Quality Week, Brussels, 1998.