NeOn-project.org

NeOn: Lifecycle Support for Networked Ontologies

Integrated Project (IST-2005-027595)

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

D7.1.2 Revised specifications of user requirements for the Fisheries case study

Deliverable Co-ordinators: Marta Iglesias Sucasas, Caterina Caracciolo, Claudio Baldassarre, Yves Jaques (FAO)

Deliverable Co-ordinating Institution: Food and Agriculture Organization of the United Nations (FAO)

Document Identifier:	NEON/2008/D7.1.2/1.0	Date due:	December 31, 2007
Class Deliverable:	NEON EU-IST-2005-027595	Submission date:	February 15, 2008
Project start date:	March 1, 2006	Version:	1.0
Project duration:	4 years	State:	Final
		Distribution:	Public

NeOn Consortium

This document is part of a research project funded by the IST Programme of the Commission of the European Community, 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 Sucasas 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

This deliverable has been prepared by FAO.

Change Log

Version	Date	Amended by	Changes
0.1	29-11-2007	Caterina Caracciolo	First draft revised reqs. for ontology lifecycle (Chapter 2)
0.2	04-12-2007	Yves Jaques	Revised Requirement for FSDAS (Chapter 3)
0.3	05-12-2007	Claudio Baldassarre	Filled Template for Life Cycle Requirements (Chapter 2)
0.4	06-12-2007	Caterina Caracciolo	Revised Chapter 2
0.5	12-12-2007	Marta Iglesias Caterina Caracciolo	Introduction and background info (Chapter 1) Revised req.s for lifecycle (Chapter 2)
0.6	13-12-2007	Yves Jaques	Revision of Chapter 3 and annex B
0.7	17-12-2007	Claudio Baldassarre	Review of the technical content according requirements review (Chapter 2)
0..8	18-12-2007	Marta Iglesias	Executive Summary, Conclusions, and general review
0.9	18-12-2007	Yves Jaques	Number FSDAS functional requirements, Add a few missing FSDAS use cases. Add FSDAS requirements summary to annex
0.9a	17-01-2008	Claudio Baldassarre Caterina Caracciolo, Yves Jaques	General revision according to QA comments
0.9b	23-01-2008	Andrew Bagdanov Caterina Caracciolo	Proofreading of the entire document Stylistic and layout revisions
1.0	04-02-2008	Hannah Beaman	Final QA

Executive Summary

This document revises and updates the user requirements for the Fisheries case study (WP7) originally prepared in D7.1.1 (in M7).

As highlighted in the introduction (Chapter 1), the purpose of this deliverable, 22 months after the launch of the NeOn project, is to refine the user requirements for the WP7 case study. This refinement of the user requirements is also meant to enhance the alignment of the case study with the NeOn project and better contribute to it as a whole.

The user requirements for the Fisheries ontologies lifecycle management system are presented in Chapter 2. This chapter also includes an updated description of user's roles and requirements for ontology engineers and ontology editors.

The requirements for the FSDAS are revisited in Chapter 3. That chapter covers the general characteristics of the FSDAS (i.e., its scope, perspective, features), the refinement and extension of the functional requirements, its foreseen user types, and a number of user interface issues and non-functional requirements.

Conclusions are presented in Chapter 4.

This deliverable also includes five annexes with additional detailed information. Annex A provides a summary of the requirements presented in Chapter 2, together with their mapping to the NeOn architecture. Annex B traces all changes to requirements, scenarios and uses cases from D7.1.1 until now. Annex C consists of tables that are meant to help the reader trace the changes (modifications/refinements) with respect to the requirements presented in D7.1.1 and D7.5.1. Finally, Annex D and Annex E list the revised use cases and the new use cases for FSDAS respectively.

Table of Contents

List of tables

List of figures

1 Introduction

The WP7 case study deals with the creation of an ontology-driven Fisheries Stock Depletion Assessment System (FSDAS). The system will use FAO and non-FAO resources on Fisheries to assist in the assessment of fish stock and will be empowered by means of a network of ontologies.

During the last 18 months of the project, various deliverables have been produced within WP7: from the initial user requirements produced at an early stage of the project (D7.1.1, M6); to the release of a first set of medium-to-large set of Fisheries ontologies (D7.2.2, M18); and the design of the software architectures for the systems needed to successfully achieve the objectives of the case study (D7.4.1 and D7.5.1, M18).

At the same time, the various NeOn work packages have been advancing in the research and development areas (WP1-WP4), as well as in the integrative areas (WP5-WP6). Constant feedback from WP7 to the other work packages (and vice versa) has helped refine the case study requirements and clarify a number of related issues that appeared early on in the project. These issues could not be addressed at the time of D7.1.1 given the early stage of the project.

The purpose of the present deliverable, 22 months after the beginning of the project, is to wrap-up, refine and clarify the case study requirements, and to enhance their alignment with the NeOn project as a whole.

1.1 WP7 deliverables influencing D7.1.2

Deliverable D7.1.1 [D7.1.1] presented the initial (pre-NeON) requirements and use cases for the WP7 case study, introducing the Fisheries and fish stock management information and knowledge domain; user and roles; the requirements for managing the whole lifecycle of the Fisheries ontologies; and the requirements for the FSDAS. The ontologies developed, tested and deployed using the Fisheries ontologies lifecycle management system will underpin the FSDAS, together with a number of other electronic resources described in D7.2.1 [D7.2.1].

An initial set of Fisheries ontologies was conceptualized, populated and published on the Internet at http://www.fao.org/aims/neon.jsp. The work leading to the creation of these ontologies, together with the description of the ontology models were presented in D7.2.2 [D7.2.2]. These ontologies were mainly derived from one of the key systems mentioned in D7.2.1, the Fisheries Reference Table Management System (RTMS). This activity helped learn some lessons (cf. Chapter 7 and 8, [D7.2.2]) about the creation and population of medium-to-large ontologies from relational databases. It contributed to sharpening the requirements given in D7.1.1 regarding ontology creation (by ontology engineers) and support for ontology population.

Deliverables D7.4.1 [D7.4.1] and D7.5.1 [D7.5.1] were released at the same time as D7.2.2. They address the design of the software architecture for the system to support the fisheries ontologies lifecycle and for the FSDAS respectively.

While carrying out the work that lead to D7.4.1, several requirements presented in D7.1.1 had to be refined and some additional requirements (not explicitly mentioned in D7.1.1) were introduced.

It was realized that most of the Fisheries ontologies required for the case study would not be developed from scratch, but rather migrated from legacy systems. Therefore the lifecycle management system needs to provide adequate support to migrate and/or map data from relational databases or XML schemas into ontologies.

Besides the requirements for ontology design, population and validation, common to ontology engineering environments, the WP7 case study needed to look for a more articulated approach paying special attention to the editorial workflow, key to ensure that users could modify and update ontologies in a controlled, supervised and coherent manner. This is especially important for those ontologies already published and available on the Internet, since it is necessary to provide appropriate support to change management and version control.

As of D7.5.1, although the scope of the FSDAS has remained the same, it also included some refinements with respect to the user requirements included in D7.1.1. Also some non-functional requirements were changed. The most important change is the fact that in D7.1.1 the FSDAS was intended to be developed with browser based technologies, while after reviewing different options, it was found that Java Web Start deployment would better suit the software performance requirements and maintain programming language homogeneity with other partners. As a consequence, the NeOn components developed as Eclipse plugins are not available for complete reuse anymore, unless they also expose their functionalities as web services, invokeable from the FSDAS application environment.

The same way that the WP7 requirements have been better defined and sharpened during this period, all the other technical and integrative work packages have also evolved. WP7 being a case study, it depends on all the other work packages and on the direction taken by NeOn as a whole. Therefore this deliverable also serves to re-align the case study requirements with the general direction taken by the project.

1.2 WP7 major objectives revisited

Although the ultimate goal of the case study remains the same, some of the major objectives have been retrenched. In particular, some of the success indicators have been eliminated, due mainly to over-ambitious expectations by FAO on the project.

The goal of WP7 is to enhance accessibility to Fisheries stocks knowledge, allowing policy makers, at national and international level, to make informed decisions.

In this line the major objectives of the case study remain:

1.2.1 Objective 1: Create and maintain networked ontologies in the fisheries knowledge community

This implies putting mechanisms in place which will allow all actors involved in the lifecycle to create and maintain distributed fisheries ontologies (and mappings) and support their continuous growth; and in particular:

- Users are able to implement the fisheries ontologies network and map them to exploit and use the Fisheries electronic resources.
- Users are provided with mechanisms to ensure that the Fisheries ontologies and mappings are kept updated and in line with the continuous growth of the information and knowledge made available in the domain.
- Users are confident with new tools and methodologies provided by NeOn.

1.2.2 Objective 2: Exploit ontologies within web applications, and develop a Fish Stock Depletion Assessment System

This objective presupposes that appropriate mechanisms are in place to ensure that the web applications using ontologies will guarantee as a minimum the same functionalities as the systems previously used. Moreover, it is expected that the use of ontologies will bring benefits and enhance accessibility of heterogeneous data and sources. In particular, it is expected that:

- NeOn provides a framework to ensure that ontologies (data model, content, relationships and mappings between them) are actually usable by systems
- Mechanisms are implemented for ensuring validity and integrity of networked ontologies.
- NeOn provides methods and support for versioning control of deployed ontologies.
- Mechanisms are provided for static and especially run-time modularization of networked ontologies. This is particularly important considering the large size of the case study ontologies and high level performance expected of applications using the ontologies.

1.2.3 Case study success indicators

The achievement of, or at least the contribution to the above mentioned objectives can be measured at the end of the project according to the following qualitative and quantitative success indicators:

1. Quantitative: increased number of KOS contributing to the networked ontologies.
2. Quantitative/qualitative: additional relationships and knowledge associations discovered and added to fisheries RTMS based KOS.
3. Qualitative: improvement in the modelling of fisheries knowledge in order to facilitate the reusability of fisheries ontologies and modules.
4. Quantitative: staff involved in the ontology lifecycle can be instructed to apply the NeOn methodologies and use the NeOn toolkit during a training course of limited length.
5. Qualitative: partial or total replacement of resource discovery functions currently provided by the Fisheries Reference Data Management System (RTMS).
6. Quantitative: increased number of Information Systems exploiting and sharing the fisheries networked ontologies.
7. Qualitative: improved assessment and monitoring of fisheries stocks in FAO member countries.

1.3 Overview of D7.1.2

The rest of this deliverable is organized as follows:

Chapter 2 presents the revised requirements for the fisheries ontology lifecycle: it begins by giving a short overview of the fisheries ontologies lifecycle (Sec. 2.1), a description of the user roles involved in it (Sec. 2.2), and a description of the workflow for editorial duties (Sec. 2.3). Requirements for ontology engineers are given in Sec. 2.4 and requirements for ontology editors are given in Sec. 2.5. All the requirements are summarized in Annex A.

Chapter 3 presents the revised requirements for FSDAS. It begins by restating the scope of the FSDAS, highlighting a few important constraints that were until recently unnoticed by some work packages. It goes on to summarise the structure of the section and the set of deliverables that have influenced the requirements' revision. It then covers broad development process changes before plunging into a reiteration, refinement and extension of the functional requirements. Following this, the chapter covers user types and finishes up with a set of sections covering user interface issues and non-functional requirements and constraints, including a few notes on software interfaces.

Annex B completely traces all changes to existing requirements, scenarios and use cases from 7.1.1 to today, though new requirements introduced in this deliverable are not considered in the annex. Annex C contains tables the will help the reader trace the changes made to the requirements through D7.1.1 and D7.5.1. Annex D contains the list of revised use cases for FSDAS, and Annex E contains the list of new use cases for FSDAS.

2 Revised requirements for the Fisheries ontologies lifecycle management system

In this chapter, after a short overview of the fisheries ontologies lifecycle, we describe the roles involved in it (Sec. 2.2) and the workflow adopted for maintenance and validation of the ontologies (Sec. 2.3). Sections 2.4 and 2.5 describe the user requirements given for ontology engineers and ontology editors respectively. All requirements are summarized in Annex A.

2.1 The fisheries ontologies lifecycle

The lifecycle of fisheries ontologies (Figure 1) starts with the conceptualization and initial population of the ontology (according to an iterative process) by ontology engineers (left most block in Figure 1). Once the ontology engineers decide that the ontology is stable, the ontology enters a second phase where ontology editors (subject experts and validators) are able to edit the ontology, focusing on updating and validating its content (central block in Figure 1). The editing and validation phase is facilitated by the implementation of an editorial workflow. Finally, selected (stable, correct and approved) versions of the ontology pass on to the third phase for publication and use (right most block in Figure 1).

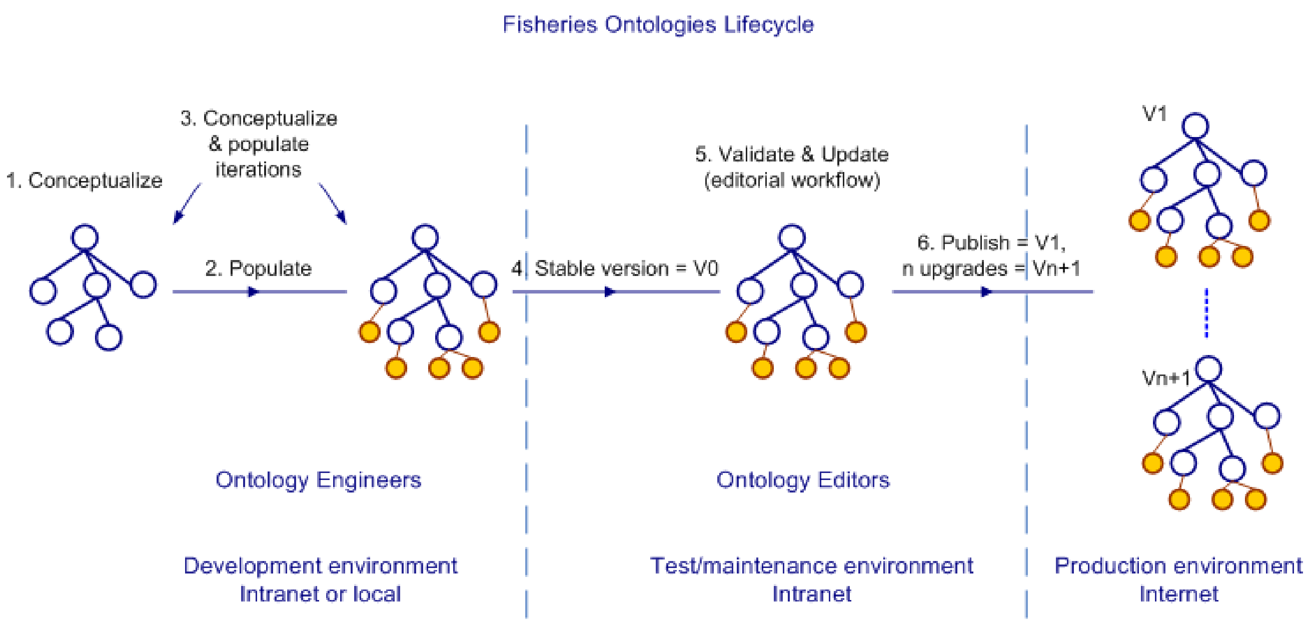

Figure 1. Fisheries ontologies lifecycle.

In the first two phases, different actors play a role and perform different actions. Therefore, different needs emerge as user requirements for the NeOn toolkit to support this lifecycle.

2.2 Fisheries ontologies lifecycle roles

FAO serves as a knowledge network and uses the expertise of its staff - agronomists, foresters, fisheries and livestock specialists, nutritionists, social scientists, economists, statisticians and other professionals - to collect, analyse and disseminate data. Given the highly interdisciplinary nature of the activities carried on in FAO, and the variety of profiles and expertise available in the organization, when dealing with streamlining the processing of managing information, lifecycles and workflows are implemented in order to involve at each stage the people who can best contribute to a given task.

This same approach is used on the fisheries ontologies lifecycle. The proper conceptualization, implementation, deployment and maintenance of networked ontologies in the fisheries domain requires that many different expertises work together, including ontology engineers, software developers, aquaculture specialists, economists, biologists, etc. Since the process is complex, many people are involved and the tasks to be performed are of very different kinds and require different skills; different people based on their ability to perform a particular task will be involved at each stage of the process.

Based on their level of responsibility within the organization, their knowledge and their skills, users are assigned the appropriate role within the ontologies lifecycle. Access rights are granted accordingly.

The possible roles comprise: ontology engineer, ontology editor - subject expert, validator – and viewer:

- **Ontology engineer:** as introduced in Sec. 2.1, ontology engineers' major tasks take place at the very beginning of the ontology lifecycle since they are involved in the conceptualization, implementation and possibly initial population of the ontology. Ontology engineers have full control of the ontology they are working on until they consider the ontology complete and stable. Then, they move the ontology from the engineering environment to the editorial environment where they may hold *view* rights only (cf. viewer role, below).

  Ontology engineers are experts in ontology modelling. They have sound knowledge of and experience with ontology tools, such as editorial tools and inference engines, but may know little about the subject to be modelled. Therefore, for ontology conceptualization they need to take into account a number of elements, such as: salient aspects of the subject (by means of interviews and information exchange with the subject experts), the purpose of the ontology, possible interactions with legacy systems, and other relevant issues.

- **Ontology editor:** they are ultimately responsible for the fisheries ontologies verification, validation and content maintenance.

  They are specialists in the various subjects related to the fisheries domain (they could be biologists, economists, or oceanographers, just to mention a few profiles), but they do not need to know much about ontology design issues or software for ontologies. For this reason, it is extremely important that GUIs for ontology editors be easy to use, intuitive and hide the complexity related to ontology modelling and its terminology.

  An editorial workflow supports ontology editors in carrying out their tasks in the Fisheries ontologies lifecycle. The editorial workflow defines the activities that can be performed by each role at a given status, i.e. who can do what and when depending on their role and the status of data. Within the fisheries ontologies editorial workflow, editors may be assigned two sub-roles: subject experts or validators:

  - **Subject expert:** users with a subject expert role are in charge of keeping ontologies updated by adding new instances or modifying or deleting existing ones.[1] They have all

[1] Please note that here we assume that subject experts will typically deal with instances only (as opposed to concept definition). This assumption is based on the observation of the fisheries ontologies produced within T7.2, and on the

the rights needed to create/update content, although their contributions need to be further validated and approved by validators (see below).

- **Validator:** users with a validator role, like subject experts, are specialists in the areas related to fisheries. In addition, they usually have a supervisor role within FAO (and a larger/broader experience) and for this reason they are in charge of content quality control. They revise, approve or reject the changes made by subject experts; they are the only ones authorized to approve ontologies to be published on the Internet as a new ontology version.

- **Viewer:** this is the only role that does not depend on particular skills or knowledge. This role has been introduced to allow authorized users to view (i.e., read-only rights) approved elements of ontologies within the editorial workflow. This particular role is used to allow ontology editors of a given ontology to consult (view) the approved data of another ontology (for which they do not have edit rights). For example, editors of the biological species ontology may need to see the water bodies ontology under development but not yet published. Since their area of expertise is different, they are not allowed to edit the water bodies ontology, but they may be granted read-only rights. This role may also be played by ontology engineers: after passing control of the ontologies they implemented on to the editors, they are no longer allowed to hold editing rights, but it may be important for them to be able to see the development of the ontologies they created. Other viewers may be users not directly involved in the editing process of the ontology, but consulted to advices on particular aspects of it.

2.3 Support of editing: editorial workflow

The central part of Figure 1 shows the activities related to the editorial processes through an editorial workflow that defines who can do what and when. The editorial workflow for fisheries ontologies is meant to allow several people to contribute to the maintenance of the ontology in a controlled and coherent manner, ensuring that only fully validated ontologies are released on the Internet. The whole editorial workflow is illustrated in Figure 2 below.

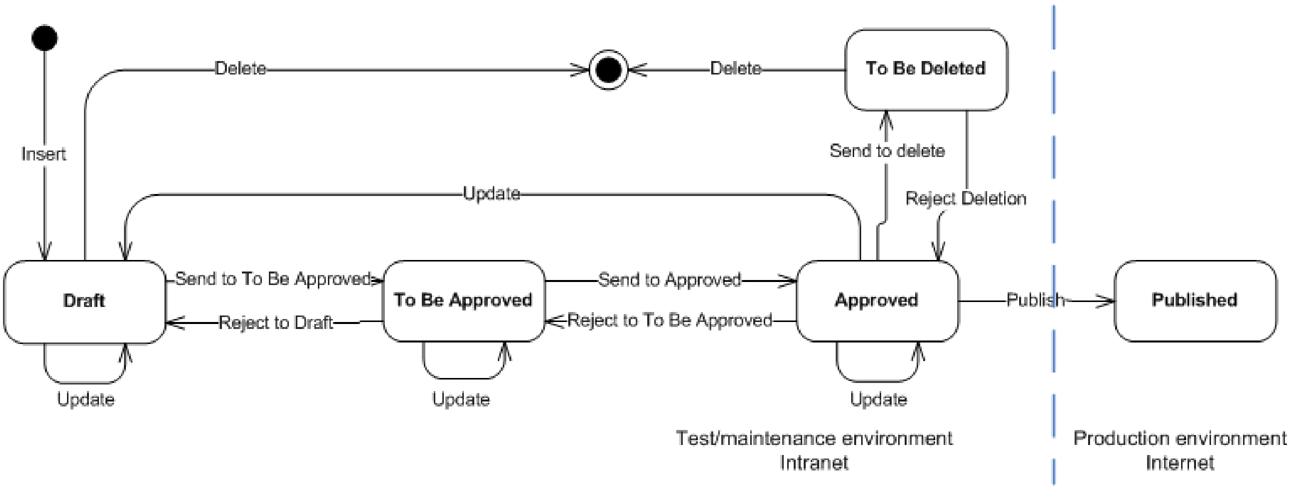

Figure 2. Overview of the whole editorial workflow.

need to not overload editors with duties that should be more appropriately assigned to ontology engineers. Here no claim is made about other use cases.

In Figure 2 states are denoted by rectangles and actions by arrows. The workflow is based on assigning a status to each element of the ontology. The ontology may be published (based on the authorization given by validators) or upgraded only when all the elements have an approved status.

Any element in the workflow has a status assigned:

- **Draft:** for a new element inserted or modified by a subject expert and not yet approved.
- **To be approved:** for a draft that, according to the subject experts who authored it, is ready to be examined by validators.
- **Approved:** for an element that has been approved by a validator.
- **Published**: for all elements in an ontology that has been released (Internet).
- **To be deleted**: if a subject expert considers that an approved element is to be deleted, the item is assigned the “to be deleted” status. A validator may approve or reject the deletion.

Draft elements may only be “seen” by the subject expert who created the new draft. An element keeps the status “draft” until the expert decides to send it for approval. Viewers are allowed to see only the approved elements (not yet published).

The workflow allows the system to define who (depending on the role of the user) can do what (actions explained below) and when (depending on the status of the element and the role of the user).

Subject experts are able to:

- Insert a new element or update an approved element. In both cases, the system automatically assigns a draft status to the element. When an element has a draft status, subject experts can update it as required. The item keeps its draft status until the subject expert decides that it is ready for approval and then sends it to the next status, “to be approved”. This action automatically moves the responsibility of the item from the subject expert to the validator. While an item has the “to be approved” status the subject expert cannot modify it.
- Delete an element with a draft status.
- Propose an approved item to be deleted. The item is assigned the “to be deleted” status, and only a validator is able to accept or reject the deletion.

Validators revise, approve or reject changes made by subject experts and they are the only users who can copy changes into the production environment. Validators are able to:

- Update an approved or a “to be approved” element. When a validator makes a modification, it does not need to be approved by others, so the element keeps its status.
- If an element is assigned the “to be approved” status, the validator can either approve, reject (to draft status), or modify it.
- If an element is assigned the approved status, the validator can either reject it (returns to the “to be approved” status), delete it (sends the element to the “to be deleted” status) or modify it.
- Send an approved element to the “to be deleted” status and even destroy an element in the “to be deleted” status.
- A validator can reject the deletion of an element that has been assigned the "to be deleted" status. If deletion is rejected, the element returns to the approved status.

- When all the elements of an ontology have been approved, the validator can publish it. This copies the approved ontology into the production environment and automatically assigns the correct version (V1 for the first release and Vn+1 for each subsequent release).

2.4 Requirements for ontology engineers

The requirements listed in this section concern the design and implementation of networked ontologies (step 1 in Figure 1), which are tasks assumed to be carried out by ontology engineers.

2.4.1 Ontology implementation and reuse

Requirements for Ontology implementation and reuse are summarized in Tables in this annex provide a summary of the requirements presented in Chapter 2.

Requirements are indicated (first column) by the section number in they appear followed by the actual requirement number. For example, "Requirement # 2.3.1 1" corresponds to the first requirement in Sec. 2.3.1.

Table 1 (in Annex A).

In this section we specify the requirements for the creation and implementation of new ontologies, either from scratch of by means of reusing (and reengineering as required) ontological and non-ontological resources. Tables in this annex provide a summary of the requirements presented in Chapter 2.

Requirements are indicated (first column) by the section number in they appear followed by the actual requirement number. For example, "Requirement # 2.3.1 1" corresponds to the first requirement in Sec. 2.3.1.

Table 1 summarizes these requirements together with their mappings to functionalities to be implemented in the NeOn toolkit.

1. **Support of ontology implementation.** The ontology engineer shall be able to create new ontologies and elements in them (i.e., classes, instances, modules, properties, axioms) using any of the most common languages and vocabularies for ontologies, such as OWL, RDFS, and SKOS.
2. **Non ontological resource reengineering.** A number of non-ontological fisheries-related resources are already available in FAO and are 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 [ISSCAAP00]), and existing FAO's Knowledge Organization Systems (KOS), such as FAOTERM, Fisheries Glossary, etc. These resources are often either stored in relational databases or in XML documents. Thus, the system shall support ontology engineers in creating ontologies based on these resources. In particular, it shall allow them to:
    a. **import data from databases.** As taxonomies, classification schemas and thesauri are commonly stored in relational databases, it shall be possible to connect to a RDBMS, view (and/or import) the relevant tables (logical structure and content) and import the data according to the defined ontological model. Deliverable D7.2.2 [D7.2.2] reports on conversion work of this type.
    b. **runtime access to databases.** In many cases it is advisable [CAR07] to keep the data in the relational database and access it through an ontological layer (ontology).

The system shall then support ontology engineers in defining the appropriate ontology, mapping it onto the database and accessing the data (i.e., without physically export it from the database). Facilities shall be provided to enable ontology engineers in “adding” and exploiting relations and mappings not present in the database.

c. **thesauri.** Mechanisms shall be in place to support (semi)automatic conversion of thesauri into ontologies (RDFS, OWL).

3. **Ontology reuse, reengineering, integration.** New ontologies may be created on the basis of existing ones, either by transforming the conceptual model of an existing and implemented ontology into a new one (reengineering) or by including the existing ontology into the new on (integration). The ontology engineer shall be able to open and visualize any ontology (at least with view rights) 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 and paste it into the ontology being created (edited). The engineer shall also able to create mappings between the two ontologies.

2.4.2 Editing

Requirements for editing are summarized in Table 2 (in Annex A).

4. **Edit (single and multiple) ontologies**. Granted the appropriate rights, ontology engineers shall be able to edit the necessary ontology elements, including mappings between ontologies and relations across ontologies.
5. **Adding a new language.** Ontology engineers shall be able to add a new language to an monolingual or already multilingual ontology.

2.4.3 Documentation

Requirements for documentation are summarized in Table 3 (in Annex A).

6. **Creation of documentation.** The system shall support ontology engineers in the creation of documentation for ontologies, in particular concerning ontology design. Documentation is necessary for both ontology engineers and ontology editors, therefore it shall be possible to use the most appropriate form of documentation for each group. An ontology should be documented at least by means of a synthetic description of the conceptual structures involved, and by means of UML-like diagrams. An example of documentation style suitable to both ontology engineers and editors is the documentation produced by the OWLDoc tool for OWL ontologies [OWLDOC].[2]

2.4.4 Ontology modularization

Requirements for ontology modularization are summarized in Table 4 (in Annex A)

Although work on modularization is currently in progress within NeOn (see deliverable D1.1.3, due at M24), we have identified when modularization is necessary (item 7 below) and examples of possible modules taken from WP7 ontologies.

[2] Depending on the configuration options adopted, OWLDoc may also be used to visualize ontologies to non expert users.

7. Especially in case of large ontologies, it is important that facilities be provided to support users in defining and selecting “fragments” of ontologies, which we call modules. 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).

8. **Some examples of modules found relevant to WP7 use case.** Mechanisms shall be in place to allow ontology engineers to create at least the following (types of) modules:

   a. **modules by language.** Multilingual ontologies may be available in several languages (e.g., fisheries ontologies described in [D7.2.2] contain three or four languages, AGROVOC sixteen) although for common applications not all of them are used. It is then useful to select only the languages used for the specific application at hand.

   b. **modules by code.** Most fisheries ontologies include one or more classification systems, although not all of them are used at the same time. It is useful to be able to define, select, visualize and utilize only the module of the ontology where the desired classification system is used. Examples of classification systems are: ISO2 [ISO2] and ISO3 [ISO3], ISSCAAP code [ISSCAAP00], the ISSCFV [ISSCFV], ISSCFG [ISSCFG], (see [D7.2.2]).

   c. **modules by “topic”.** In case of large ontologies, covering more than one domain (e.g. AGROVOC, ASFA) it is useful to be able to select modules on the basis of the domain covered, such as “fisheries”, “aquaculture”, “pests”, etc.

   d. **modules for editorial duties.** Another module specification is related to editorial work. Especially when dealing with large ontologies, it is important to be able to select the ontology elements that are involved in the editorial work of a given editor. This is to avoid editors being given the entire ontology, when smaller, more manageable and focused parts of the ontology can be defined and extracted.

2.5 Requirements for ontology editors (subject experts and validators)

The requirements specified in this section concern the everyday editorial work of maintenance, update and validation of ontologies. This work is carried out by ontology editors within a workflow (cf. Sec. 2.3). We use the term “editing” to refer to the activities of adding and modifying instances (with properties and relations), metadata and annotations.

2.5.1 Editing

Requirements for editing are summarized in Table 5 (in Annex A)

9. **Editing ontology elements.** The editing environment shall allow editors to edit one or more ontologies at a time (assuming the appropriate editing rights are granted). Depending on the specific ontologies at hand, editorial duties may be more focused on specific ontology elements (instances, classes, properties, relations). The control over the action performed is left to the validators within the editorial workflow.

10. **Editing relations between ontologies.** Editors shall be able to establish relations between ontology instances residing in different ontologies.

11. **Editing mappings between ontologies.** The user shall be able to draw mappings (manually) between pairs of ontology elements (and consequently between ontologies) such as:

    a. concept – concept

    b. instance – instance

    c. module – module

    In Sec 2.5.4 we consider semi-automatically created mappings, provided by software facilities to support editors in their duties.

12. **Creation and management of metadata.** Metadata is essential to a number of activities, including collaborative editing. The editing environment shall automatically attach and manage the following pieces of metadata:

    a. date of creation/editing of the element and author (automatically);

    b. information about author/editors of the ontology elements,

    c. history of changes,

    d. summary statistics on the ontology,

13. **Creation and management of annotations.** Editors shall be able to enrich an ontological element with textual and multimedia annotations. Multilingual textual annotation shall be supported. Some examples are: "scope notes" (as commonly used in thesauri), images, free-text comments.

2.5.2 Quality check

Requirements for quality check are summarized in Table 6 (in Annex A)

14. **Support of quality check.** When revising work done by others, validators shall be provided with mechanisms for checking the quality of the development of the ontology to be revised. In particular, it shall be possible to:

    a. **check for duplicated elements**, such as instances having exactly the same pieces of information (e.g., labels, properties values).

    b. **check for "similar concepts".** A simple case of similarity that shall be taken into account applies when two instances appear identical to humans but are not identical to a machine (for example in case of spelling mistakes). For human editors and users, these two instances are identical;

15. **Compare ontologies.** In order to compare two ontologies, ontology editors need support from the system. This support shall be both visual (i.e., visualize two ontologies at the same time, cf. Sec. 2.5.5), and by means of statistics (that could also be visually shown side by side) about the two ontologies to compare (see below).

16. **Summary statistics.** A number of summary statistics are useful to control the development of an ontology, including:

a. depth of the ontology,

b. number of ontology elements (classes, instances per class, relations, properties),

c. number of mappings and relations between external ontologies,

d. distribution of subclasses per top level classes,

2.5.3 Support of editing: ontology population

Requirements for population are summarized in Table 7 (in Annex A)

Ontology population tools shall be accessible to editors through the editorial environment, in order to suggest candidate instances to add to the ontology, on the basis of structured, semi-structured and unstructured data sources. We assume that ontology editors only work with instances.

17. **Suggest candidate elements.** On the basis of given textual corpora, the tool shall provide the author with a list of candidate elements suitable for inclusion in the ontology. Ontology editors may use support for the creation of instances or relations between instances (also between instances that do not reside in the same ontology).

18. **Support candidate selection.** Editors shall have facilities to inspect and select the candidates suggested by the system. In particular:

    a. The tool shall show the documents and excerpts supporting the extracted terminology, including document metadata such as title of the document, author, data owner, publication date.

    b. The tool shall provide the editor with a way to inspect and select the appropriate candidate, and add the selected ones to the ontology.

19. **Preview and check consistency of the newly added elements.** Before final inclusion in the ontology, it shall be possible to visualize the ontology including the newly added elements, and check it for consistency.

2.5.4 Support of editing: mapping creation

Requirements for mapping creation are summarized in Table 8 (in Annex A)

Tools supporting editors in the creation of mapping between ontologies shall be available.

20. Editors shall be able to define mappings between ontology elements. Editors shall also be supported by the system in the process of establishing ontology mappings: the system would present the editor a list of candidate mappings, the relevant ones would be chosen by the editor and added to the ontologies.

2.5.5 Visualization

Requirements for visualization are summarized in Table 9 (in Annex A)

Visualization and browsing facilities are fundamental in order to allow people working and interacting with the system. Therefore, it should be assumed that all of the requirements previously given come with accompanying requirements on visualization. In this section we group together all visualization-related requirements, adding references to requirements previously given, as appropriate. Note that in this section we do not mention the activity of browsing ontologies, as it is assumed that all interfaces for ontologies must be able to visualize and browse them.

21. **Dedicated interface for ontology editors.** Ontology editors (both subject experts and validators) shall be provided with a simplified interface for all editing actions. In particular, the terminology adopted in the interface shall be such that the editor is not required to know details of the specific language used to encode the ontology.

22. **Visualization modes.** Ontologies (and modules) shall be visualized in different ways, depending on the task to be performed (e.g. editing vs. revision of the ontology), and the purpose and the preference of the editor/author. The following view modes shall be allowed: diagram-like, indented tree, node by node, possibly together with parents and children.

23. **Visualization of single ontologies.** Ontology editors shall be able to visualize and browse single ontologies (appropriate visualization shall be provided in case mappings and relations to other ontologies are present).

24. **Visualization of multiple ontologies.** The visualization of more than one ontology at the same time is very important. This will typically happen when comparing two ontologies (and possibly editing one on the basis of the other), or establishing, visualizing or validating mappings and relations between them.

    Editors shall be able to visualize:

    a. ontology elements from both ontologies plus mappings and relations between them;

    b. the overlapping between ontologies, i.e., ontology elements present in both ontologies. A clear visualization of overlapping between ontologies is useful as a support to mapping creation, and during editorial workflow

25. **Visualization of metadata**. During the process of updating and validating the ontology, it is important that a number of pieces of information be highlighted to the user. In particular: editing history of the ontology element, including its authors and provenance, summary statistics.

26. **Visualization of multilingual information.** Multilingual ontologies present challenges to visualization, especially when contents in many languages is available. Ontology editors shall be able to select the languages (one or more, out of those available) to use to display the ontology.[3]

27. **Visualization of candidate elements for inclusion in an ontology.** The requirement listed in 2.5.3 concerning the editorial support to ontology population shall be provided with an adequate visualization and interface in order to allow ontology editors to select, inspect, approve and include candidate elements in the ontology.

28. **Print visualization.** It shall be possible to print out the chosen visualization(s).

[3] The interfaces for navigation also need to support multilinguality: users can chose among a list of languages other than English to use and interact with the system (i.e., including menus, navigation bars, error and warning messages etc.). This is a general requirement, which does not only apply to ontology editors but to all human users involved in both the ontologies lifecycles and the FSDAS. Besides English, at least Spanish and French shall be considered.

2.5.6 Workflow visualization

Requirements for Workflow visualization are summarized in Table 10 (in Annex A)

It is important to provide a simple, intuitive interface for each of the user roles in the editorial workflow (cf. Sec. 2.3) so that tasks are performed efficiently and effectively. Four specific views are required, based on the user roles and element status:

- **Draft** view: approved information plus changes made by the current editor, with the difference between the two states clearly visible. This view should be available only to the subject expert who made the draft.
- **To be approved** view: approved information plus all the pending elements to be approved by validators, with the difference between the two states clearly visible. This view is for validators.
- **Approved** view: approved information only. For all users, including viewers.
- **To be deleted** view: approved information plus elements that are assigned “to be deleted”, with the difference between the two states clearly visible. This view is for validators and for the editor who made the draft.

Beyond the status of the ontology elements, it is important that a number of pieces of information be visualized properly in order to support their work (both as subject expert and as validators), in particular metadata as mentioned in Sec. 2.5.5.

2.5.7 Ontology export

Requirements for Ontology export are summarized in Table 11 (in Annex A)

29. **Export ontologies into other formats for backward compatibility.** It shall be possible to export ontologies in several output formats in order to facilitate data exchange with legacy systems and uniformity with existing FAO resources. In particular, it shall be possible to export ontologies converting the schema (and included instances) according to relational database design principles, or as SKOS ontologies [SKOS] .

2.5.8 Search within ontology

Requirements for Search are summarized in Table 12 (in Annex A)

Search facilities are commonly used by ontology editors (subject experts and validators) and ontology engineers in their daily work. Editors need it in order to check the presence of ontology elements in the ontology they are working on (or in other available ontologies). Validators use search to support to their work when checking the contribution provided by the subject expert.

30. **Search.** Users shall be able to search within ontologies and legacy systems (i.e., thesauri, classification schemas, terminology systems). Search shall be possible on the ontologies being visualized and across the used repository of ontologies :

    a. **structural search.** It is useful, especially to validators and ontology engineers, to be able to perform searches that exploit structural aspects of the ontology. For example, to identify instances of classes with a common ancestor, to select relations with a given domain and/or range, or to find instances with one or more given properties;

    b. **textual search.** It is useful to be able to search for text across ontologies, independently of wher the text appears (labels, properties, annotations, etc.);

c. **search by URI.** It may be useful to be able to search for ontology elements given a URI.

2.5.9 Multilinguality

Requirements for Multilinguality are summarized in Table 13 (in Annex A)

31. **Deal with multilingual ontologies.** Editors deal with the multilingual aspect of fisheries ontologies. They shall be able to:

    a. **selection of languages.** select at least two languages (or more, if required), one in view mode, the other in editing mode;

    b. **edit multilingual labels.** add/edit/delete multilingual labels to individual concepts;

3 Revised requirements for FSDAS

The following section summarizes the changes that have taken place since the delivery of the pre-NeOn requirements document D7.1.1. It reiterates the scope of FSDAS, touching on some important points that have at times gone unnoticed by other work packages. It then goes on to discuss the major modifications, refinements and extensions to the existing requirements.

For a complete traceable breakdown that passes from D7.1.1 Requirements through D7.5.1 FSDAS architecture to D7.1.2 Updated requirements please refer to Annex C – FSDAS Traceability Tables. Note that the annex does not contain new requirements introduced in this deliverable.

3.1 FSDAS scope

The scope of the FSDAS within the NeOn project has remained the same. As noted in the introduction to 7.1.1, "The aim of NeOn - to create the first ever service-oriented, open infrastructure, and associated methodology to support the development lifecycle for a new generation of semantic applications - *provides FAO Fishery systems with a great opportunity to develop an appropriate framework to manage fishery ontologies and their lifecycle, as well as to implement a semantic fishery stock depletion alert system that exploits those ontologies*."

The only real change has been in the unfortunate naming of the product, which was done before the Fisheries department could give its considered opinion. The use of the word "Alert" has proven extremely unpopular and totally unviable for the community. The decision was made early on to change this to "Assessment". Any references to the word "Alert" in 7.1.1 are invalid and shall be treated as "Assessment".

It should also be noted that though not explicitly stated in the NeOn project formulation, there was an expectation by at least some project members that FSDAS would be able to function completely within a local instantiation of the toolkit, which is not the case due to a variety of non-functional requirements such as fire-walled databases.

Although a client-server design was already implicit in the D7.1.1 requirements delivered in November 2006 and re-iterated at the Bled plenary of January 2007, many teams expressed surprise at this design choice during the draft presentation of the architecture in June 2007 at the Dubrovnik plenary. It is therefore important to reiterate the fact that FAO as a case study is testing the toolkit's handling of the ontology lifecycle within the Eclipse platform as expected. FAO is also testing an FSDAS instantiation that while using basic architectural components, is nevertheless not wholly integrated in the Eclipse environment as it requires a client-server model not fully supported by the toolkit's plug-in architecture.

3.2 FSDAS perspective

The requirements for FSDAS as presented in D7.1.1 were prepared at the beginning of the project lifecycle when little was known about the proposed functionality of supporting toolkit components. Thus, some areas were deliberately underspecified or left fairly generic. The 7.2.1 fisheries inventory made clearer the underlying data that would form the knowledge base. This was followed by the preparation of D7.5.1 FSDAS Architecture, a time in which many aspects of the overall NeOn project were becoming clearer, and this necessitated some modifications to the requirements in order to prepare an architecture that was coherent with the reality of toolkit development.

The requirements presented in D7.1.1 have been analysed and compared to modifications made during the creation of the software architecture in D7.5.1 in order to arrive at a global vision of the changes that have occurred. The changes to requirements are summarised in Tables in this annex trace the changes (modifications /refinements /additions /deletion /extensions /generalizations) with respect to the changes presented in D7.1.1 and D7.5.1.

Table 16 (Annex C). As it might be expected, the majority of the changes are to non-functional requirements, i.e. the tasks that FSDAS is expected to perform have remained largely the same, while some of the *ways* in which it is expected to perform those tasks have changed.

In addition to requirements changes, during the creation of D7.5.1 there were a number of small changes made to the use cases developed for 7.1.1, mostly a case of clarification and either refinement or generalisation of the case. These modifications can already be found in the appendices of D7.5.1. Here the changes are documented in Tables in this annex list the use cases presented in D7.1.1 and revised in this deliverable.

Table 17 (Annex C).

The following sections summarise the modifications to 7.1.1. As was done for D7.1.1, the headings consist of a subset of the IEEE SRS.

3.3 FSDAS features

This section summarises functional changes to the product requirements. While the functional scope of FSDAS remains largely the same, some ambitious or overly sensitive functionality has been reduced or removed:

> "Some of the functionalities the alert-system will provide are: a search environment, user customization based on user knowledge level, user specification of updates (e.g. email notifications or RSS feeds) and ~~**alerts**~~ (reports) based on countries or species."

Other requirements were not abandoned but relegated to low-priority or second iteration (underlined passages) due to the perceived difficulty of their implementation by project partners:

> *Reduced data sources:*
>
> "Fisheries resources will be exploited by ontologies to return time-series statistics on capture, production, commodities and fleets by stock, together with direct links to related documents, web pages, news items, images and multi-media."
>
> *Simpler ontology navigation:*
>
> "Users shall be able to perform browse-based and query-based searches on single ontologies or on the union, intersection or complement of more than one ontology. They will also be able to navigate the associated data instances[4]."

[4] The term "data instance" refers to any digital resource or combination thereof accessible via a software service including but not limited to WSDL, JDBC, REST, ODBC, RMI and CORBA that are stored in FAO or other information systems, and that are capable of some user interaction via a user interface, e.g. a pdf document. The data instance may be ontologically represented by the instantiation of the concept(s) used to index that digital resource.

Little reporting functionality:

"To the extent possible, the FSDAS will directly introduce and/or combine resources into the web page to create dynamic, synthetic views of the state of fish stocks."

Finally, it should be noted that following the work done in D7.5.1 on architecture, statements regarding interfaces or API's such as the following can be safely ignored: "The Services and API relevant for the alert system are: concept service, inference service and data warehouse."

3.4 FSDAS function summary

This section reprises, reorganises and refines the functional requirements described in 7.1.1. It also extends them by adding some requirements discovered during the design phase.

Users will use the FSDAS to gather, analyze and produce information on the status and trends of fish stock. For this purpose, the ontology-based FSDAS will allow authorized users to browse and query a semantic knowledge base of fisheries digital resources (FAO and selected non-FAO). The major user requirements for the FSDAS are:

3.4.1 Interface

1. Users shall be able to register, login and maintain a profile to store and access favourite queries, ontologies, ontology concepts, default language and ontology subsets.
2. Users shall be able to join a particular user type, e.g. biologist, economist, that will determine the default system settings such as ontology loaded and underlying data systems used for reasoning and data instance display.
3. Users shall be able to search for available ontologies and load them into the system.
4. Users shall be able to annotate (including comments, keywords and quality scores) and rate concepts, attributes, relations and associated data instances.
5. Users shall be able to select to rank results based on the rankings filtered by their user type or other annotation data.
6. Users shall be able to email formatted results.
7. Users shall be able to suggest ontology modifications that will be forwarded to the ontology owner via email.
8. Users shall be able to generate a URL RSS feed from a query result such that the feed can be added to any RSS aggregator and so monitor additions to the knowledge base related to that query result.
9. Users shall be able to view all concepts, relations and the application interface in the five languages of FAO, when available.
10. The system shall support several modes of viewing ontologies:
    a. Rubber-band

   b. Hierarchical

3.4.2 Model

1. The system shall allow authorized users to browse and query fisheries concepts, attributes and relations.

2. Fisheries concepts, attributes and relations will be ontological models.

3. The system shall be able to display multiple ontologies, including the relations between those ontologies as if they were one ontology.

4. The system shall be able to display portions or modules of ontologies:

   a. It shall be possible for a user to disable view of an entire class.

   b. It shall be possible for a user to disable view of all instances of a certain relation.

   c. It shall be possible for a user to view only one branch of an ontology.

5. The restriction of an ontological view to a subset of that view shall also limit the semantics of any search or reasoning against underlying data instances.

6. Users shall be able to input ad-hoc queries, both using free-text and by highlighting concepts/relationships suggested by the currently loaded ontology set that shall return either related data instances or related concepts/relationships found in the currently loaded ontology set.

   a. In this context Boolean logic, phrase matching and query refinement shall be supported by the system.

3.4.3 Data instances

1. Users shall be able to view the digital resources related to concept instances.

2. Raw data that is not associated with a proprietary application shall be viewable within the system, and not require the launching of a separate application. This applies specifically to tabular data such as CSV files, hierarchical data such as XML files and HTML files, and image data such as PNG, JPEG and GIF files.

   a. Text views shall highlight the areas related to the ontological query; e.g. as done by *Magpie* [MAG].

   b. It shall be possible to view two data sources side by side for comparison.

   c. It shall be possible to identify trends across returned data instances

   d. For tabular data it shall be possible to view them also as charts or graphs, such as provided by reporting tools such as BIRT.

   e. For time-series type data, it shall be possible to compare two or more time periods.

3. Data instances linked to proprietary applications such as DOC and PDF files are either:

   a. URL hyperlinks to the data, or

   b. the actual data or a portion of the data (by using a parser/extractor such as is done by popular search engines).

4. Data instances shall be presented within their associated ontological context and associated metadata crucial to understanding shall also be displayed.

   a. If present, data instance quality and/or ranking metadata shall be displayed.

   b. The data instance provenance shall be displayed.

   c. If known the last modification of the data shall be displayed.

   d. If known the creation date of the data instance shall be displayed.

5. Data instances returned by a query shall be grouped according to their related concepts and relationships.

6. It shall be possible to re-organize data instances according to any of the concepts/relationships that are associated to those data instances.

3.5 FSDAS User types and characteristics

The user types in D7.1.1 were too FAO-based and unrealistic considering the institutional arrangements that the Fisheries and Aquaculture department has at both regional and national level. The user types have therefore been extended and should read:

> FSDAS users will be officers in FAO Fisheries Department attached to FAO headquarters and regional and sub-regional offices around the world, as well as Regional Fisheries Bodies and national-level fisheries scientists.

3.6 FSDAS operating environment

In D7.1.1 it was not made explicitly clear that there are <u>*no* FAO databases that can be accessed via ODBC/JDBC from outside the firewall</u>. This is an important constraint that influenced the initial D7.1.1 requirements in the direction of a client/server model, but was not made explicit.

As noted in *3.8 Software Interfaces,* the FAO operating environment now also permits MySQL databases.

Given continuing reductions in hardware costs and the brute fact that ontological systems are memory intensive, the original specification for an application with a memory footprint below one gigabyte can be changed to four. Note that this refers to the <u>server-side only. The client-side machine remains at one gigabyte.</u>

3.7 FSDAS design and implementation constraints

This section summarises the non-functional changes to requirements that do not fit into the other categories. Several references to non-functional requirements of implementation technology have changed, particularly the original idea to realize FSDAS as a web application. During the development of D7.5.1 FSDAS Architecture, the decision was made to take advantage of the greater power offered by the Java programming language to meet the needs of the user interface as described in the requirements. Numerous references to "web application" and "browser-based" are now incorrect and should read "Java web-start application". As stated in chapter three of D7.1.1, *Product perspective*:

> "FSDAS will be an ontology-driven decision support system for fisheries managers, policy makers and researchers. It will be an (~~**web-based)**~~ intelligent agent that uses networked ontologies consisting of various fisheries, taxonomic, and geographical ontologies, together with their mappings and contexts, to aid users in discovering resources and relationships related to stock depletion."
>
> "End users will experience FSDAS as a browsable and queryable ~~**web**~~ application that returns organized, ranked, quality-rated, linked results that can be used to make decisions about the state and trends of various fish stocks."

In light of the decision to use a client-server model due to fire-walled databases, some other implementation-based statements now also look simplistic and must be revised:

> "The ontology-driven FSDAS will comprise the FSDAS user interface, a network of ontologies, and a number of web services extending the NeOn toolkit infrastructure."

should read instead:

> "The ontology-driven FSDAS will comprise on the client-side the FSDAS user interface, and on the server-side a network of ontologies and a number of supporting services extending the NeOn toolkit infrastructure."

In addition, the following constraints apply:

- For multi-lingual support the system shall be designed such that extension to new languages involves no more than adding additional language files .
- For ontological language support, the system shall support OWL.
- Underlying data instances shall include tabular data both textual and numeric and expressed either as an SQL query result or as a CSV file, XML data, HTML data, multimedia, and various proprietary formats, specifically Microsoft Office documents and Adobe PDF documents.
- Data instances may be contained in relational databases, document repositories, ftp sites or web sites and the system should be able to maintain a dynamic view of these related data systems, either by run-time queries or periodic polling. For folder-based systems such as ftp and web sites, the system shall be able to crawl these systems by following links and/or folder structures in order to determine the true extent of contained resources.

3.8 FSDAS user interfaces

As noted in Sec. *3.7 Design and Implementation Constraints*, several references made to the FAO guidelines on web development are no longer true, e.g. "The minimum set of requirements for web interfaces are provided in Annex II: WAICENT checklist for the clearance of web sites" can now be ignored. Specifically, the following is no longer valid:

> "The FSDAS user interface will be web-based and compatible with Microsoft Internet Explorer 6.0+ and Mozilla Firefox 1.5+. Given the large amount of data to be loaded and managed, the use of AJAX technology would need to be explored."

3.9 FSDAS software interfaces

D7.1.1 noted the need for FSDAS to interact with Oracle databases, the FAO corporate standard. Since then the standard has changed and now also allows for MySQL databases.

Given the expansion of FSDAS to a wider set of users, the requirement for a FAO-compatible authentication service has been removed. Therefore the following statement is no longer valid:

> "should be compatible with current FAO standard to allow FAO users to login with their network credentials."

4 Conclusions

As in any long project, user and software requirements evolve with time, and the same happened with the requirements for the Fisheries case study. This deliverable presented the status of the user requirements for the case study 22 months after the launch of the NeOn project.

Requirements have not change substantially, however other deliverables produced within WP7, and the work of all the other work packages the case study is linked with, helped in refining some of the functional requirements and influenced a number of decisions related to non-functional requirements.

Chapters 2 and 3 of this document present a comprehensive status of the requirements for the case study as of today, taking into consideration research work, feedback and lessons learned from NeOn deliverables to date.

Although substantial changes are not foreseen in the future, the evaluation of the first prototypes for the Fisheries ontologies lifecycle management system, due in M24, and the FSDAS, due in M26, will certainly allow users to provide further feedback to the NeOn project and contribute to the continued enhancement of the usability of NeOn technologies and methodologies by the users.

Bibliography

[AGROVOC] FAO. AGROVOC thesaurus. http://www.fao.org/aims/ag_intro.htm

[ASFA] FAO. ASFA thesaurus. http://www4.fao.org/asfa/asfa.htm

[CAR07] C. Caracciolo, M. Iglesias Sucasas, J. Keizer. Towards Interoperability of Geopolitical Information within FAO.

[D1.1.1] D1.1.1. Networked Ontology Model. NeOn project report. http://www.neon-project.org/web-content/index.php?option=com_weblinks&catid=17&Itemid=35

[D7.1.1] D7.1.1. Specification of user requirements on the case study. 2006. NeOn project report. http://www.neon-project.org/web-content/index.php?option=com_weblinks&catid=17&Itemid=35

[D7.2.1] D7.2.1. Inventory of fishery resources and information management systems. 2007. NeOn project report. http://www.neon-project.org/web-content/index.php?option=com_weblinks&catid=17&Itemid=35

[D7.3.1] D7.3.1: Results from the experiments on ontology learning (incl. evaluation and recommendations)

[D7.4.1] D7.4.1. Software Architecture for managing the Fisheries Ontologies Lifecycle. NeOn project report.

[D7.5.1] D7.5.1: Software architecture for Fisheries Stock Depletion Assessment System 2007. http://www.neon-project.org/web-content/index.php?option=com_weblinks&task=view&catid=17&id=99

[FA] FAO. Fisheries Fact Sheet. http://www.fao.org/fi/website/FISearch.do?dom=factsheets

[FAOdiv] CWP Handbook of Fishery Statistical Standards. Fishing Areas for Statistical Purposes. http://www.fao.org/fi/website/FIRetrieveAction.do?dom=ontology&xml=sectionH.xml

[FISTAT] FAO. Fisheries and Aquaculture Department. Statistics. http://www.fao.org/fi/website/FIRetrieveAction.do?dom=topic&fid=16062

[FS] FAO. Fisheries fact sheets. http://www.fao.org/fi/website/FIRetrieveAction.do?dom=topic&fid=16062&lang=en

[FSdic] FIGIS XML. List of elements. http://www.fao.org/fi/figis/devcon/diXionary/figisdoc3.5.html

[FSschema] XML schema for Fisheries Fact Sheets. http://www.fao.org/fi/figis/devcon/schema/3_6/fi.xsd

[ISO2] International Standard Organization (ISO): ISO 3166 ALPHA-2, 2006.

[ISO3] International Standard Organization (ISO): ISO 3166 ALPHA-3, 2006.

[ISSCAAP00] FAO. International Standard Statistical Classification of Aquatic Animals and Plants (ISSCAAP). Version in use from 2000 available at: ftp://ftp.fao.org/FI/DOCUMENT/cwp/handbook/annex/AnnexS2listISSCAAP2000.pdf

[ISSCFV] International Standard Statistical Classification of Fishery Vessels (ISSCFV) by Vessel Types, in use until 1995. 1984. ftp://ftp.fao.org/FI/DOCUMENT/cwp/handbook/annex/annexLII.pdf

[ISSCFG] International Standard Statistical Classification of Fishing Gear (ISSCFG) ftp://ftp.fao.org/FI/DOCUMENT/cwp/handbook/annex/AnnexM1fishinggear.pdf

[GLOS] M. C. Suárez-Figueroa. NeOn glossary of activities. http://www.neon-project.org/web-content/images/Publications/neonglossaryofactivities.pdf

[MAG04] M. Dzbor, E. Motta, J.B. Domingue. Opening Up Magpie via Semantic Services. In *Proc. of the 3rd Intl. Semantic Web Conference*, November 2004, Japan. Magpie website: http://kmi.open.ac.uk/projects/magpie/main.html

[OWLDOC] http://www.co-ode.org/downloads/owldoc/

[SKOS] Simple Knowledge Organization Systems (SKOS). http://www.w3.org/2004/02/skos/

Annex A – Summary of revised requirements for the fisheries ontologies lifecycle

Tables in this annex provide a summary of the requirements presented in Chapter 2.

Requirements are indicated (first column) by the section number in they appear followed by the actual requirement number. For example, "Requirement # 2.3.1 1" corresponds to the first requirement in Sec. 2.3.1.

Table 1 Requirements for ontology engineers: ontology implementation and reuse.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.4.1 1	Create ontology model	Model Creation	FR	Engineering	4.3.1 in D7.1.1	High
2.4.1 2	Create ontology reengineering existing non ontological data model	Model Creation	FR	Engineering	4.3.1 in D7.1.1 UC5 in D7.4.1	High
2.4.1 2.a	Create ontology for interfacing DB data	Model Creation	FR	Engineering	4.3.1 in D7.1.1 UC5 in D7.4.1	High
2.4.1 2.b.	Assisted transformation of existing data model to ontology model	Model Creation	FR	Engineering	4.3.1 in D7.1.1 UC5 in D7.4.1	High
2.4.1 2.c	Create ontology integrating existing ontologies	Model Creation	FR	Engineering	New	Low
2.4.1 3	Create ontology reusing/re-engineering existing Ontologies	Model Creation	FR	Engineering	New	Low

Table 2 Requirements for ontology engineers: editing.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.4.2 4	Change textual attributes of elements of the ontology	Editing	FR	Distributed Components	New	High
3 4	Run simultaneous editing jobs for multiple Ontologies	Editing	FR	Distributed Components	New	Medium
3 5	Add lexicalization in a new language (localization)	Editing	FR	Engineering	4.4.7 in D7.1.1 UC3 in D7.4.1	High

Table 3 Requirements for ontology engineers: documentation.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.4.3 6	Creation of documentation	Ontology Documentation	FR	Engineering	4.3.1 in D7.1.1 UC13 in D7.4.1	High

Table 4 Requirements for ontology engineers: ontology modularization.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.4.4 4.a	Identify ontology module by specifying characterizing criteria of the excerpted fragment	Modularize ontology	FR	Engineering	4.3.2 in D7.1.1 UC8 in D7.4.1	High

Table 5 Requirements for ontology editors (subject experts and validators): editing.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.1.9	Edit single element of the ontology	Editing	NFR	Distributed Components	UC7 in D7.4.1	High
2.5.1. 9	Run simultaneous editing jobs for multiple ontologies	Editing	FR	Engineering	4.4.1 in D7.1.1	Medium
2.5.1.12	Generate metadata related to the editorial workflow of the edited ontology element	Metadata Creation	FR	Engineering	4.4.1 in D7.1.1 UC9 in D7.4.1	High
2.5.1.13	Annotation of the edited ontology element by means of free text (with multilingual support) and image association	Ontology Annotation	FR	Engineering	4.4.1 in D7.1.1	Medium

Table 6 Requirements for ontology editors (subject experts and validators): quality check.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.2.14.a	Duplication detection for pairs of concepts or instances over sensitive attributes	Ontology Evaluation	FR	Engineering	4.4.1 in D7.1.1 UC11 in D7.4.1	High
2.5.2.14.b	String similarity support for pair of literal values attributes	Ontology Evaluation	FR	Engineering	4.4.1 in D7.1.1 UC11 in D7.4.1	High
2.5.2.15	Support for the user to have two Ontologies at glance to visually compare them	Ontology Evaluation	NFR	GUI	D7.1.1 UC7 in D7.4.1	High

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.2. 16	Generate matrix of values for ontology structural properties	Ontology Evaluation	FR	Engineering	4.4.1 in D7.1.1 UC9 in D7.4.1	High

Table 7 Requirements for ontology editors (subject experts and validators): ontology population.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.3.17	Use text corpora to suggest candidate instances and relations between identified ones	Ontology Population and Restructuring	FR	Engineering	4.4.2 in D7.1.1 UC10 in D7.4.1	High
2.5.3.18	Selection of suggested instance to include in the ontology	Ontology Population	FR	GUI	4.4.2 in D7.1.1	High
2.5.3.19	Test ontology consistency before committing the changes	Ontology Restructuring	FR	Engineering	4.4.2 in D7.1.1	High

Table 8 Requirements for ontology editors (subject experts and validators): ontology mapping.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.4 20	Supervised or manual mapping of Ontologies	Ontology Alignment	NFR	Engineering	4.4.3 in D7.1.1 UC6 in D7.4.1	High

Table 9 Requirements for ontology editors (subject experts and validators): visualization.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.5 21	Simplified interface for editors	Visualization	NFR	GUI	4.4.4 in D7.1.1 UC7 in D7.4.1	High
2.5.5.22	Visualization of the ontology is customizable according the editorial activity; diagram-like, indented tree, node by node are required visualization types	Visualization	NFR	GUI	4.4.4 in D7.1.1	Medium
2.5.5.23	Instantiate a graphic environment for each ontology opened	Visualization	FR	GUI	4.4.4 in D7.1.1 UC7 in D7.4.1	Medium
2.5.5.24.a	Visualize mappings list for ontology pair	Visualization	FR	Distributed Component	4.4.4 in D7.1.1 UC7 in D7.4.1	High
2.5.5.24.b	Visualize overlapping ontology elements for ontology pair	Visualization	NFR	Distributed Component	4.4.4 in D7.1.1 UC7 in D7.4.1	High
2.5.5.25	Visualize ontology information inherent to editorial work: change log for an element, participating authors, frequency of changes	Visualization	FR	Distributed Component	4.4.4 in D7.1.1 UC7 in D7.4.1	High
2.5.5.26	Select one or more languages to visualize lexicalization of	Browsing and visualization	FR	GUI	4.4.4 in D7.1.1 UC7 in	High

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
	ontology elements				D7.4.1	
2.5.5.27	Support visualization for ontology population from text	Browsing and visualization	NFR	GUI	4.4.4 in D7.1.1 UC7 in D7.4.1	High
2.5.5.28	Print visualization	Visualization	FR	Engineering	4.4.4 in D7.1.1 UC7 in D7.4.1	Medium

Table 10 Workflow visualization.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.6	Visualization of ontology elements masked by Work Flow status	Workflow visualization	FR	Distributed Component	UC7 in D7.4.1	High

Table 11 Ontology export.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.7.29	Export in other formats	Data export	FR	Engineering	4.4.5 in D7.1.1 UC-4 in D7.4.1	High

Table 12 Search within ontology.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.8.30	Search in to the ontology repository	Search	FR	Distributed Component	UC1 in D7.4.1	
2.5.8.30	Search a loaded ontology	Search	FR	Engineering	UC1 in D7.4.1 (extension)	
2.5.8.30.30.a	Search for ontology elements based on semantic constraints	Search	FR	Engineering	4.4.6 in D7.1.1 UC1 in D7.4.1	High
2.5.8.30.30.b	Search for ontology elements using string comparison criteria	Search	FR	Engineering	UC1 in D7.4.1	High

Table 13 Multiliguality.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
2.5.9.31	Edit multilingual lexicalization	Multilinguality	FR	Engineering	4.4.7 in D7.1.1 UC3 in D7.4.1	High

Annex B – Summary of FSDAS requirements not part of lifecycle requirements

Tables in this annex traces the changes that requirements, scenarios and use cases have undergone from D7.1.1 until now.

Requirements are indicated with the same numbering system adopted in Annex A (i.e., the section number in them appear followed by the actual requirement number).

Table 14 Functional Requirements for FSDAS not elsewhere covered.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
3.4.1.3	Find an ontology	Ontology search	FR	Engineering	3.4.1 in D7.1.2	High
3.4.1.4	Annotate (including comments, keywords and quality scores) and rate concepts, attributes, relations and associated data instances.	Annotation	FR	Engineering	3.4.1 in D7.1.2	Medium
3.4.1.5	Rank results based on annotation data.	Ranking	FR	Engineering	3.4.1 in D7.1.2	Medium
3.4.1.6	Email results.	Email	FR	Engineering	3.4.1 in D7.1.2	Low
3.4.1.7	Suggest ontology modification.	Email modification suggestion	FR	Engineering	3.4.1 in D7.1.2	Low
3.4.1.8	Generate RSS feed from a query result.	Generate RSS feed.	FR	Engineering	3.4.1 in D7.1.2	Low
3.4.1.9	Change language of the interface.	Multi-lingual interface	FR	Engineering	3.4.1 in D7.1.2	High
3.4.2.4.a	Hide view of a class	Advanced view	FR	Engineering	3.4.2 in D7.1.2	Medium
3.4.2.4.b	Hide relation	Advanced view	FR	Engineering	3.4.2 in D7.1.2	Medium
3.4.2.4.c	Hide branch	Advanced view	FR	Engineering	3.4.2 in D7.1.2	Medium
3.4.2.6	Query for individual	Query	FR	Engineering	3.4.2 in D7.1.2	High

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
3.4.2.6.a	Query using Boolean logic, phrase matching and query refinement	Advanced query	FR	Engineering	3.4.2 in D7.1.2	High
3.4.3.1	View related data	View related individual	FR	Engineering	3.4.3 in D7.1.2	High
3.4.3.2	View related data text	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium
3.4.3.2.a	View related data text highlighted	View related individual	FR	Engineering	3.4.3 in D7.1.2	Low
3.4.3.2.b	View multiple related data text	View related individual	FR	Engineering	3.4.3 in D7.1.2	Low
3.4.3.2.c	View trends across related data	View related individual	FR	Engineering	3.4.3 in D7.1.2	Low
3.4.3.2.d	View related tabular data	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium
3.4.3.2.e	View compared related tabular data	View related individual	FR	Engineering	3.4.3 in D7.1.2	Low
3.4.3.3.a	View link to related data	View related individual	FR	Engineering	3.4.3 in D7.1.2	High
3.4.3.3.b	View related data extraction	View related individual	FR	Engineering	3.4.3 in D7.1.2	Low
3.4.3.4	View related data metadata	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium
3.4.3.4.a	View related data quality	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium
3.4.3.4.b	View related data provenance	View related individual	FR	Engineering	3.4.3 in D7.1.2	High
3.4.3.4.c	View related data modification	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium
3.4.3.4.d	View related data creation	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium
3.4.3.5	Group related data by concept and relation	View related individual	FR	Engineering	3.4.3 in D7.1.2	High

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
3.4.3.6	Re-group / Rank related data by concept and relation	View related individual	FR	Engineering	3.4.3 in D7.1.2	Medium

Table 15 Non-functional requirements for FSDAS not elsewhere covered.

Req. #	Functionality	Requirement extracted (= title)	Type	Layer of architecture	Traceability	Importance
3.7	Deploys using JWS	Deployment mode	NFR	Installer	3.7 in D7.1.2	Medium
3.8	Load OWL ontology	OWL support	NFR	Engineering	3.8 in D7.1.2	High
3.8	Read data from SQL queries, CSV files, XML data, HTML data, multimedia, and various proprietary formats, specifically Microsoft Office documents and Adobe PDF documents	Data import	NFR	Engineering	3.8 in D7.1.2	High
3.8 / 3.9	Retrieve data from Oracle and MySql relational databases, document repositories, ftp sites or web sites	Data import	NFR	Engineering	3.8 / 3.9 in D7.1.2	High
3.8	Crawl folder based systems such as ftp and web sites in order to determine and map true extent of contained resources.	Data discovery	NFR	Engineering	3.8 in D7.1.2	Low

Annex C – FSDAS Traceability tables

Tables in this annex trace the changes (modifications /refinements /additions /deletion /extensions /generalizations) with respect to the changes presented in D7.1.1 and D7.5.1.

Table 16 Revised requirements.

D7.1.1 - Reqs.	D7.5.1 - Architecture	D7.1.2 - Updated Reqs.	Was	Changed to	Change type	Req. type
all	all	FSDAS features	Alert	Assessment	modification	n/a
3 - OVERVIEW OF THE ONTOLOGY-BASED FSDAS	2.2.3 Visualization components set	FSDAS features	web-based	java web-start-based	modification	NF
3 - OVERVIEW OF THE ONTOLOGY-BASED FSDAS	2.1 Design architecture overview	FSDAS features	The ontology-driven FSDAS will comprise the FSDAS user interface, a network of ontologies, and a number of web services extending the NeOn toolkit infrastructure.	The ontology-driven FSDAS will comprise on the client-side the FSDAS user interface, and on the server-side a network of ontologies, and a number of supporting services extending the NeOn toolkit infrastructure.	refinement	NF
3.1.3 Services and API	2.1 Design architecture overview	FSDAS features	The Services and API relevant for the alert system are: concept service, inference service and data warehouse. They are described in detail in Section 4.7.2.	null	deletion	n/a
3.2 Constraints	n/a	FSDAS features	The minimum set of requirements for web interfaces are provided in Annex II: “WAICENT Checklist for the clearance of web sites”	null	deletion	NF

D7.1.1 - Reqs.	D7.5.1 - Architecture	D7.1.2 - Updated Reqs.	Was	Changed to	Change type	Req. type
3.2 Constraints	3.1 Architecturally significant non-functional software requirements	FSDAS Operating environment	null	Access to underlying databases containing data instances must be from the server not the client due to firewall-based access restrictions.	addition	NF
5 - REQUIREMENTS FOR THE FISHERIES STOCK DEPLETION ALERT SYSTEM	2.2.3 Visualization components set	FSDAS User Interfaces	web-based	java web-start-based	modification	NF
5 - REQUIREMENTS FOR THE FISHERIES STOCK DEPLETION ALERT SYSTEM	2.1 Design architecture overview	FSDAS features	The ontology-driven FSDAS will comprise the FSDAS user interface, a network of ontologies, and a number of web services extending the NeOn toolkit infrastructure.	The ontology-driven FSDAS will comprise on the client-side the FSDAS user interface, and on the server-side a network of ontologies, and a number of supporting services extending the NeOn toolkit infrastructure.	refinement	n/a
5.2.1 - Users	4.2 Use-Case Realizations	FSDAS User types and characteristics	FSDAS users will be mainly officers in FAO Fisheries Department (including Fishery Policy and Planning Division, Fisheries Resources Division, Fishery industry Division, as well as the Fishery Information, Data and Statistics Unit) attached to FAO headquarters and regional and sub-regional offices around the world.	FSDAS users will be officers in FAO Fisheries Department attached to FAO headquarters and regional and sub-regional offices around the world, as well as Regional Fisheries Bodies and national-level fisheries scientists.	extension	n/a
5.2.2 – Requirements	N/A	FSDAS Function Summary	See relevant section.	See relevant section. Functional requirements have been reiterated, refined and extended.	refinement, extension	F
5.3.1 - User Scenario	5.4.2.2 User Profile Manager	FSDAS Software Interfaces	logs in with his FAODOMAIN user name and password	logs in with his user name and password	generalization	F

D7.1.1 - Reqs.	D7.5.1 - Architecture	D7.1.2 - Updated Reqs.	Was	Changed to	Change type	Req. type
5.3.1 - User Scenario	5.4.3.3 Communication Manager	n/a	Jose selects “e-mail results”. The system opens a web-browser with a web-mailer page pre-filled with Jose’s email address and the applicable concepts in the subject area.	Jose selects “e-mail results”. The system opens his default email client, i.e. system application registered to the mailto: protocol. pre-filled with Jose’s email address and the applicable concepts in the subject area.	modification	NF
5.3.3 - User Scenario	5.4.5 Rank Components	n/a	The selected documents are presented to Ms Lopez in order of likely relevance, and the salient words occurring in them are highlighted	Links to the selected documents are presented to Ms Lopez in order of likely relevance, and the salient words occurring in them are displayed	refinement	F
5.3.3 - User Scenario	5.4.1.2 Ontological Resource Display Manager	n/a	and of the visual cues used for in the document highlighting	and the visual cues provided by the salient words	refinement	F
5.5.1.1 - User interface	2.2.3 Visualization components set	FSDAS User Interfaces	The FSDAS user interface will be web-based and compatible with Microsoft Internet Explorer 6.0+ and Mozilla Firefox 1.5+. Given the large amount of data to be loaded and managed, the use of AJAX technology would need to be explored.	The FSDAS user interface will utilize java web-start and shall be compatible with Windows XP, Windows Vista, Mac OSX and Linux Ubuntu.	modification	NF
5.5.1.1 - User interface	2.2.3 Visualization components set	FSDAS User Interfaces	In addition, the user interface must be compliant with the WAICENT Checklist for the clearance of web sites provided in Annex II.	null	deletion	n/a
5.5.1.2 - Software interfaces	10.2 Implementation Constraints	FSDAS Software Interfaces	The software will need to interface with the NeOn infrastructure components and with ORACLE DBMS.	The server-side software will need to interface with the NeOn infrastructure components and with ORACLE and/or MySQL DBMS.	generalization	NF

D7.1.1 - Reqs.	D7.5.1 - Architecture	D7.1.2 - Updated Reqs.	Was	Changed to	Change type	Req. type
5.5.2 - Operating environment	10.1 Operating Environment	FSDAS Operating environment	Client must function on Windows/Linux/Mac OS-X operating systems; e.g. in a browser.	Client must function on Windows/Linux/Mac OS-X operating systems.	generalization	NF
5.5.2 - Operating environment	5.4.2.1 User Account Manager	FSDAS Software Interfaces	The authentication service should be compatible with current FAO standard to allow FAO users to login with their network credentials.	null	deletion	n/a
5.5.3 - Design and Implementation Constraints	10.2 Implementation Constraints	FSDAS Operating environment	Memory requirements should be below one gigabyte.	Server memory requirements shall be below four gigabytes.	refinement	NF
5.5.3 - Design and Implementation Constraints	10.2 Implementation Constraints	FSDAS Software Interfaces	Application should be able to use ORACLE as its DBMS	The server-side software will need to be able to use ORACLE and/or MySQL as its DBMS.	generalization	NF
5.5.3 - Design and Implementation Constraints	5.4.2.2 User Profile Manager	FSDAS Software Interfaces	Application user security must be a separate component allowing for the use of FAO's current user authentication system (FAODOMAIN).	null	deletion	n/a

Annex D – FSDAS revised use cases

Tables in this annex list the use cases presented in D7.1.1 and revised in this deliverable.

Table 17 Revised use cases.

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.1	UC1	Login	Login	Basic flow	null	System loads user rights	extension	High
5.4.2	UC2	Logout	Logout	Basic flow	System asks if user wants to save non-saved data she is working on	System asks if user wants to save session; System updates user profile	refinement	High
5.4.2	UC2	Logout	Logout	Alt flow	User not found, is invited to try again or register	null	deletion	High
5.4.2	UC2	Logout	Logout	Related UC	null	Save Session	addition	High
5.4.3	UC3	Register	Register	Post condition	Registration request is sent to admin	Creation of generic user; Registration request is sent to admin for role specification; System displays a success page to the user	refinement	High
5.4.3	UC3	Register	Register	Basic flow	null	User inputs profile: user type, organization, ontologies preferred language, etc.	extension	High
5.4.3	UC3	Register	Register	Notes	Registration is sent for approval by system administrator	Registration is sent for action by system administrator	generalisation	High
5.4.4	UC4	Modify profile	Modify profile	Basic flow	User modifies user details, including default language, ontologies and favourite concepts	User modifies profiles details	generalisation	Medium

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.4	UC4	Modify profile	Modify profile	Notes	The user profile contains information about the user (name, organization, email, editing rights, ...) and all preferred options including list of ontologies and collections of documents and statistics commonly used for search, browsing and reading.	The user profile contains information about the user preferred options including list of ontologies and collections of documents and statistics commonly used for search, browsing and reading.	generalisation	Medium
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Description	Modify the set of ontologies listed in the user profile	User modifies his account.	generalisation	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Precondition	User profile is open for editing	null	deletion	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Trigger	User selects <modify ontology set>	User clicks <modify account> from main page	modification	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Post condition	Set of ontologies is modified	Account is modified; Registration request is sent to admin for role specification; User is shown main page	modification	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Basic flow	User selects <modify ontology set>; System shows the currently loaded ontologies and all available unloaded ontologies; User modifies the set of loaded ontologies; System stores the modified set	User clicks <modify account> from main page; System loads account page; User modifies account details; User validates changes; Modify request is sent to the system administrator with details; System returns to main page	modification	High

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Alt flow	null	User abandons action by clicking <cancel>; System returns to main page, no changes are made	addition	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Alt flow	null	User makes some kind of invalid input; System invites user to try again	addition	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Related UC	Modify profile	null	deletion	High
5.4.5	UC5	Modify the set of ontologies in profile	Modify User Account	Notes	All ontologies in the network are "available ontologies"	The user account contains information about the user (name, organization, email, editing rights, ...)	modification	High
5.4.6	UC6	Search concept in ontology	Search ontological resource in ontology	All headings	concepts	ontological resources	name	High
5.4.6	UC6	Search concept in ontology	Search ontological resource in ontology	Precondition	null	At least one ontology is loaded	addition	High
5.4.6	UC6	Search concept in ontology	Search ontological resource in ontology	Basic flow	System consults **ontology server**	System consults **loaded ontologies**	name	High
5.4.7	UC7	Search for related concepts	Search for related ontological resources	All headings	concepts	ontological resources	name	High
5.4.7	UC7	Search for related concepts	Search for related ontological resources	Basic flow	User selects a concept; User clicks <search for related concepts>; System queries ontology server; Related concepts are displayed	User selects a ontological resource; User specifies relation(s); User selects direct/indirect relation (reasoning); User clicks <search for related ontological resources>; System queries loaded ontologies; Related ontological resources are displayed	extension	High
5.4.8	UC8	Browse concepts	Browse taxonomy	Precondition	null	Taxonomy panel is visible	addition	High

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.9	UC9	Change language of the interface	Change Language of the interface	Basic flow	null	System asks the user whether to extend the choice to ontological resources	addition	High
5.4.9	UC9	Change language of the interface	Change Language of the interface	Related UC	null	Change language of the ontological resource shown	addition	High
5.4.9	UC9	Change language of the interface	Change Language of the interface	Notes	null	Object annotation should also switch if users selected to extend the choice to ontological resources	addition	High
5.4.10	UC10	Change language of the ontological resource shown	Change language of the ontological resource shown	n/a	n/a	n/a	n/a	High
5.4.11	UC11	Add Concept to query	Query composition	All headings	concepts	ontological resources	name	High
5.4.11	UC11	Add Concept to query	Query composition	Related UC	Browse concept	Browse Taxonomy	name	High
5.4.12	UC12	Query Concept instances	Query for Data related to individual	Precondition	null	User composed a query	addition	High
5.4.12	UC12	Query Concept instances	Query for Data related to individual	All headings	concept instance	Data source	name	High
5.4.12	UC12	Query Concept instances	Query for Data related to individual	Basic flow	null	System considers user tags (local/remote)	addition	High
5.4.12	UC12	Query Concept instances	Query for Data related to individual	Basic flow	System displays related concepts for which it has concept instances, if any	System displays related cluster of documents, if any	modification	High

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.12	UC12	Query Concept instances	Query for Data related to individual	Basic flow	It is considered that <concept instances> are actual links to data resources associated to concept classes, e.g. a link to a <Yellow fin Tuna fact sheet> that is linked to the concept <Yellow fin Tuna>.	null	deletion	High
5.4.13	UC13	Visualize Concept instance	Visualize Data Source related to individual	All headings	concept	Data source	name	High
5.4.14	UC14	Refine Query	Refine Query	Precondition	At least one concept instance is listed in the query results	At least one query has been performed	generalisation	High
5.4.15	UC15	View concept metadata	View ontological resource annotation	All headings	concept metadata	ontological resource annotations	generalisation	High
5.4.15	UC15	View concept metadata	View ontological resource annotation	All headings	concept	ontological resource	generalisation	High
5.4.15	UC15	View concept metadata	View ontological resource annotation	Basic flow	System queries **ontology server** and displays concept together with associated metadata	System queries **loaded ontologies** and displays ontological resource together with associated annotations	name	High
5.4.16	UC16	View concept instance metadata	View Data Source Annotation	All headings	concept instance metadata	Data Source annotation	generalisation	Low
5.4.16	UC16	View concept instance metadata	View Data Source Annotation	All headings	concept instance	Data Source	generalisation	Low
5.4.16	UC16	View concept instance metadata	View Data Source Annotation	Notes	The selected instance can be the result of a query already performed	null	deletion	Low

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.17	UC17	Email results	Email results	Precondition	At least one concept or concept instance is displayed	At least one Data Source link is displayed	refinement	Low
5.4.18	UC18	Propose ontology modification	Propose ontology modification	All headings	ontology editors	ontology editors/reviewers	extension	Medium
5.4.18	UC18	Propose ontology modification	Propose ontology modification	Precondition	null	At least one ontology is loaded	addition	Medium
5.4.19	UC19	Add to favourites	Add to favourites	n/a	n/a	n/a	n/a	Medium
5.4.20	UC20	Save session	Save session	Precondition	A search result page	User is logged in	modification	Medium
5.4.20	UC20	Save session	Save session	Trigger	null	User Logs out	addition	Medium
5.4.21	UC21	Generate RSS feed from current query	Generate RSS feed from current query	n/a	n/a	n/a	n/a	Low
5.4.22	UC22	Annotate retrieved document with comments on quality	Index enrichment with quality ratings	Precondition	User is logged in with appropriate rights	User is logged in	generalisation	Low
5.4.22	UC22	Annotate retrieved document with comments on quality	Index enrichment with quality ratings	Precondition	null	User has annotation rights	addition	Low
5.4.23	UC23	Annotate retrieved document with keywords	Index enrichment with keywords	Precondition	null	User has annotation rights	addition	Low
5.4.24	UC24	Annotate retrieved document with comments	Index enrichment with comments	Alt flow	user writes comment concerning part of the document -- User selects an excerpt from the document -- User selects <add your comment> -- Process continues as above	null	deletion	Low

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.24	UC24	Annotate retrieved document with comments	Index enrichment with comments	Description	After reading a retrieved document, user annotates it (or part of it) with keywords	After reading a retrieved document, user annotates it	generalisation	Low
5.4.24	UC24	Annotate retrieved document with comments	Index enrichment with comments	Related UC	null	Index enrichment with quality ratings	addition	Low
5.4.24	UC24	Annotate retrieved document with comments	Index enrichment with comments	Notes	The possibility of adding keywords not taken from any resources could be discussed; The possibility of letting the system suggest which keywords to assign to the document should al	null	deletion	Low
5.4.25	UC25	Select ontologies to use for browsing documents or web pages	Select ontologies to use for browsing documents or web pages	Notes	null	(MAGPIE FUNCTIONALITY)	addition	Low
5.4.26	UC26	Use ontology to support browsing	Use ontology to support browsing	Notes	null	(MAGPIE FUNCTIONALITY)	addition	Low
5.4.27	UC27	Identify trend	Identify trend	Precondition	null	At least one concept instance has to be selected	addition	Low
5.4.28	UC28	Compare data by reporter	Compare data by reporter	Precondition	null	At least two data source results have been returned	addition	Low

D7.1.1	D7.5.1	Former name	Current name	Type	Was	Change	Change type	Priority
5.4.28	UC28	Compare data by reporter	Compare data by reporter	Basic flow	User selects a domain (e.g. capture, production, commodities, fleet, ...); User selects a species and/or a geographical area; User click <search>; System returns a list of results; User selects <order by data source>; User selects two items from the list and clicks <compare>; System opens the two selected items, and shows them side by side	User makes a query that returns a result set of at least two documents; User selects two items from the list and clicks <compare>; System opens the two selected items, and shows them side by side	refinement	Low
5.4.29	UC29	Compare historical data	Compare historical data	Basic flow	User selects a domain (e.g. capture, production, commodities, fleet, ...); User selects a species and/or a geographical area; User click <search>; System returns a list of results; User selects <order by data source>; User selects two items from the list and clicks <compare>; System opens the two selected items, and shows them side by side	User makes a query that returns a result set of at least two documents; User selects two items from the list and clicks <compare>; System opens the two selected items, and shows them side by side	refinement	Low

Annex E – FSDAS new use cases

This annex lists new use cases with respect to D7.1.1.

UC29 Discover Ontology

ACTORS

- All

DESCRIPTION

- User searches for ontology

PRECONDITIONS

- User is logged in

TRIGGERING EVENT(s)

- User clicks “find ontologies”

POST CONDITIONS

- One or more ontologies have been loaded.

FLOW OF EVENTS

a. BASIC FLOW

- User makes an ontology discovery query that returns a result set from the registry including metadata.
- User selects one or more ontologies and clicks “load”.
- System opens the ontologies focused on the concepts that were related to the user's ontology discovery query.

RELATED USE CASES

- N/a

NOTES / ISSUES

UC30 Filter Ontology

ACTORS

- All

DESCRIPTION

- User filters an ontology to view only the parts that interest them.

PRECONDITIONS

- User is logged in
- At least one ontology is loaded

TRIGGERING EVENT(s)

- User clicks “filter ontology”

POST CONDITIONS

- Ontology has been filtered.

FLOW OF EVENTS

a. BASIC FLOW

- User clicks “filter ontology”
- System displays “class”, “relationship” or “branch”.
- User selects “class”, “relationship” or “branch”.
- System displays ontology model (if class or relationship).
- User selects part of model or branch to filter and clicks “filter”
- Ontology is re-displayed minus filtered part of model / ontology

RELATED USE CASES

- N/a

NOTES / ISSUES

UC31 View data instance summary

ACTORS

- All

DESCRIPTION

- User views a summary extraction from a data instance (document file) related to an ontological concept.

PRECONDITIONS

- User is logged in
- At least one ontology is loaded
- At least one data instance is mapped to the ontology
- User has made a query that has returned at least one data instance

TRIGGERING EVENT(s)

- User selects a data instance and clicks “view summary”

POST CONDITIONS

- Summary is displayed.

FLOW OF EVENTS

a. BASIC FLOW

- User selects a data resource and clicks “view summary”
- Summary is displayed.

RELATED USE CASES

- N/a

NOTES / ISSUES

UC32 Re-group / rank data resource by annotation

ACTORS

- All

DESCRIPTION

- User re-groups or re-ranks a set of data resource related to an ontological concept by their annotation data.

PRECONDITIONS

- User is logged in
- At least one ontology is loaded
- At least one data resource is mapped to the ontology
- User has made a query that has returned a set of data resources

TRIGGERING EVENT(s)

- User selects “re-group”

POST CONDITIONS

- Results are displayed re-grouped.

FLOW OF EVENTS

a. BASIC FLOW

- User selects a “re-group”
- System displays a list of possibilities; e.g. date, provenance, quality, keyword
- User selects a ranking parameter
- System re-groups results.

RELATED USE CASES

- N/a

NOTES / ISSUES