Chapter 7
Ontology Development by Reuse

Mariano Fernández-López, Mari Carmen Suárez-Figueroa, and Asunción Gómez-Pérez

Abstract This chapter presents methodological guidelines that allow engineers to reuse generic ontologies. This kind of ontologies represents notions generic across many fields, (*is part of*, *temporal interval*, etc.). The guidelines helps the developer (a) to identify the type of generic ontology to be reused, (b) to find out the axioms and definitions that should be reused and (c) to adapt and integrate the generic ontology selected in the domain ontology to be developed. For each task of the methodology, a set of heuristics with examples are presented. We hope that after reading this chapter, you would have acquired some basic ideas on how to take advantage of the great deal of well-founded explicit knowledge that formalizes generic notions such as time concepts and the *part of* relation.

7.1 Introduction

Ontologies play an important role in many knowledge-intensive applications, by formally defining the conceptualization used by the application and by facilitating interoperability. Building ontologies from scratch can in general be expensive. In this sense, one way of reducing the time and costs associated with the ontology development process is by reusing available ontological resources. Ontologies developed by reuse can also build on existing good practices (from well-developed ontologies), thus increasing the overall quality of the results.

M. Fernández-López (✉)
Escuela Politécnica Superior, Universidad San Pablo CEU, Urbanización Montepríncipe sn., 28668 Boadilla del Monte, Madrid, Spain
e-mail: mfernandez.eps@ceu.es

M.C. Suárez-Figueroa • A. Gómez-Pérez
Ontology Engineering Group, Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo sn., 28660 Boadilla del Monte, Madrid, Spain
e-mail: mcsuarez@fi.upm.es; asun@fi.upm.es

M.C. Suárez-Figueroa et al. (eds.), *Ontology Engineering in a Networked World*, DOI 10.1007/978-3-642-24794-1_7, © Springer-Verlag Berlin Heidelberg 2012

As mentioned in Chap. 2, the NeOn Methodology presents nine scenarios for building networks of ontologies. One of these scenarios is *Building Ontology Networks by Reusing Ontological Resources*. In this scenario, ontology developers analyze whether existing ontological resources can be reused in the context of building an ontology.

The reuse of ontological resources is encouraged by a recent increase in the number of ontologies available online.

According to our experience, the reuse of ontological resources is useful for (a) saving time and resources during the ontology development and (b) refining the Ontology Requirement Specification Document (ORSD) (see Chap. 5) taking into account the knowledge represented in the candidate ontological resources to be reused. The latter case refers to the situation in which the engineer finds axioms and/or definitions of terms that did not appear in the ORSD. For example, in the development of a drug ontology, the engineer may find a type of drug that had not been considered in the ORSD. For the sake of simplicity, in this chapter, it is assumed that the reuse does not imply modifications in the ORSD. If such modifications are required, an iterative-incremental life cycle model should be followed (see Chap. 2).

The ontological resource reuse process is often influenced by the type of ontology to be reused. Ontologies can model domain entities (e.g., drug, disease, pharmaceutical product) or generic entities, which are considered to be generic across many fields (van Heijst et al. 1997). For example, the *part of* relation can be used to link objects in the mechanical domain (the spark plug is part of the motor) and also in the domain of cultural activities (the interpretation of Radetzky March is part of the New Year Concert). Hence, such generic ontologies can be reused in a wider range of domains.

However, the reuse of large ontologies such as WordNet[1] or the NCI ontology (Golbeck et al. 2003) can cause difficulties because they tend to contain far more definitions than most applications would normally need. Hence, in the context of a reuse process, sometimes elements of an ontology (e.g., modules or statements) have to be extracted first, to be integrated in the new ontology (d'Aquin M et al. 2007b). For this reason, different levels of granularity in the reuse of ontologies can be distinguished:

- Ontologies can be reused as a whole if they closely meet the expectations and the needs of the ontology engineer.
- In certain cases, only one part or *module*[2] of an ontology is relevant for reuse. For example, when building an ontology about lung cancer, it is not always necessary to reuse an entire ontology about the human body; it suffices to reuse a module describing concepts related to the lung.

[1] http://wordnet.princeton.edu/

[2] We consider a module (d'Aquin M et al. 2007b) as a part of the ontology that defines the relevant set of terms for a particular purpose.

- In other cases, only some knowledge components from the ontology (the description of a particular entity, the branch in the taxonomic hierarchy in which an entity appears, or entity neighborhoods in the ontology) are relevant for the development needs. In these cases, the reuse of ontological knowledge is performed at the *statement*[3] level, providing the ontology developer with better control over the material being reused.

This chapter focuses on providing methodological guidelines for the reuse of generic ontologies, although most of the recommendations are also applicable to the reuse of domain ontologies.

7.2 Methodological Guidelines for Reusing Generic Ontologies

Table 7.1 presents a filling card with the information concerning the generic ontology reuse process. The card includes the definition, the goal, the inputs and outputs, the performer of the process, and the time scheduled for the process.

Figure 7.1 shows the workflow and the activities for carrying out the generic ontology reuse process, that is, selecting the ontology to be reused, and customizing and integrating it in the ontology to be developed.

The activities shown in Fig. 7.1 are explained in more detail in the rest of the chapter. For the sake of simplicity, the different activities involved in the whole process are explained, and it is considered the reuse of just one ontology. When reusing more than one ontology, the process described should be performed iteratively.

Along the exposition, an example of reusing a generic ontology in the development of the pharmaceutical product ontology network (PPO) (see Chap. 20) is presented. This ontology will be used as a bridge between proprietary systems for managing financial and product knowledge interoperability in pharmaceutical laboratories, companies, and distributors in Spain. In this ontology reuse task, we have taken into account the four *competency questions* (CQs) shown in Table 7.2. They have been obtained from Chap. 7 of the NeOn deliverable D5.4.1 (Suárez-Figueroa et al. 2008).

The reader can find additional information on ontology reuse in (Suárez-Figueroa 2010).

[3] An ontology statement (or triple) contains the following three components: *subject*, *predicate*, and *object*.

Table 7.1 Generic ontology reuse filling card

Generic Ontology Reuse	
Definition	
Generic OntologyReuse refers to the process of using general ontologies in the solution of different problems.	
Goal	
The goal of this process is to find and select generic ontologies and integrate them in the ontology network being developed.	
Input	*Output*
Competency questions (CQs) included in the ORSD of the ontology network to be developed and the implementation language of such ontology. Optionally, there may be a set of tables that compare with the same criteria the candidate ontologies to be reused.	A generic ontology integrated in the ontology network being developed.
Who	
Software developers and ontology practitioners involved in the ontology development. It may be required the help of an ontology practitioner familiarized in formal ontologies or theories.	
When	
The generic ontology reuse process should be carried out after the ontology specification activity.	

7.2.1 Activity 1: Selecting the Generic Ontology to be Reused

The goal of this activity is to select the most appropriate generic ontologies to be reused in the ontology being developed. It is worth mentioning that instead of reusing available ontologies, practitioners can implement from scratch the necessary axioms and definitions according to some existing formalization, for example, the one appearing in Annex. On the one hand, the advantage of reusing available ontologies implemented in a formal language is that ontology developers will save effort in the transformation of a formalization that is not suitable for run-time reasoning. On the other hand, the advantage of starting from an existing formalization is that ontology developers will save effort in the searching, comparison, and evaluation of candidate ontologies to be reused. In this chapter, we focus on the reuse option.

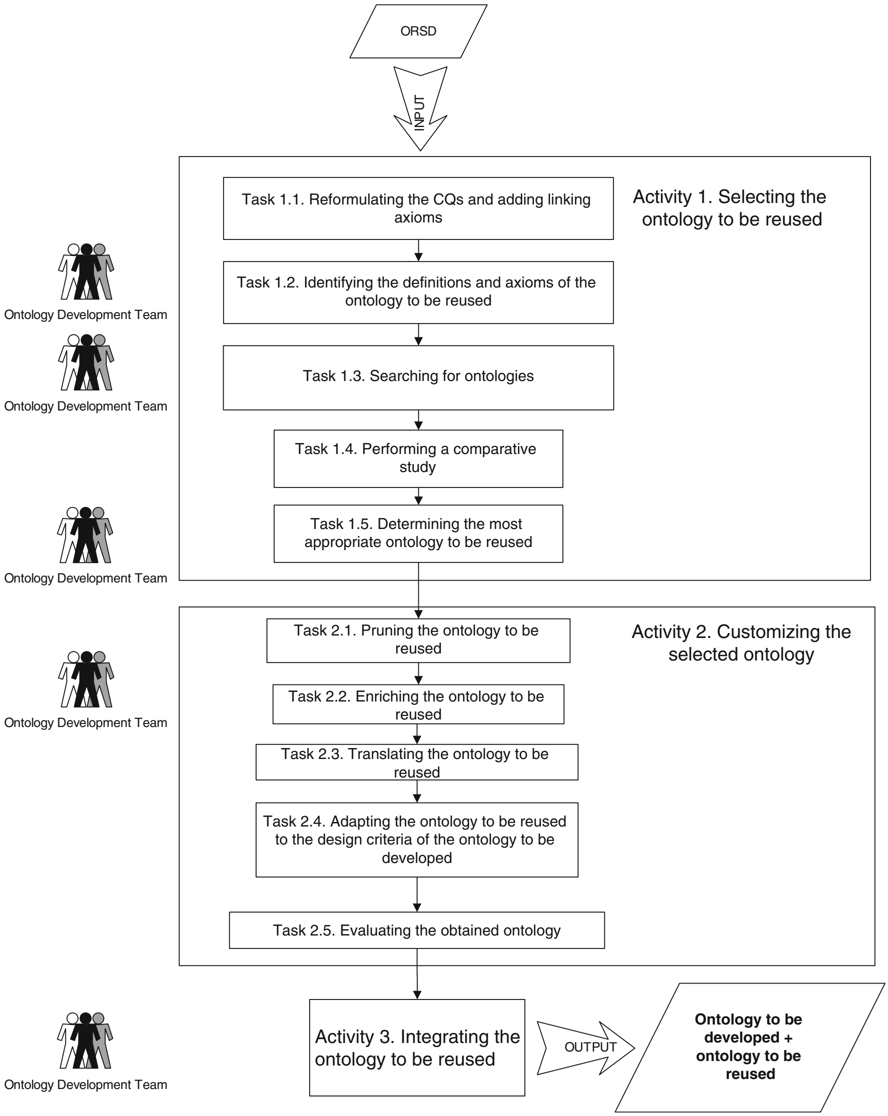

Fig. 7.1 Activities for reusing generic ontologies

This activity takes as input the ORSD (Chap. 5) and is divided in the following tasks:

Task 1.1 Reformulating the CQs and adding linking axioms. The main goal of this task is to reformulate the CQs included in the ORSD of the ontology that is being developed with vocabulary that could potentially belong to ontologies to be reused but that do not explicitly appear in the CQs. Additionally, another goal of

Table 7.2 Excerpt of informal host competency questions (pharmaceutical product ontology case)

CQ id	Informal CQ	Example of answer
CQ_1	What drugs do have paracetamol?	Algidol® Apiretal® Bisolgrip® Cortafriol® Dolgesic® Dolostop® Efferalgan® Frenadol® Gelocatil® Pharmagrip® Termalgin®
CQ_2	Which is the composition of Frenadol®?	Caffeine Chlorpheniramine citrate Dextrometorphan Paracetamol
CQ_3	Which is the main active ingredient of Frenadol®?	Paracetamol
CQ_4	Which substances do Frenadol® interacts with?	Ethyl alcohol Isoniazid Propranolol Rifampicin

this task is to identify axioms that link terms of the CQs to terms that could be reused. The first column of Table 7.3 shows some typical cases (case 1, case 2, and case 3[4]) that guide the engineer in transforming CQs and adding linking axioms. The third column shows the action to carry out in each case. Finally, as an example, the second and fourth columns present the PPO CQ that matches each case and the result of applying the action corresponding to the case. For example, given that the case 2 (Table 7.3) proposes to reformulate CQs using the term *is part of*, the CQ_1 (Table 7.2), *what drugs do have paracetamol?*, can be expressed as *which drugs is paracetamol part of?* Given that the term *is part of* appears in the new formulation, the engineer knows that a mereology can be reused (see Annex to review basic mereology notions).

This task is useful to make explicit abstract terms such as *is part of*, *temporal point*, and *temporal interval* that can be reused from mereological or time ontologies.

Task 1.2 Identifying the definitions and axioms of the ontology to be reused. The goal here is to identify which definitions and axioms can be potentially reused in the ontology to be developed. The terms whose definition could be reusable from other ontologies are those terms appearing in the pre-glossary of the ORSD (specifically in slot 7) (see Chap. 5) and the new terms that appear in the reformulated CQs

[4] The rest of the cases are presented in Suárez-Figueroa (2010).

Table 7.3 Analysis and transformation of the competency questions and addition of linking axioms and rules (Task 1.1)

Case	Competency question	Action to carry out	Result of the action
Case 1. Ontology developers are interested in knowing the parts of an object without including the object itself	CQ_2. Which is the composition of Frenadol®?	Reformulate the CQ to mention the term *is proper part of*	Which are the proper parts of Frenadol®?
Case 2. Ontology developers are interested in knowing the parts of an object including the object itself	CQ_1. What drugs do have paracetamol? (The inclusion of the substance itself is because paracetamol itself could be a drug)	Reformulate the CQ to mention the term *is part of*	Which drugs is paracetamol part of?
	CQ_4. Which substances do Frenadol® interacts with?		Which substances do the parts of Frenadol® interacts with?
Case 3. The CQ refers to a relation *S* that is subrelation of *isPartOf*	CQ_3. Which is the main active ingredient of Frenadol®?	Introduce a linking axiom establishing that *S* is subrelation of *is part of*	Introduce, when the mereology implementation is reused, the following axiom: *Is main active ingredient of* is a subrelation of *is part of*?

obtained in Task 1.1. The second column of Table 7.4 presents some heuristics that are useful to find mereology definitions and axioms that could be potentially reused (for the rest of the heuristics see footnote 4). Such a table shows that the properties of *is part of* that are useful for PPO are reflexivity, antisymmetry, transitivity, and the weak supplementation principle. For example, if ontology developers are interested in knowing what substances contains a particular substance (e.g., iron), they need to apply transitivity, since the substance in question can be an indirect part of the drug. For instance, iron is part of ferrous sulfite, and this is, in its turn, part of Mol Iron®, which is a drug. Moreover, the definition of the term *is proper part of* should be reused to answer questions like CQ_2, where the interest is not located in the drug itself.

Task 1.3 Search for ontologies. The ontology development team should search for ontologies that implement the axioms and definitions identified in Task 1.2.

To perform this task, ontology developers can use a general purpose search engine (e.g., Google[5]), Semantic Web search engines (e.g., Swoogle[6], Watson[7],

[5] http://www.google.es/

[6] http://swoogle.umbc.edu/

[7] http://watson.kmi.open.ac.uk

Table 7.4 Identification of definitions and axioms to reuse from a mereology (Task 1.2)

Axioms and definitions	When they are useful	The condition is fulfilled
A.1. *Is part of* reflexivity	Recommended if its implementation is possible, to ensure the right meaning of *part of*	*Yes*
A.2. *Is part of* antisymmetry	Recommended if its implementation is possible, for consistency verification	*Yes*
A.3. *Is part of* transitivity	X has parts $X_1, X_2, \ldots, X_n$. In its turn, there is some X_i with parts $X_{i1}, X_{i2}, \ldots, X_{im}$. That is, X has several levels of parts. Besides, ontology developers are interested in all the levels when they ask: *which are the parts of X?*	*Yes*
D.1. *Is proper part of* definition	The case 1 (see Table 7.3) is fulfilled	*Yes*
A.4. *Is part of* weak supplementation	Recommended if its implementation is possible, for consistency verification	*Yes*

Sindice[8], Sigma[9]), repositories (e.g., the Protégé ontology library[10], the Open Biological and Biomedical Ontologies[11], and Cupboard[12]), and other known ontologies (for instance, mereology terms can be reused from Dolce-Lite[13], SUMO-OWL[14], etc.).

For example, Watson is a Semantic Web search engine developed as part of the NeOn project which provides features to search, select, and integrate ontologies available online (d'Aquin and Motta 2011). Watson collects, indexes, and provides access to ontologies crawled from the web. From a user interface perspective, it can be seen as a classical search engine, taking as input keywords (e.g., based on the ORSD) and providing as a result a list of ontologies that match these keywords, together with information about each ontology, and about entities in them that are relevant to the given keywords. Ontologies and entities can be further explored online, using the provided navigation features. As part of its indexing process, Watson also extracts information about each ontology, such as the underlying language, its size, and metadata that the corresponding file might include. Search results from Watson often include thousands of ontologies. They can be further reduced by using filters (search options) regarding the scope of the search (in local names, labels, comments, or any other literal of an entity), the type of entities to consider (classes, properties, or individuals), and how strict the match should be. In addition, developers' background knowledge helps in the filtering.

[8] http://sindice.com/

[9] http://sig.ma/

[10] http://protege.cim3.net/cgi-bin/wiki.pl?ProtegeOntologiesLibrary

[11] http://www.obofoundry.org/

[12] http://cupboard.open.ac.uk

[13] http://www.loa-cnr.it/ontologies/DOLCE-Lite.owl

[14] http://www.ontologyportal.org/translations/SUMO.owl.txt

In addition to its user interface, Watson includes a set of open APIs making it possible for application developers to find and exploit online ontologies directly from the provided infrastructure. This API has been used to create an interface to Watson from the NeOn Toolkit, where definitions of specific classes and properties can be found and reuse: the Watson plugin (d'Aquin et al. 2008). Using the Watson plugin, an initial "skeleton" model can be defined as a basis for searching relevant definitions from online ontologies. Selecting a concept or a property, the user can obtain list of statements that corresponds to alternative representations of this class and properties, and directly integrate such representations (partially or completely) in the ontology under development.

In addition to the Watson plugin, other developments have been integrated with Watson with the goal of facilitating ontology search and reuse. For example, in addition to extracted information, Watson provides a simple visual summary of each ontology using the key concept extraction mechanism described in (Peroni et al. 2008). Mechanisms such as visual summaries and the Watson indexing process were also reused to create Cupboard (d'Aquin and Lewen 2009), an ontology repository system, where users can publish ontologies and search them in a way similar to Watson.

As an example, Table 7.5 presents some ontologies that define mereological relations.

Task 1.4 Performing a comparative study. The goal here is to compare the candidate ontologies obtained in Task 1.3 with the axioms and definitions identified in Task 1.2. This comparative study is represented in the form of a table to facilitate its use. In the table, each row represents the set of definitions (or axioms) identified in Task 1.2, and each column, the ontologies found in Task 1.3.

As an example, a comparative table of ontologies implementing mereologies is shown in Table 7.6. The symbol "X" means that the feature is represented in the ontology. In the example, the definitions of *underlap* and *disjoint*, and the weak supplementation principle are formalized in formal mereologies (see Annex), but they do not appear in any of the OWL ontologies that appear in the table.

Task 1.5 Determining the most appropriate ontology to be reused. The goal of this task is to determine which of the candidate ontologies identified in Task 1.3 is the most appropriate to be reused in the ontology being developed. To determine such an ontology, the analysis following Fig. 7.2 is carried out.

Table 7.5 Mereology implementations (Task 1.3)

Found mereology implementations	*Project or institution*
Single part whole[a]	*W3C*
SUMO-OWL	*IEEE Standard Upper Ontology working group*
Dolce-Lite	*Italian Research Council (CNR)*
Oswebsite[b]	*OS Open data*
OBO	*Open Biological and Biomedical Ontologies*[c]

[a]http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.owl
[b]http://www.ordnancesurvey.co.uk/oswebsite/ontology/Mereology.owl
[c]http://www.berkeleybop.org/ontologies/obo-all/relationship/relationship.owl

Table 7.6 Comparative study of mereology ontologies (Task 1.4)

Axioms and definitions	Single part whole	SUMO-OWL	Dolce-Lite	Oswebsite	OBO
Includes the relation *isPartOf*	X	X	X	X	X
A.1. Reflexivity					
A.2. Antisymmetry					
A.3. Transitivity	X		X	X	X
D.1. Proper part		X	X		X
D.2. Direct part	X			X	
D.3. Overlap		X	X		
D.4. Underlap					
D.5. Disjoint					
A.4. Weak supplementation					

The shadow features are required by the host ontology of section 6 use case (Task 1.2)

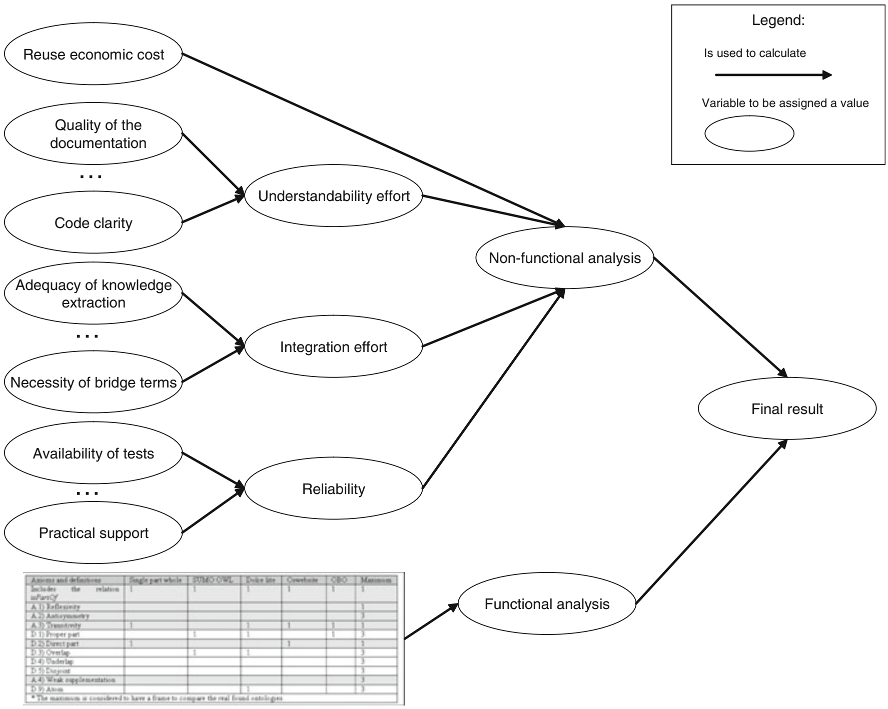

Fig. 7.2 How to take the decision of choosing an ontology (Task 1.5)

Features identified in Task 1.2 (reflexivity, transitivity, etc.) are called functional features, while the tag non-functional is used for the rest of features (reuse economic cost, code clarity, etc.). A weight of 0.75 has been assigned to non-functional features and 0.25 to the functional ones. These weights have been assigned because, according to the author's experience, adding new functional features to an ontology that scores well with respect to the non-functional ones is, in most cases, easier than overcoming the lack of compliance in non-functional properties. The exact value of each weight can be obtained using different procedures. One of them is by means of the utility theory (Jiménez et al. 2003). Another one is by means of former experience; this is the option we have used. That is, we have adjusted the weights so that the quantitative result applied to different cases of generic ontology reuse is equal to the one recommended by experienced people in this task. The first option follows a prescriptive approach, whereas the second, a descriptive one.

With the objective of having a reference to compare the scores, the score of an ideal ontology has been considered as a normalization denominator. Let us note that if this ideal reference is not provided, it is not easy to know the significance of the difference between the ontology scores. Thus, for example, without this ideal reference, if the difference between ontologies o_1 and o_2 is 0.4, the engineer cannot necessarily determine how large such difference is.

Given an ontology *ont*, the following formula to calculate the score of the functional features analysis is used:

$$ScoreFunctionalFeatures_{\text{ont}} = \frac{\sum_i \text{value}_{\text{ont}}(\text{functionalFeature}_i)}{\sum_i \text{value}_{\text{idealOnt}}(\text{functionalFeature}_i)} \times 100\% \quad (7.1)$$

where $\texttt{value}_{\text{ont}}(\texttt{functionalFeature}_i)$ is the value of functional feature *i* for the ontology *ont*, and $\texttt{value}_{\text{idealOnt}}(\texttt{functionalFeature}_i)$ is the value of functional feature *i* for an ideal ontology, that is, the number of features (axioms and definitions) obtained in Task 1.4.

Concerning non-functional features analysis, it is carried out on the basis of the following four dimensions:

- *Reuse economic cost*. It refers to the estimate of the economic cost needed for accessing and using the candidate ontology. If the candidate ontology has any type of license, then the cost of acquisition and/or exploitation should be taken into account (Gómez-Pérez and Lozano-Tello 2005).
- *Understandability effort*. It refers to the estimate of the effort needed for understanding the candidate ontology. In this case, the following criteria should be analyzed:
  - *Quality of the documentation*. It refers to whether there is any communicable material used to describe or explain different aspects of the candidate ontology (e.g., modeling decisions). The documentation should explain the statements contained in the ontology so that a nonexpert could understand them (Pinto and Martins 2001).
  - *Availability of external knowledge sources*. It refers to whether the candidate ontology has references to documentation sources and/or if experts are easily available.
  - *Code clarity*. It refers to whether the code is easy to understand and modify, that is, if the knowledge entities follow unified patterns and are intuitive

(Pinto and Martins 2001). It is advantageous to use the same pattern to make sibling definitions, thus improving ontology understanding and making it easier to include new definitions (Gómez-Pérez and Rojas 1999). For example, if it has been decided to distinguish between Frenadol product and Frenadol substance, the same distinction should be made for the rest of drugs (Efferalgan, Dolostop, etc.). Clarity also refers to whether the code is well documented, that is, if it includes clear and coherent definitions and comments for the knowledge entities represented in the candidate ontology. The difference between this criterion and the *quality of the documentation* is that *clarity* refers to the comments and the definitions inside the code; meanwhile, the *quality of the documentation* refers to external documentation (papers, manuals, etc.).

- *Integration effort*. It refers to the estimate of the effort needed for integrating the candidate ontology into the ontology being developed. In this case, the following criteria should be analyzed:

  – *Adequacy of knowledge extraction*. It refers to whether it is easy to identify parts of the candidate ontology to be reused and to extract them. For example, in large and not modularized ontologies (e.g., SUO), the difficulty to extract the part of the knowledge we are interested in is especially high.
  – *Adequacy of naming conventions*. It refers to whether both ontologies (the candidate and the one being developed) follow the same rules for naming the different ontology components (e.g., concept names should start with capital letters, relation names should start with non-capital letters).
  – *Adequacy of the implementation language*. It refers to whether both languages (the candidate ontology's and the ontology's being developed) are the same, or at least are able to represent similar knowledge with the same granularity.
  – *Knowledge clash*. It refers to whether there are contradictory bits of knowledge between the candidate ontology to be reused and the ontology being developed (e.g., discrete time versus continuous time assumption).
  – *Adaptation to the reasoner*. It refers to whether the adaptation of definitions and axioms that satisfy the existing restrictions of the reasoner is needed (e.g., explicit definitions can be included in OWL ontologies; however, this kind of definitions cannot be included in ontologies written in Prolog).
  – *Necessity of bridge terms*. It refers to whether it is necessary to create new linking axioms and/or relations to integrate the candidate ontology to be reused into the ontology being developed.

- *Reliability*. It refers to an analysis of whether ontology developers can trust the candidate ontology to be reused. In this case, the following criteria should be considered:

  – *Design criteria*. It refers to whether the ontology has been built according to the design criteria assumed by the development team of the domain ontology.

For example, one design criterion is the use of standards of the domain (on pharmacy, medicine, etc.) if they exist.

– *Availability of tests*. It refers to whether tests are available for the candidate ontology to be reused. Although it is not still usual in ontological engineering, the development team could publish the tests used during the ontology construction.
– *Former evaluation*. It refers to whether the ontology has been properly evaluated, not only by means of automatic unit tests but also by domain and ontology modeling experts.
– *Theoretical support*. It refers to whether the candidate ontology is supported by a sound theory, explicitly described in a document.
– *Development team reputation*. It refers to whether the development team of the candidate ontology is known to be experienced and competent.
– *Purpose reliability*. It refers to whether the candidate ontology has been developed as a simple example or for a stronger purpose.
– *Popularity*. It refers to whether there are well-known projects or ontologies reusing the candidate ontology (Lozano-Tello 2002).

Table 7.7 shows the criteria (organized by dimensions) and the ways to measure them. In the table, for each criterion, there is (a) a range of values (an interval of linguistic values or a natural number), (b) an explanation of how to measure the criterion, and (c) a numerical weight. The numerical weights are proposed here by default, according to the importance the authors give to the different criteria; for example, the criterion *design criteria* is extremely important for us, and therefore we assign a numerical weight of 10; however, the criterion *purpose reliability* is not so important for us, therefore, we give it a weight of 3. It is worth mentioning that such numerical weights depend on the importance the ontology developer gives to the different criteria, and that such weights can be modified. The symbols (+) and (−) in the weights are specified to indicate whether the criterion counts in a positive or a negative way, respectively. These symbols cannot be modified by the ontology developer.

Thus, ontology developers should fill a table and analyze the candidate ontologies with respect to the abovementioned criteria, taking into account the different ways to measure each criterion and the possible values that can be assigned.

Having filled Table 7.7 with different values for each criterion and for each candidate ontology, ontology developers should obtain a score for each candidate ontology and then decide which one is the most appropriate. To obtain such a score, the following method is proposed:

- To transform linguistic values, the following transformation rules are proposed:
  – Value = Unknown → $\text{Value}_T = 0$.
  – Value = Low → $\text{Value}_T = 1$ if the weigh is (+), 3 otherwise.
  – Value = Medium → $\text{Value}_T = 2$.
  – Value = High → $\text{Value}_T = 3$ if the weigh is (+), 1 otherwise.

Table 7.7 Decision criteria to select an ontology (Task 1.5)

Criteria	Range of values	How to measure it	Weight
Reuse cost			
Reuse economic cost	{Unknown, low, medium, high}	Asking the owner for an estimate	(−) 10
Understandability effort			
Quality of the documentation	{Unknown, low, medium, high}	Analyzing if the ontology has documentation and if such documentation really explains the ontology itself, as well as modeling criteria used during the ontology development	(+) 8
Availability of external knowledge	{Unknown, low, medium, high}	Analyzing if in the ontology documentation there is any reference to external sources that could be used to better understand the ontology	(+) 7
Code clarity	{Unknown, low, medium, high}	Inspecting the ontology code analyzing the complexity of the definitions (and axioms) implemented in the ontology	(+) 8
Integration effort			
Adequacy of knowledge extraction	{Unknown, low, medium, high}	Analyzing if the ontology is modularized or if it can be modularized in an easier way	(+) 9
Adequacy of naming conventions	{Unknown, Low, Medium, High}	Comparing the naming conventions of both ontologies	(+) 5
Adequacy of the implementation language	{Unknown, low, medium, high}	Comparing the ontology language of both ontologies. If both languages are different, analyzing the loss of knowledge in the translation	(+) 7
Knowledge clash	{Unknown, low, medium, high}	Comparing modeling decisions of both ontologies	(−) 7
Adaptation to the reasoner	{Unknown, low, medium, high}	Comparing the reasoners related to the ontology language of both ontologies	(+) 7
Necessity of bridge terms	{Unknown, low, medium, high}	Inspecting the ontology code and the result of Task 1.1 (see Table 7.3)	(−) 6
Reliability			
Design criteria	{Unknown, low, medium, high}	Analyzing if the ontology is built according to the design criteria assumed by the development team of the domain ontology	(+) 10
Availability of tests	{Unknown, low, medium, high}	Analyzing if the ontology documentation refers to existing unit tests	(+) 8
Former evaluation	{Unknown, low, medium, high}	Analyzing if the ontology documentation refers to different types of evaluation (automatic unit tests, human evaluation, etc.)	(+) 8

(continued)

Table 7.7 (continued)

Criteria	Range of values	How to measure it	Weight	
Theoretical support	{Unknown, low, medium, high}	Analyzing if the ontology documentation refers to the theory on which the ontology is based	(+)	10
Development team reputation	{Unknown, low, medium, high}	Searching for information about the ontology development team (other ontologies developed, papers published, etc.)	(+)	8
Purpose reliability	{Unknown, low, medium, high}	Analyzing if the ontology documentation refers to the purpose for which the ontology was developed	(+)	3
Popularity	{Unknown, low, medium, high}	Analyzing if the ontology documentation refers to other ontologies and/or projects reusing the ontology	(+)	7

– where:
– Value_T is the transformed value
– Value is the linguistic value provided by the ontology developer

Given that we want to penalize ontologies about which we have less knowledge, we have assigned a value of 0 to *unknown*.

- The score that synthesizes the non-functional features contribution is the following weighted mean:

$$ScoreNon\text{-}FunctionalFeatures_{\text{ont}} = \sum_i \text{Value}_{\text{Tont},i} \times \frac{\text{Weight}_j}{\sum_i \text{Weight}_i} \times 100\% \quad (7.2)$$

where:

– $ScoreNon\text{-}FunctionalFeatures_{\text{ont}}$ is the score for the candidate ontology *ont* for the set of criteria
– j is a particular criterion of those included in Table 7.7
– $\text{Value}_{\text{Tont},j}$ is the transformed value for the criterion j in the ontology *ont*
– Weight_k is the numerical weight associated to the criterion k
– Finally, applying the aforementioned weights of 0.25 and 0.75 for functional and non-functional features respectively, the following formula is applied:

$$Score = 0.25 \times ScoreFunctionalFeatures + 0.75 \times ScoreNon\text{-}FunctionalFeatures. \quad (7.3)$$

After applying the previous formula to all the candidate ontologies, ontology developers should select the candidate ontology with the best normalized scored.

Table 7.8 Determining the most appropriate mereology implementation (Task 1.5)

Criteria	Weight		Values				
			Single part whole	SUMO-OWL	Dolce-Lite	Oswebsite	OBO
Reuse cost							
Reuse economic cost	(−)	10	Low	Low	Low	Low	Low
Understandability effort							
Quality of the documentation	(+)	8	High	High	High	Unknown	Unknown
Availability of external knowledge	(+)	7	High	High	High	Unknown	Unknown
Code clarity	(+)	8	High	High	High	High	High
Integration effort							
Adequacy of knowledge extraction	(+)	9	High	High	Low	Low	Low
Adequacy of naming conventions	(+)	5	Low	High	Low	High	Low
Adequacy of the implementation language	(+)	7	High	High	High	High	High
Knowledge clash	(−)	7	Low	Low	Low	Low	Low
Adaptation to the reasoner	(+)	7	High	High	High	High	High
Necessity of bridge terms	(−)	6	Low	Low	Low	Low	Low
Reliability							
Design decisions	(+)	10	High	High	High	High	High
Availability of tests	(+)	8	Unknown	Unknown	Unknown	Unknown	Unknown
Former evaluation	(+)	8	Unknown	Unknown	Unknown	Unknown	Unknown
Theoretical support	(+)	10	High	High	High	Unknown	Unknown
Development team reputation	(+)	8	High	High	High	High	High
Purpose reliability	(+)	3	Low	Unknown	Unknown	High	High
Popularity	(+)	7	Unknown	Unknown	Unknown	Unknown	Unknown

As an example, in the context of the PPO case, we have filled in the values associated with the OWL versions of SUO-OWL and Dolce-Lite, which are shown in Table 7.8. The scores of the functional features have been obtained from Table 7.6.

The results in Task 1.5 have been very close. Given that we have found top-level concepts of SUMO-OWL like *biologically active substance* and *molecule* (and their ancestors) useful for PPO, the criterion *adequacy of knowledge extraction* has been assigned *high* for this ontology and, consequently, has obtained the best score (Table 7.9).

For this example, we have used a spreadsheet. For the future, we plan to support the automation of this task in NeOn Toolkit.

Table 7.9 Synthesis of the results of determining the most appropriate mereology implementation (Task 1.5)

	Single part whole	SUMO-OWL	Dolce-Lite	Oswebsite	OBO
Score for non-functional features. See formula (7.2). Henceforth, this result will be referred as (**3**)	85.33%	87.2%	79.73%	64.8%	62.13%
Score for functional features resulting from task. See Table 7.6 and formula (7.1). Henceforth, this result will be referred as (**4**)	33.33%	33.33%	50%	33.33%	33.33%
Final score = **0.75** × (**3**) + **0.25** × (**4**)	72.33%	**73.73%**	72.3%	56.93%	54.93%

7.2.2 Activity 2: Customizing the Selected Generic Ontology

The goal of this activity is to customize the ontology selected in Activity 1 according to the needs of the domain ontology being developed. This activity consists of the following tasks:

Task 2.1 Pruning the ontology to be reused according to the needed features. The goal of this task is to prune the selected ontology taking into account the features needed in the domain ontology that is being developed. Thus, for example, if the definition of overlap is defined in the generic ontology, but it is not necessary in the resulting ontology, it should be removed.

Task 2.2 Enriching the ontology to be reused. The goal of this task is to extend the ontology selected with the new conceptual structures needed in the domain ontology being developed. In the PPO example, we have added transitivity to the *part* and *properPart* object properties, reflexivity and antisymmetry to *part*, and asymmetry and irreflexivity to *properPart*.

When pruning and enriching the ontology, it is necessary to take into account that the axioms and definitions to be reused may be applicable to a category that does not completely include all the individuals of interest in our domain ontology. If this happens, an adaptation of the axioms and definitions should be performed.

Task 2.3 Translating the ontology to be reused into the implementation language of the domain ontology being developed. The goal of this task is to translate the selected ontology into the implementation language of the domain ontology being developed if those two ontologies are in different languages.

An ontology can be translated in an automatic or manual way. It is important to point out that a complete translation into different languages is not always possible. For example, let us suppose the following implementation in Prolog of `overlaps` and `disjoint`:

```
overlaps(X, Y) :- isPartOf(Z, X), isPartOf(Z, Y).
disjoint(X, Y) :- \+overlaps(X, Y).
```

The rule corresponding to disjoint cannot be implemented in OWL. In fact, let us note that given that Prolog works under the closed world assumption, if common parts of `substance1` and `substance2` have not been represented, the answer to the query:

```
?:- disjoint(substance1, substance2).
```

will be `true`. However, it is not possible to attain this effect directly with OWL (open world assumption).

Task 2.4 Adapting the ontology to be reused to the design criteria followed in the ontology to be developed. The following modifications have to be done in most cases: (a) changing names (concepts, properties) to adapt them to the naming conventions used in the ontology network being developed and (b) adding range to properties. For example, we have adapted the names to the convention used in PPO. Thus, *part* has been changed to *isPartOf*.

Task 2.5 Evaluating the obtained ontology. The goal of this task is to evaluate from a content perspective if there are no errors in the ontology. This task is described in detail in Chap. 9.

7.2.3 Activity 3: Integrating the Generic Ontology to be Reused in the Ontology Being Developed

The goal of this activity is to integrate the ontology obtained in Activity 2 in the ontology being developed. The development team should decide whether:

- To import the customized ontology. The advantage is that the resulting developed ontology will be structured in different modules[15] (see Chap. 10).
- To copy the customized ontology. This can be a good solution if the customized ontology belongs to the same domain as the one of the ontology to be developed. For example, if the customized ontology adds more drug types to a drug ontology.

In any case, links between terms of the reused ontology and the ontology to be developed should be established. In the case of PPO, we have taken advantage of the possibilities that SUMO-OWL offers us to easily represent different perspectives of the notion of drug, for example, drug as a substance that acts in our organism and drug as a product that can be sold. Moreover, given that transitivity, antisymmetry, etc. involve individuals, we have added an individual for each type of substance and product. Therefore, the application that uses the PPO maintains the individuals corresponding to particular entities (e.g., Frenadol C243, corresponding to Frenadol box with manufacturing lot C243) and the individuals that represent products and substances in a general way. Thus, for example, the

[15] The term *module* has here the pragmatic sense equivalent to the d'Aquin's reference cited in the Introduction.

Fig. 7.3 Partial view of the concept and the object property hierarchies (Snapshot taken from NeOn Toolkit)

system can infer that *caffeine* is part of *Frenadol* because there is an individual of *caffeine* (also with tag "*caffeine*") that is part of an object *Frenadol*, that is, an individual of *Frenadol*. We have also added the axioms identified in Table 7.3 (see Sect. 7.2.1) (e.g., *isMainActiveIngredient* is subrelation of *isPartOf*).

To answer CQ_4, we have added this rule to the ontology:

```
interactsWith(?x, ?y), isPartOf(?x, ?z) ->
      interactsWith(?z, ?y)
```

Table 7.10 Formal host competency questions that require *part of* modeling (for the sake of simplicity, prefixes, and value data types are omitted in the answers)

<table>
<tr><th>Informal CQ</th><th>Formal CQ</th><th>Example of answer</th></tr>
<tr><td>What drugs do have paracetamol?</td><td># CQ1

SELECT ?X
WHERE
{
?X rdf:type ub:DrugSubstance .
ub:Paracetamol ub:isProperPartOf ?X .
}</td><td>---------------------
|X|
=====================
|FrenadolSubstance
|BisolgripSubstance
|CortafriolSubstance
|DolgesicSubstance
|TermalginSubstance
|AlgidolSubstance
|EfferalganSubstance
|DolostopSubstance
|GelocatilSubstance
|ApiretalSubstance
|PharmagripSubstance
---------------------</td></tr>
<tr><td>Which is the composition of Frenadol®?</td><td># CQ2

SELECT ?X
WHERE
{
?X ub:isProperPartOf
ub:FrenadolSubstance .
}</td><td>---------------------
|X|
=====================
|Dextrometorphan
|CitrateOfChlorpheniramine
|Caffeine
|Paracetamol
---------------------</td></tr>
</table>

(continued)

Table 7.10 (continued)

Informal CQ	Formal CQ	Example of answer
Which is the main active ingredient of Frenadol®?	`# CQ3` `SELECT ?X` `WHERE` `{` ` ?X ub:isMainActiveIngredientOf` `            ub:FrenadolSubstance` `.` `}`	`---------------` `| X             |` `===============` `| Paracetamol |` `---------------`
Which substances do Frenadol® interacts with?	*`# CQ4`* *`SELECT ?X`* *`WHERE`* *`{`* *` ub:FrenadolSubstance`* *`ub:interactsWith`* *`   ?X .`* *`}`*	`----------------` `| X              |` `================` `| Rifanpicin    |` `| Propranolol   |` `| Isionazid     |` `| EthylAlcohol |` `----------------`

That is, if a substance ?*x* interacts with another substance ?*y*, then the latter interacts with every part of ?*x*. Thus, for example, given that paracetamol interacts with the ethyl alcohol, Frenadol® also interacts with ethyl alcohol.

A partial view of the resulting ontology is shown in Fig. 7.3.

The resulting ontology should be evaluated. In the PPO case, besides other tests, we have checked that the CQs are answered (see Table 7.10).

7.3 Conclusions and Future Work

The reuse of (well-developed) ontologies allows spreading good practices and increasing the overall quality of ontological models. In this chapter, we have presented how to carry out this process. The guidelines shown here provide the methodological assistance to Scenario 3 in the NeOn Methodology (Chap. 2).

Given that the reuse of an ontology usually implies pruning it, ontology reuse usually implies statement reuse (see (Suárez-Figueroa 2010) to know more about how to reuse domain ontologies as well as ontology statements). Consequently, we have not distinguished between these two classes of reuse.

It is also worth mentioning that interesting knowledge represented in ontologies may be found by chance. For instance, part of the knowledge on substances and products reused from SUMO-OWL has been found when we were searching for mereology knowledge.

The NeOn Toolkit includes the Watson plugin to support ontology search. An objective for future development will be to develop the necessary plugin to assist with the other tasks associated with ontology reuse, especially for the selection of the most appropriate ontology (Task 1.5).

In addition, it would be interesting to perform a comparison of the costs of (a) reusing generic ontologies versus (b) developing what is required from scratch.

Annex: Mereology

A *mereology* is a formal theory of parts and associated concepts (Borst 1997; Schneider 2003). We have said "*a* mereology" instead of "*the* mereology" because different assumptions can be taken into account in the formalization of parthood. Therefore, different mereologies can be proposed.

In the following paragraphs, we will show one of the mereologies presented by Varzi (2007).

Theory *M*. Most of the authors agree on the following core of axioms (named with A) and definitions (named with D) (Varzi 2007). Along these paragraphs, we use examples of territories to clarify the meaning of axioms and definitions. The mentions to administrative units really refer to their physical territories.

- *A.1. Reflexivity*. Every object of the universe of discourse is a part of itself. For instance, the EU is part of the EU.
- *A.2. Antisymmetry*. If an object x is a part of y, and y is a part of x, then x and y are the same object. For instance, if the territory T_1 is part of the territory T_2, then the only way so that T_2 is part of T_1 is being T_1 and T_2 the same territory.
- *A.3. Transitivity*. If x is a part of y, and y is a part of z, then x is a part of z. For instance, the Community of Madrid is part of Spain, and Spain is part of the EU; therefore, the Community of Madrid is a part of the EU.

A number of additional mereological predicates can be then introduced by definition:

- *D.1. Proper part*. A proper part is a part that is other that the individual itself. For example, Spain is proper part of the EU, since Spain is part of the EU and they are different entities.
- *D.2. Direct part*. *X* is direct part of y if and only if x is proper part of y and there is no part between x and y[16]. For example, Spain is direct part of the EU, but Madrid is not, since Spain is a part between Madrid and the EU.

[16] http://hcs.science.uva.nl/projects/NewKACTUS/library/lib/mereology.html

- *D.3. Overlap*. The relation *overlaps* is defined as a sharing part. That is, *x* and *y* overlap if and only if there is a *z* such that *z* is part of *x* and part of *y*. For instance, Nordic countries and the EU overlap, since there are Nordic countries which are parts of the EU.
- *D.4. Underlap*. The relation *underlaps* is defined as a sharing whole. That is, *x* and *y* underlap if and only if there is a *z* such that *x* and *y* are parts of *z*. For example, the Netherlands, Sweden, and Spain underlap the same common whole: the EU.
- *D.5. Disjoint*. The *disjoint* relation is the logical negation of *overlaps*. For example, Belgium and the Netherlands are disjoint territories.

Theory *M* may be viewed as embodying the common core of any mereological theory. A.1–A.3 should be extended to build a mereology.

Minimal mereology (*MM*). A way to extend *M* is assuming the following principle (Varzi 2007):

- *A.4. Weak supplementation principle*. Every object *x* with a proper part *y* has another part *z* that is disjoint from *y*. The domain of territories, for example, fulfills this principle. For example, given that Spain is proper part of the EU, then the EU has other parts that are disjoint from Spain: the Netherlands, Luxemburg, Sweden, etc.

Most of the authors strengthen that A.4 should be incorporated to *M* as a further fundamental principle on the meaning of *part of*. Other authors provide scenarios that could be counterexamples of this principle. However, it is far from being demonstrated that such supposed counterexamples have implications in computer applications.

The rest of mereologies starting from MM are explained with examples in (Fernández López et al. 2008; Suárez-Figueroa 2010).

References

Borst WN (1997) Construction of engineering ontologies. Centre for Telematica and Information Technology, University of Tweenty, Enschede

d'Aquin M, Lewen H (2009) Cupboard – a place to expose your ontologies to applications and the community. Demo at European Semantic Web Conference, ESWC 2009, Heraklion, Greece

d'Aquin M, Motta E (2011) Watson, more than a semantic web search engine, Semant Web J 2, IOS Press

d'Aquin M, Sabou M, Dzbor M, Baldassarre C, Gridinoc L, Angeletou S, Motta E (2007a) Watson: a gateway for the semantic web. Poster session of the European Semantic Web Conference, ESWC 2007, Busan

d'Aquin M, Schlicht A, Stuckenschmidt H, Sabou M (2007b) Ontology modularization for knowledge selection: experiments and evaluations. In: 18th international conference on Database and Expert Systems Applications, DEXA 2007, Regensburg, Germany

d'Aquin M, Sabou M, Motta E (2008) Reusing knowledge from the semantic web with the Watson Plugin. Demo at International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany

Fernández López M, Gómez-Pérez A, Suárez-Figueroa MC (2008) Selecting and customizing a mereology ontology for its reuse in a pharmaceutical product ontology. In: Grüninger M, Eschenbach C (eds) Formal Ontology in Information Systems. Fifth international conference (FOIS-2008), Saarbrücken, Germany. IOS Press, Amsterdam, pp 181–194

Golbeck J, Fragoso G, Hartel F, Hendler J, Parsia B, Oberthaler J (2003) The national cancer institute's thesaurus and ontology. J Web Semant1(1):75–80

Gómez-Pérez A, Lozano-Tello A (2005) Applying ONTOMETRIC method to measure the suitability of ontologies. In: Green P, Rosemann M (eds) Business systems analysis with ontologies. Idea Group Publishing, Hershey, pp 249–269

Gómez-Pérez A, Rojas MD (1999) Ontological reengineering and reuse. In: Fensel D, Studer R (eds) 11th European workshop on Knowledge Acquisition, Modeling and Management (EKAW 1999), Dagstuhl Castle, Germany, Lecture Notes in Artificial Intelligence LNAI 1621 Springer, Berlin, Germany, pp 139–156

Jiménez A, Ríos-Insua S, Mateos A (2003) A decision support system for multiattribute utility evaluation based on imprecise assignments. Decis Support Syst 36:65–79

Lozano-Tello A (2002) Métrica de idoneidad de ontologías. PhD Thesis, Universidad de Extremadura, Cáceres, Spain 2002

Peroni S, Motta E, d'Aquin M (2008) Identifying key concepts in an ontology through the integration of cognitive principles with statistical and topological measures. In: Third Asian semantic web conference, Bangkok, Thailand

Pinto HS, Martins JP (2001) A methodology for ontology integration. In: Gil Y, Musen M, Shavlik J (eds) First international conference on Knowledge Capture (KCAP 2001), Victoria, Canada. ACM Press, New York, pp 131–138

Schneider L (2003) How to build a foundational ontology: the object-centered high-level reference ontology OCHRE. In: Günter A, Kruse R, Neumann B (eds) Proceedings of the 26th annual German conference on Artificial Intelligence (KI-2003), Hamburg, Germany. Lecture Notes in Artificial Intelligence (LNAI-2821), Berlin, Germany, pp 120–134. (http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=12B7EC62C9601245457C735C07AA07A0?doi=10.1.1.1.3440&rep=rep1&type=pdf)

Suárez-Figueroa, MC (coordinator) (2008) D5.4.1. NeOn Methodology for building contextualized ontology networks. NeOn project

Suárez-Figueroa, MC (2010) NeOn Methodology for building ontology networks: specification, scheduling and reuse. PhD Thesis, Facultad de Informática, Universidad Politécnica de Madrid, Spain

van Heijst G, Schreiber ATh, Wielinga BJ (1997) Using explicit ontologies in KBS development. Int J Hum-Comput Stud 45:183–292

Varzi A (2007) In: Aiello M, Pratt-Hartmann I, van Benthem J (eds) Spatial reasoning and ontology: parts, wholes, and locations. Springer, Heidelberg, pp 945–1038