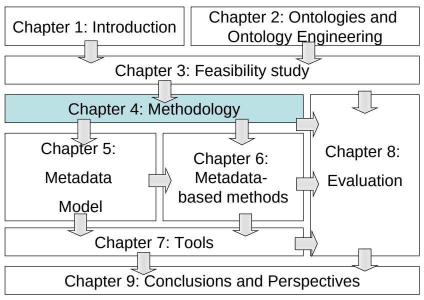
4 Ontology Reuse Methodology



Chapter 1: Introduction
Chapter 2: Ontologies and Ontology Engineering
Chapter 3: Feasibility study
Chapter 4: Methodology
Chapter 5: Metadata Model
Chapter 6: Metadata-based methods
Chapter 8: Evaluation
Chapter 7: Tools
Chapter 9: Conclusions and Perspectives

This chapter describes ontology reuse from a process-oriented point of view. The process is divided into three phases (cf. Section 4.1). For each process phase (Sections 4.2 to 4.4) we introduce the participants, the activities accomplished by them and the expected results, pointing out application-specific optimizations. Further on, in order to enhance the efficiency and effectivity of the proposed methodology in real-world scenarios, we analyze which methods, techniques and tools could aid the methodology applicants during particular phases of the process. The results of this investigation are compiled in Section 4.5 into a requirements specification, which forms the basis for the design and development of the methods and tools introduced the proximate chapters. Section 4.6 completes this chapter with a short summary of its contents.
References*: This chapter is based on the publications [168, 169, 170, 172].*

4.1 Overview

As the Semantic Web grows an increasing number of private and public sector communities are developing ontologies which represent their domain(s) of interest. As ontologies are also intended to act as *shared* and *reusable* domain conceptualizations [84] it is expected that they will be put into widespread use, possibly under some license, on the Web. This will bring the benefit of ontology engineers and users being able to deploy existing models or align them to local ones, thus reducing implementation costs, improving the quality of ontological sources, which are, by re-use, subject of continuous revisions and refinements, and increasing the interoperability between ontology-based applications.

4.1.1 General Process

Ontology reuse is an integral part of ontology engineering. Resorting to the terminology in [80] the decision whether to build (fragments of) an application ontology by reusing (an adapted version of) existing ontological sources is one of the designated *support activities* performed in addition to the core *ontology development*. Consequently, ontology reuse can not be exercised in a standalone manner, but within a framing ontology engineering process.

The latter pre-defines the *characteristics* of the prospected ontology. Some of these form the basis for the requirements that should be satisfied by the evaluated ontologies, be that application-specific or general purpose requirements.

Once the analysis of the domain has been completed (cf. Section 2.2) and this ontology building alternative has been positively evaluated by the engineering team, the ontology reuse process can be initiated (cf. Figure 4.1). As input the participants are provided with an *ontology requirements specification document*, which entails a compilation of the most important features of the planned ontology and of the application setting [214]. In particular the subdomains of the final ontology which are to be built by reuse are defined in advance as part of this activity. Further on, the document gives full particulars on the *application setting*, the *use cases*, the *purpose* and the expected *size* of the outcomes. The reuse process is completed with the integration of the reuse candidates into the original application setting. Technically this is equivalent to a new application ontology which incorporates the parts constructed by reuse and by other building activities (such as manual building, ontology learning, information extraction etc.) The resulting ontology is then evaluated using the methods and against the criteria which have been pre-defined in the ontology engineering methodology. If the ontology does not fulfill the expected requirements to a satisfactory extent, the engineering team may initiate a re-iteration of the overall ontology building process at various stages. This includes under circumstances ontology reuse.[1]

Just as ontological engineering, the ontology reuse process involves four categories of participants:

Ontology/Knowledge engineers : develop the ontology independently of its future implementations. They are experts in Knowledge Representation and Knowledge Modelling and are not required to be familiar with the domain of the ontology.

Domain experts : impart knowledge about the domain that is modelled with the help of the ontology. They usually have little to no experience in Ontological Engineering.

Programmers : implement the ontology and embed it into the target application system. Their focus is on technologies and tools for the creation and management of ontologically represented data.

Users : utilize the outcomes of the ontology engineering process within the specific application setting. They participate in the ontology engineering process in order to ensure an appropriate balance between the application-independence of the model and its usability in the target setting.

Accounting for the empirical findings of the case studies (cf. Chapter 3) we decompose the reuse process in three sequentially ordered steps/phases (cf. Figure 4.2):

1. **Ontology discovery**: this step is dedicated to finding a list of potential reuse candidates given a set of minimal, easy-to-evaluate requirements.

[1]Note that Figure 4.1 does not commit to a particular ontology life cycle. The decision whether an ontology is built incrementally or as a result of a sequential workflow is out of the direct scope of ontology reuse, though it does have implications on the way this is eventually performed.

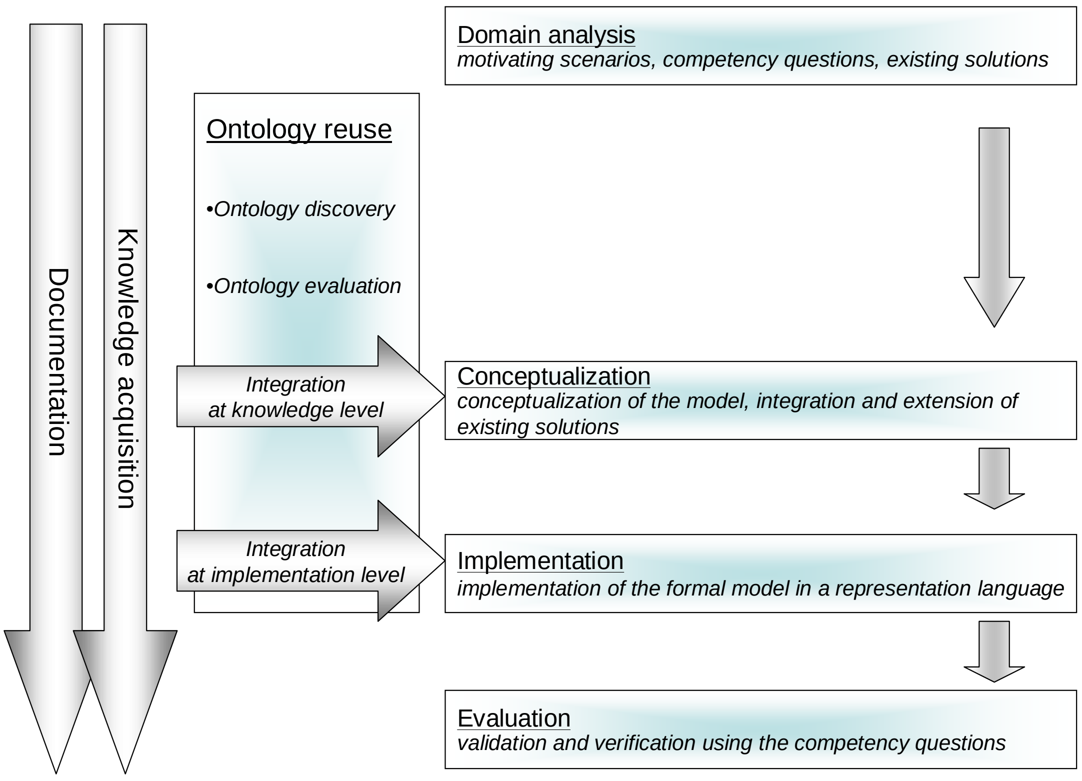

Documentation
Knowledge acquisition
Ontology reuse
•Ontology discovery
•Ontology evaluation
Integration at knowledge level
Integration at implementation level
Domain analysis
motivating scenarios, competency questions, existing solutions
Conceptualization
conceptualization of the model, integration and extension of existing solutions
Implementation
implementation of the formal model in a representation language
Evaluation
validation and verification using the competency questions

Figure 4.1: Ontology Reuse in Relation to Ontology Engineering

2. **Ontology evaluation**: this phase is concerned with assessing the usability of the ontological sources to be reused with respect to the target application setting.

3. **Ontology merging and integration**: in this step the appropriate ontologies are customized, merged and integrated into the final application. The result is in form of a single or a group of ontologies represented in a particular representation language. They can be utilized in the target application system for the completion of particular tasks.

The first step of the process is primarily *technological*. The engineering team resorts to conventional search techniques and browses repositories of ontological resources in order to build a list of potential reuse candidates. At Web scale the search is usually performed automatically. The participants specify a minimal set of desired features in terms of machine-understandable queries and review the query results.

In the second step the reuse candidates are subject to an in-depth evaluation performed by domain experts and ontology engineers in terms of a pre-defined strategy. By contrast to ontology discovery, the evaluation step is inconceivable without an adequate *methodological* background. While technological support is definitely helpful, the usability assessment task is targeted in principle at humans. The result is a set of *reusable* ontologies complemented

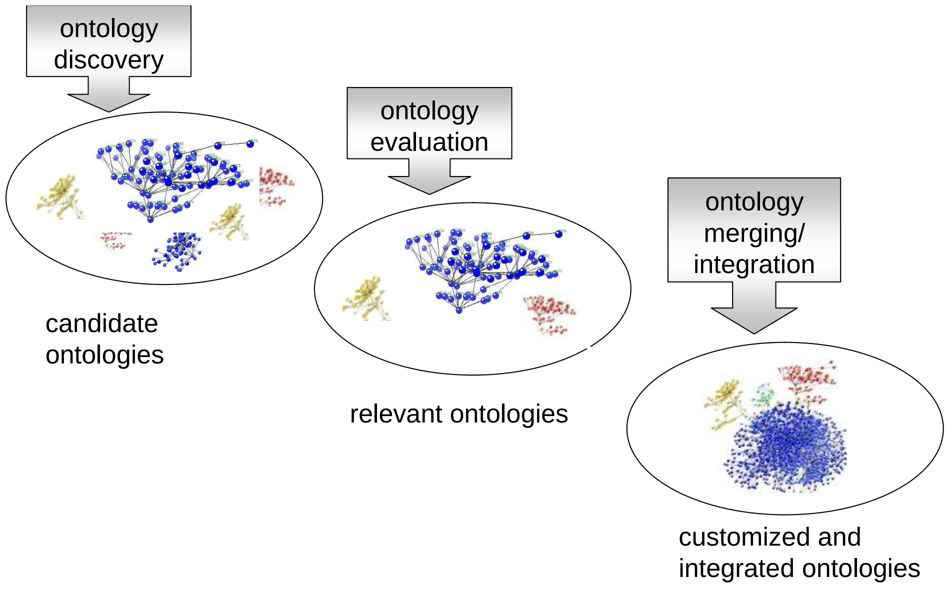


Figure 4.2: Ontology Reuse Process

by an inventory of actions to be taken for their customization.

The last phase of the reuse process comprises the execution of the customization operations followed by the integration of the reused ontologies into the application system. It requires both *methodological* and *technological* assistance. The choice upon a specific merging/integration strategy is not trivial and should be therefore guided with the help of a dedicated methodology. Further on, feasible tools and APIs are essential for the completion of the integration step; a manual processing is realistic, though implying significant efforts, only for small inputs containing several hundreds of ontological primitives at most (cf. for example [5, 81]).

Orthogonal to the ontology life cycle, the reuse process is typically performed sequentially; iterations are possible and recommended within the scope of the three main process phases, and are associated with major efforts otherwise (cf. Sections 3.3 and 3.4 in the previous chapter).

4.1.2 Ontology Reuse Context

Unlike existing approaches in the field, our methodology explicitly considers *the context* in which the reuse process is being performed, refining and optimizing the reuse strategy accordingly. The context roughly refers to additional information about the participants in the reuse process, the ontologies being examined and the application setting at which the final ontology is targeted. The result is a fine-grained and, in the same time, pragmatical description of the reuse process, which is expected to provide a more effective guidance to both ontology engineers and domain experts in application scenarios widely acknowledged to take benefit from using ontologies.

Accounting for the conclusions derived from the feasibility study (cf. Chapter 3) the general reuse process differentiates between five application scenarios for ontologies:

1. **Integration**: the ontology provides an integrating environment, an inter-lingua, for information repositories or software tools. In this scenario the ontology is intended to be applied (semi-)automatically to merge between heterogeneous data pools in the same or in adjacent domains. In the context of programmed agents or software tools, in which ontologies mediate between the ways these conceptualize a shared domain of interest, integration is envisioned to be performed automatically.

2. **Semantic search/retrieval**: the scenario characterizes how ontologies are used to refine common (keyword-based) search algorithms using domain knowledge in form of subsumption relations or logical constraints. Ontology-driven search is usually performed automatically by means of reasoning services handling particular aspects of an ontology representation language. Additionally to this "classical" perception we encounter ontology-based retrieval applications, in which ontologies, just as in the semantic indexing scenario, are used to control the accepted query vocabulary or to browse the returned result list on the basis of domain-specific patterns. In these cases the ontology is not associated with reasoning services and its interaction with the system users is not necessarily automatized.

3. **Semantic indexing/annotation**: in this scenario the goal of the ontology is to provide a controlled vocabulary, as well as a clearly defined classification and browsing structure for the information items in a repository. This task can be performed manually by domain experts or as part of an application in an automatic or semi-automatic way.

4. **Software engineering**: the usage of ontologies in the context of software engineering is strongly influenced by the emergence of so-called *"model-driven architectures"*, which envision to apply them for software verification and validation.[2]. A second application is software configuration: an ontology is used to separate the configuration parameters of a software application from its concrete implementation. In doing so, it offers means to represent configuration data and valid configuration settings in a machine-understandable manner. The task is executed automatically in that the underlying application system uses the ontologically-represented information, possibly in correlation with reasoning services, in order to realize the desired run-time configuration. The verification and validation tasks resort to the same characteristics of ontologies.

5. **Knowledge representation**: the ontology is used as a means to formalize the kinds of things that can be talked about in a system or a context.

The roles an ontology might play in these application scenarios can be summarized into four categories:

1. **Vocabulary**: the ontology is used as a controlled vocabulary describing the most important concepts of a domain of interest, their properties and their relations to each other. This role is sometimes termed as *inter-lingua* or *lingua-franca*.

[2]`http://www.omg.org/mda` last visited in September, 2005

2. **Formal model**: by contrast to the previous category the ontology is used as a model describing the way concepts in a domain are interconnected and the axioms constraining the meaning and the behavior of the concepts. The formality aspects refers to the usage of a representation language with machine-understandable semantics.

3. **Index**: this role characterizes how ontologies are applied as classification structure to index the information items of a repository, be that the Web, or the corporate memory of an organization. The focus is again on the usage of a controlled vocabulary: the ontology pre-defines a set of domain categories, but also on the hierarchical organization of these categories.

4. **Filter**: in this role the ontology is applied to refine the results of a specific algorithm, usually in an information retrieval context.

Table 4.1 shows the aforementioned ontology role in relation to the application scenarios.

	Vocabulary	Formal model	Index	Filter
Integration	✓	✓	-	-
Semantic search/retrieval	✓	✓	✓	✓
Semantic annotation/indexing	✓	✓	✓	-
Software engineering	✓	✓	-	-
Knowledge representation	✓	✓	-	-

Table 4.1: Ontology Roles in Relationship to Application Scenarios

The classification was compiled on the basis of research conducted in several recent large scale European projects in the area of Semantic Web [115, 123, 163, 215]. Each of the aforementioned application scenarios is implicitly correlated to different reusability constraints or imposes a different prioritization of the scenario-independent requirements [223, 225]. As indicated by the case studies these dependencies play a central role for the feasibility of a reuse-oriented ontology engineering approach as compared to manual ontology building or ontology learning.

Each application scenario is primarily aimed at a particular target user group, be that *humans*, *machines* or both. This distinction is in direct relation with the automatization level of the underlying task; if a scenario is aimed at humans, the corresponding task is accomplished *manually*; similarly, tasks targeted at machines are termed as *(fully) automatic* or *(fully) automatized*. The two user groups impose different requirements upon the ontology to be reused. One of the determining factors for the reusability of human-processable ontologies is for instance its *comprehensibility/readability*: it is important that ontological primitives such as concepts, properties or axioms are described by natural language, unambiguous labels and definitions. In the same time, issues like syntactic validity or consistency with respect to the ontology representation language semantics can be considered as secondary—if no automatic reasoning services are deployed with the help of the planned ontology. However, if ontologies are to be machine-understandable, it is fundamental that they are represented in a formally defined language and that they are syntactically valid and correct [225, 180]. By contrast natural language comments, definitions or labels are not mandatory. Further similar application-dependent issues are raised by the task the ontology is expected to be used for and by particular characteristics of the application scenario, be that of technological or organizational nature. Technological aspects are critical due to three categories of factors: firstly, semantic technologies are still in their infancy and have not achieved a feasible level of maturity yet; secondly, the scalability and performance problems which are expected to be handled by these technologies (e.g., reasoning over large knowledge bases) can be sometimes solved solely at the expense of disclaiming specific ontology features; third, there is no clear evidence for the solvability of some of these problems whatsoever. While the first issue might be temporary, the second and the third definitely raise questions about the feasibility of ontology-driven applications at large scale. Consequently, the impact of the technological circumstances of an application scenario should not be underestimated during the reuse process. These considerations are elaborated in more detail in each of the three process steps described below.

Besides the application scenario our methodology differentiates among three levels of reuse of ontologies:

1. **Reusing the vocabulary**: at this level the engineering team utilizes the set of natural language labels denominating ontological primitives (concepts, instances, properties). There is no guarantee that the semantics of the underlying model, partially constrained by the choice upon a formal knowledge representation language, is preserved in the target ontology. However the reused ontologies are a helpful information source for the description of the application domain.

2. **Reusing the vocabulary and the semantics**: at this level the engineering ontology is additionally interested in preserving at least parts of the intended meaning of the original domain modelling. In this category one can further distinguish between the following situations:

   a) **Reusing the classification**: the specialization/generalization hierarchy is re-utilized in the new context.

   b) **Reusing properties**: in addition to the taxonomy, the engineering team preserves the domain-specific properties connecting the ontological concepts.

   c) **Reusing axioms**: at this level the engineering team re-utilizes the complete original conceptualization of the domain of interest.

3. **Reusing instance data**: instances are valuable source of information, which can be utilized in new application settings with or without the corresponding terminological knowledge. If the reuse of the data is carried out separately from the underlying schema, the engineering team is required to map the imported data to the new conceptual structures.

REUSE PROCESS	
PROCESS STEPS	Ontology discovery Ontology evaluation Ontology customization, merging and integration
PROCESS PARTICIPANTS	Ontology engineers Domain experts Programmers Users
CONTEXTUAL INFORMATION	
ONTOLOGY TASK	Integration Semantic search/retrieval Semantic indexing/annotation Software engineering Knowledge representation
ONTOLOGY ROLE	Vocabulary Formal model Index Filter
REUSE LEVEL	Vocabulary Vocabulary and semantics Instance data

Table 4.2: Key Dimensions of the Context-sensitive Ontology Reuse Methodology

This additional process dimension is justified by the observation made during the feasibility study that the main challenges reported in the case studies were caused by the all-or-nothing strategy applied by the majority of methods and tools available. This inflexibility was in contradiction with the fact that in most of the cases the engineering team was reusing solely a part of the available knowledge. This applies for two situations:

- **Expressivity mismatches**: the expressivity of the target ontology might be different than the ones of the considered sources. Consequently the reuse methodology—including the associated support methods and tools—should profit from this complexity reduction and be optimized for exploiting the lowest common expressivity level of the source and target ontologies to a maximal extent.

- **Lack of support methods and tools**: the expressivity of the source ontologies is not adequately supported by the methods and tools applied during the reuse process. If the corresponding activities can not be carried out automatically, the engineering team might consider performing them at a lower expressivity level prior to manually post-processing the results.

In order for the proposed methodology to provided real added value to its applicants in a wide range of situations, it should explicitly consider this additional dimension as well.

We summarize the main coordinates of the ontology reuse methodology in Table 4.2 before turning to a detailed description of the process model. This introduces each process phase using the subsequent pattern.

DESCRIPTION an overview of this process step and its main activities.

PARTICIPANTS roles involved in this process step and their responsibilities.

RESULTS Results expected after the completion of this step in terms of ontologies and actions to be performed as a transition to the next steps.

SUPPORT METHODS AND TOOLS means to automatize particular activities performed in this step.

CONTEXT-SPECIFIC ISSUES guidelines for the optimization of the process in particular contexts.

4.2 Ontology Discovery

DESCRIPTION The objective of this process phase is the generation of a preliminary list of reuse candidates. To find ontologies one can currently choose between

- accessing dedicated repositories
- using general-purpose search engines, and
- trying to decide which organizations best represent their domain of interest and checking whether they announced the release of any relevant ontologies or ontology-like structures.

On the basis of the information provided in the ontology requirements specification document, the engineering team agrees on a number of keywords optimally describing the domain(s) of the expected ontologies and translates these to the format required by the specific search technology, be that general-purpose engines or dedicated repositories. Further feature-driven restrictions are recommended only if fully-fledged ontology repositories including rich metadata descriptions of ontologies are available. Instead, if conventional search engines are used to find the ontologies, it is recommended that the corresponding query should be extended with keywords paraphrasing the notion of ontology such as *ontology*, *classification*, *taxonomy*, *thesaurus*, *dictionary*. These denominate particular types of lightweight ontologies [80, 225], whose theoretic, human-understandable semantics can be explicitly formalized to a large extent. If availability issues (primarily costs and licences) are essential to the project, they provide a very efficient selection criterion for the generation of the reuse candidates list.

PARTICIPANTS Ontology engineers, domain experts, programmers, users. Domain experts and users decide upon the terms which are likely to be representative for the description of the domain of interest and evaluate the results of the search from a content-oriented perspective. Participants with a technical background complete this superficial evaluation by reviewing a core set of ontology engineering-related features. Users are involved in deciding upon availability issues.

RESULTS A list of potential reuse candidates and associated documentation. If an ontology is not freely available, while still being relevant for the subsequent reuse phases, its reusability could be determined with the help of its documentation to a specific extent.

SUPPORT METHODS AND TOOLS As underlined before ontology discovery is inconceivable without (minimal) computer support. This includes *ontology repositories* providing search and browse services for ontological resources they manage. General-purpose *search engines* can be applied to discover ontologies which are not included in any of the known repositories. Furthermore, an interface to the reuse candidates, be that via a common Web browser or a direct access to the data store, is required.

CONTEXT-SPECIFIC ISSUES Human-targeted application scenarios are likely to take benefit from the re-usage of lightweight ontologies or ontology-like structures. For instance, using popular classification structures for the semantic indexing/annotation of information items offers significant advantages both with respect to the individuals accomplishing the task and the application users.

4.3 Ontology Evaluation

DESCRIPTION The goal of this phase is to assess the usability of the pre-selected reuse candidates with respect to the requirements of the actual context. Due to its well-acknowledged complexity, correlated to its fundamental role in the success of a reuse approach (cf. Chapter 3), we break down the evaluation phase to the following dimensions:

Content evaluation: the evaluation of the information contained in the ontology with respect to its relevance to the domain of interest of the application.

Knowledge representation evaluation: the evaluation of the ontological representation of the content with respect to the quality of the modelling.

Technical evaluation: the assessment of the ontology usability with respect to the technical context of the application.

Application evaluation: the usability of the ontology as required by the application in which the final ontology will be embedded in and the tasks which should be accomplished by the ontology.

Availability evaluation: the evaluation of the availability constraints imposed by an ontology and its provenance institution.

We explicitly do not impose either a specific execution order of the enumerated steps, or a ranking of their importance. These aspects should be agreed prior to the assessment task within the engineering team. However, following the empirical findings mentioned in Chapter 3, we consider steps 1, 4 and 5 to be fundamental to obtaining meaningful usability assessment results independently of any contextual details. Regarding the execution order of the individual phases, executing the content evaluation at first proved to be beneficial in the scenarios in which we applied the methodology (cf. Chapter 8). However, in a scenario with high availability constraints (e.g., freely available ontologies permitted) the availability evaluation provides a useful selection procedure and should be performed prior to other activities in order to avoid investing resources in evaluating unsuitable ontologies. These two dimensions are also mentioned by related research approaches as highly relevant to the ontology evaluation task (e.g., [128]). A technical evaluation is particularly helpful as long as ontology-driven technologies do not achieve a feasible maturity level. The knowledge representation evaluation is in most of the cases associated to major operating expenses, while its benefits in *application-oriented context* are debatable [114]. The application evaluation is a novel evaluation dimension in this context. Application-driven requirements and their implications on the usability of available ontologies have been marginally exploited in the ontology engineering research so far. Their impact on the success of an ontology reuse endeavor was derived from the feasibility study we performed prior to designing this methodology.

The results of the individual evaluation operations are aggregated by the engineering team with the help of specific methods towards a final list of usable ontologies. The methods employed to assemble them might differ from setting to setting (see below).

PARTICIPANTS Ontology engineers, domain experts, programmers, users. The task distribution is described separately for each of the evaluation dimensions.

RESULTS Usable ontologies. For each ontology the engineering team specifies the reuse level and the activities required to customize them according to the application-specific requirements. The type of these activities depends on the evaluation dimension being performed.

SUPPORT METHODS AND TOOLS Currently the process is not supported by dedicated environments. However every sub-task might take benefit from specific (ontology management) methods and tools. Further on, as the evaluation can not be performed without human contribution, the availability of descriptive information about the analyzed ontologies and the usage of a *uniform format* to represent this information influences the perceived complexity and comprehensibility of the ontological resources—and thus the costs of the total reuse process.

Firstly the participants should decide upon the way they rank the ontologies and upon the threshold marking the positively evaluated ones. The ranking criteria might vary across the evaluation dimensions, while the threshold might be adapted to the number and the quality of the examined resources. Commonly used in this context is the setting of absolute or relative values (e.g., a fixed number as in "the first five hits" or a percent of the total hits as in "20% of the results"). The easiest way to decide upon the final reusable ontologies from the results of the five evaluation sessions is to pick up the ones included in each or most of the categories. If the evaluation dimensions themselves are not equally important or none of the analyzed ontologies was assigned a positive rating on all of the dimensions considered we suggested case study participants to rate the usability of each ontology using a uniform scale system and to weigh the partial results in order to ease the comparison and to quantify the final assessment results (cf. Chapter 8) . Chapter 6 introduces a method to operationalize the evaluation task, which is implemented prototypically as part of the PROMI framework (cf. Chapter 7).

CONTEXT-SPECIFIC ISSUES The analysis of the aforementioned application scenarios revealed further correlations between the structure of the usability assessment process and specific characteristics of the setting in which the ontologies are to be reused:

- **Integration**: content, availability, application and technical evaluation.
- **Semantic search/retrieval**: content, availability, application and technical evaluation. If these tasks are not performed automatically, the technical evaluation is not mandatory. If the semantic search is centered on the intensive usage of inferences then a knowledge representation evaluation is necessary to prevent subsequent reasoning faults.
- **Semantic indexing/annotation**: content, availability, application and technical evaluation. The technical evaluation is not mandatory for human-driven tasks.
- **Software engineering**: content, availability, application, knowledge representation and technical evaluation. The knowledge representation evaluation is relevant in cases in which the ontology is intended to represent the domain of interest of the prospected software.
- **Knowledge representation**: content, availability, and knowledge representation evaluation.

Some of the five evaluation dimensions are essential to obtain meaningful results: the content, the availability and the application evaluation. The way the former two tasks are performed is influenced by contextual aspects only to a limited extent. The application evaluation is operated in a context-independent manner, though the decisions to be made highly rely on the context of the reuse process. The same applies for the remaining two evaluation dimensions; the importance of the technical and knowledge representation as well as their outcomes are context-sensitive, though the methodology remains unchanged.

Complementarily to these application-related considerations the requirements upon a particular *reuse level* might influence the selection of the evaluation dimensions or their prioritization:

- **Vocabulary**: if the required level of reuse is limited to a simple vocabulary, then the engineering team should focus on the content, availability and application evaluation.
- **Vocabulary and semantics**: here we further differentiate among three sub-levels, i.e. the classification, the properties and the axioms. The higher the level of axiomatization imposed by the application scenario, the more stringent the need for an in-depth

knowledge representation evaluation. In this case, one also recommends the exercise of a technical evaluation from the properties sub-level upwards.

- **Instance data**: again, the engineering team should focus on evaluating the content, the availability and the applicability of the knowledge sources.

4.3.1 Content Evaluation

DESCRIPTION Domain experts and users decide whether the information captured by the ontology is relevant for the scope of the application. In order to have access to this information the programmers are required to provide technical support in form of tools for visualizing the ontology in a language-independent manner. If the ontology is available in a proprietary representation language or in more informal forms (such as Web sites or textual documents), the engineering team might consider implementing utility tools to ease non-technical team members the access to the information. This decision primarily depends on the size of the ontology, the complexity of its structure (e.g., taxonomy vs. axiomatized ontologies) and the human-perceived understandability of the ontology representation language.

If the domain of the ontology is adequate, the domain experts and the ontology users still need to specify missing information. This task can be accomplished for instance by comparing a textual description of the ontology (e.g., its vocabulary or the documentation) with the description of the domain to be modelled. The latter is expressed in the ontology requirements specification in various forms: the name of the domain, the main concepts of the domain, reference documents describing it [62, 214].

A further issue is related to the correctness of the modelled content. While this can be partially achieved by performing an ontological analysis of the formal model (as foreseen by the knowledge representation evaluation below), the focus here is on determining the errors which are not necessarily detectable as modelling faults, but are related to misunderstandings, misinterpretations or questionable assumptions and commitments in the domain. In order to complete this task a textual description of the model is required. This is then studied by the experts with respect to its value of truth. Nevertheless whether this activity is really necessary might be influenced by provenance details. Ontologies authored by trustworthy organizations or the ones commonly used in similar settings are likely to contain a negligible number of errors of content. In these cases the engineering team might assume a satisfactory quality of the ontology as regards correctness and center on clarifying the question of whether the reuse candidate covers a relevant domain and to which extent.

PARTICIPANTS Domain experts, programmers, ontology engineers, users. The domain experts are the major actors in the content evaluation task. Users support them with materials describing the domain of the application which will utilize the ontology. Programmers are responsible for providing tools which ease the access to the analyzed sources, while ontology engineers possess the necessary background for interpreting knowledge representation languages into more human-readable domain descriptions.

RESULTS List of domain-adequate ontologies and an inventory of actions to be taken in order to revise the ontological content. The scope of these actions is principally focused at the following reuse levels:

- **Vocabulary**: at this level the domain experts might suggest terms which are not covered by the evaluated ontologies or might eliminate irrelevant ones.
- **Vocabulary and semantics**: at the classification level the domain experts might discover inconsistencies in the modelling granularity and missing or revisable concepts. Further on, if properties are included in these investigations, possible actions to be taken might be related to insertions of new ontological properties and their revision.
- **Instance data**: this situation corresponds to the content-based evaluation of the underlying terminological base.

SUPPORT METHODS AND TOOLS Besides technologies to access the ontologies in a user-friendly manner this step can be operationalized with the help of methods to compare the textual description of an ontology with the one of the domain to be modelled. Means for articulating the knowledge expressed in forms of ontologies in natural language are essential for determining the coverage and the correctness of the ontological sources. These techniques are naturally related to the area of computer linguistics.

Further on processing these examinations might benefit from the availability of metadata information about the evaluated ontologies (description of the domain, of the main concepts in the ontology, of the type of the ontology and the particularities of its representation) and metrics capturing the similarity between the user-defined keywords specifying the domain to be covered by the ontology and the existing ones.

CONTEXT-SPECIFIC ISSUES This step is equally important independently of the application setting. However, the task in which the planned ontology will be involved in fosters particular types of ontologies: the semantic search/retrieval task can be accomplished meaningfully when ontologies capture information about specialization/generalization relationships in the domain of interest, at least. If reasoning services are relevant for this application scenario (e.g., software engineering), the larger size of the ontology and a higher degree of axiomatization are likely to be beneficial [225]. If the ontology will be used for integration purposes, the ontology should cover at least one of the domains overlapping in the interrelated applications. The usage of the ontology for classification tasks ideally implies the availability of taxonomical information about the domain of interest. In case the ontology is expected to be utilized in relation to information objects such as textual documents, but also multimedia data, the users of the prospected ontology might provide a representative excerpt of the application repository in order to concretize the domain coverage measurements.

4.3.2 Knowledge Representation Evaluation

DESCRIPTION Ontology engineers analyze the quality of the formal model describing a domain of interest in order to discover knowledge representation errors such as inconsistencies or redundancies.[3] In order to accomplish this task they apply existing evaluation frameworks

[3]The terminology used in the ontology evaluation field is not well-established yet. This type of evaluation is sometimes termed as "ontological" or "logical" evaluation in the literature.

(e.g., [79, 89]) which introduce general-purpose criteria for high-quality knowledge formalizations and methods to manually apply them. The real-world feasibility of such knowledge representation evaluation approaches is still a controversial issue in the community. While the theoretical added value of a correct formal model is indisputable, understanding and applying this kind of methods are noted for their resource-intensive nature even for moderately sized ontologies of several hundreds of concepts. This is confirmed by various case studies, which further often question the relevance of these general-purpose methods in the context of application-oriented usability assessments [114].

PARTICIPANTS Ontology engineers.

RESULTS List of correct knowledge formalizations and an inventory of actions to be taken in order to revise the ontological content (insertion, deletion or modification of ontological primitives).

SUPPORT METHODS AND TOOLS Low-level technological support is offered by syntactic validators and reasoners. Further on, the ontology engineers can be aided with tools for visualizing and editing ontologies. For example the method described in [89] is integrated into several ontology editing tools [114]. Graph-theoretic methods can be also applied to detect cyclic definitions or redundancies [79].

CONTEXT-SPECIFIC ISSUES This evaluation dimension is principally relevant for the knowledge representation and for the software engineering scenarios. For machine-targeted scenarios a low-level variant of the evaluation (i.e. syntactic validity complemented by consistency checking as performed by multiple reasoners) is also recommended. If the expected reuse level is that of a vocabulary or instance data, this type of evaluation is not necessary; this does not apply, however, for a reuse level in which the semantics of the source ontologies is preserved and these ontologies are axiomatized. The operation of the knowledge representation task is context-independent and can be accomplished in conformity with the aforementioned research approaches.

4.3.3 Technical Evaluation

DESCRIPTION Programmers are responsible for the integration of the reused ontologies in the overall application system. For the implementation of the ontology the programmers might be required to translate the reused components to a new representation language or to merge several components. Further on, the management of the resulting final ontology is unfeasible in case of poor technological support in terms of APIs, storage systems, reasoners etc. If these activities involve manual processing, the underlying costs are directly proportional to the size and the complexity of the reuse candidates. These parameters might also be responsible for scalability and performance problems, as the functionality of individual services might not be optimized for largely sized ontologies yet. Currently Semantic Web technologies, though rapidly advancing, have not been proved and tested in real-world scenarios to a satisfactory extent. Consequently, ontologies of more than, for instance, 10,000 concepts are likely to be unappropriate for the majority of ontology management tools.

PARTICIPANTS Programmers, ontology engineers. Ontology engineers supply the expertise necessary to handle aspects related to knowledge representation languages and reasoning services. Programmers posses knowledge with respect to various tools for creating, managing and storing ontologies.

RESULTS List of technically suitable ontologies and inventory of actions necessary for the improvement of their technical usability. These activities perform changes at the implementation level of the ontology: translations, partitioning, automatic population. Further on, the technical evaluation might motivate the decision of reusing only less expressive fragments of the considered ontologies, depending on the maturity level achieved by the employed technologies.

SUPPORT METHODS AND TOOLS In this task the participants use various ontology engineering tools in order to analyze their functionality in conjunction with the candidate ontologies. The evaluation itself primarily depends on the level of experience and intuition of the programmers. It is not supported by any particular methods or tools yet. The examination of the tools might be facilitated by standardized benchmarks and test units. Moreover programmers can make use of metadata information describing the tools applied to build the ontologies to be reused.

CONTEXT-SPECIFIC ISSUES This evaluation dimension is relevant for application scenarios in which the ontology is required to be machine-processable. For data or application integration, semantic retrieval and automatic semantic indexing, as well as software engineering it is helpful when the ontology is represented in a semi-structured or structured form in order to ease subsequent translation operations. Further on, tool availability with respect to specific ontology management activities like merging, integration, mapping, alignment, matching is a fundamental requirement if multiple ontologies are considered. Reasoning-focused tasks such as those involved in software engineering or semantic search might require tools which support particular types of inference services.

4.3.4 Application Evaluation

DESCRIPTION Ontology engineers and programmers assess the usability of a potential reuse candidate by analyzing the application scenarios in which the ontology was previously developed or deployed. If no previous usage information is available, this process step is performed by investigating whether the ontology satisfies a set of core application-specific requirements:

- **Integration**: a first requirement in this application scenario is related to the representation language of the ontology. The ontology should be formal, syntactically and semantically valid. Further on, the usage of standard representation languages is beneficial from a tool-oriented perspective. [225] recommends the usage of small, simply structured ontologies. If matching on the basis of ontology structures (i.e. graph-related) is an issue, it is equally important that the ontology is well-balanced and with a sufficient number of inheritance levels.

- **Semantic indexing/annotation**: concepts should be denominated in natural language, while the natural language used in the ontology should be the same as the one used by the users and in the documents to be annotated. Further on, if the indexing is performed automatically, it is essential that the concepts are labeled according to naming conventions and in a linguistically predictable form [169]. The majority of these requirements are naturally fulfilled by linguistic ontologies. Additionally, large-sized ontologies are likely to be more appropriate for this task as small-sized ones, as stated in [225].
- **Semantic search/retrieval**: the ontology should be formal to enable automatic reasoning. Ontology-based query formulation is supported only if ontological primitives are labeled in natural language and if the natural language is appropriate to the prospected application user community. Further on, besides issues of syntactic and semantic correctness, the ontology should provide a rich semantic representation of the domain to improve and refine the retrieval algorithm [169]. If the semantic search involves any form of ontology-based matching the graph structure underlying the ontology should be well-balanced and should contain a sufficient number of inheritance levels.
- **Software engineering**: ideally the domain conceptualization should be highly axiomatized in order to enable the detection of invalid system parameter settings. A second requirement is related to its syntactic and semantic correctness. Similarly to the integration setting, it is essential to use a tool-supported representation language. Small to medium ontologies (maximally 2,000 - 3,000 concepts) are likely to be more advantageous as large-sized ones, due to potential inference performance and maintainability issues.
- **Knowledge representation**: the ontology should be comprehensible by humans. This requirement is optimally satisfied by well-documented and human-readable *lightweight* ontologies for two reasons; on one hand ontological primitives whose meaning is expressed as unambiguous natural language labels, definitions and comments are likely to be easily understood by humans; on the other hand, heavily axiomatized ontologies impose specific comprehension challenges even when expertise in Knowledge Representation is available. Further on, it is essential that the ontology is application-independent, since ontological sources encoding application-specific knowledge do not provide an optimal description of a domain of interest. A third requirement is related to the extensibility of the ontology. Again, lightweight, well-documented ontologies are likely to satisfy this requirement as well.

PARTICIPANTS Ontology engineers, programmers, users. No specific task distribution.

RESULTS List of application-relevant ontologies and inventory of actions required to revise them at implementation and knowledge level.

SUPPORT METHODS AND TOOLS The major activity performed in this process step is the comparison between application scenarios in which the ontology was already used or for

which it was originally designed for with the actual context. There are no standard tools or methods supporting this activity. However, methods to automatically compute similarities between present and previous application scenarios could be used to speed it up, if the corresponding information is available in a structured form. Further on, metadata information capturing the level of formality, the natural language, the representation language and the size of the ontology being analyzed contributes to the enhancement of this task, as the engineering team is not required to acquire this information during the reuse process.
CONTEXT-SPECIFIC ISSUES: This evaluation dimension is equally important no matter which application scenario the reuse process is associated to.

4.3.5 Availability Evaluation

DESCRIPTION The engineering team investigates the constraints related to the usage of an existing ontology. Besides costs and licence conditions, it is important to take into consideration issues like multi-versioning and updates, as well as provenance and level of maturity. If availability issues are fundamental for the goals of the project, the availability evaluation could provide a very efficient pre-selection of the total set of reuse candidates.

PARTICIPANTS Programmers, ontology engineers, domain experts, users. No particular task distribution.

RESULTS Available reuse candidates as well as information about their availability constraints.

SUPPORT METHODS AND TOOLS No tools or methods available. The accomplishment of this task, which is mainly performed by humans, can however take benefit from the availability of ontology metadata information on topics such as versioning, creation and modification date, provenance, license models etc.

CONTEXT-SPECIFIC ISSUES: Availability issues are not related to a particular application context. However, application settings like integration and software engineering impose less constraints on updates and maintenance.

4.4 Ontology Merging and Integration

DESCRIPTION Provided a list of reusable ontologies complemented by a set of required customization operations, the goal of the ontology merging and integration step is twofold: the execution of these operations in order to adapt the ontologies to the application context, and the integration of multiple sources to the target system. This might be realized in form of

- an ontology-oriented integration, in which several sources are aggregated to a single ontology to be used in the system, or
- by directly embedding the customized ontologies to the system components they have been assigned to in the software engineering process.

Beyond this distinction, this step is primarily technological. Tools and APIs are required to transform the ontologies to be reused to a new representation and to merge and integrate them. A conceptual level description of the operations which are required to be performed by these tools is given in [180].

In relation to the increasing popularity of ontologies in various application fields, the Semantic Web community has developed a wide range of ontology management methods and tools in the last decades. However, while the main ingredients for a feasible ontology merging and integration are meanwhile available, the existing solutions can not be applied to arbitrary application scenarios with the same success rate. Apart from the complexity of the generic ontology merging problem, existing dedicated tools— prototypical implementations mostly originating from academia projects—lack an extensive and systematic real-world operation. Consequently they are (implicitly) optimized for a particular class of merging or integration problems (e.g for manageable input sizes, for simple ontological structures). Programmers are required to analyze the suitability of these tools being aware of their strengths and weaknesses, and to decide upon the concrete integration strategy.

From a methodological perspective this reuse phase consists of the following steps:

1. **Ontology customization**: the customization operations specified during the evaluation phase are executed on the selected ontologies.
2. **Tool and method selection**: in this step the engineering team specifies the adequate tools and the integration strategy.
3. **Integration preparation**: the aim of this step is to clearly specify the integration execution activity.
4. **Integration execution**: the tools are executed in the specified order.
5. **Integration evaluation**: the results of the integration procedure and inherently of the entire reuse process are evaluated.

Depending on the outcomes of a particular step the integration workflow can be executed in a linear or iterative manner. This applies primarily to the last three of the aforementioned steps (see below).

PARTICIPANTS Ontology engineers, domain experts, programmers. The exact task distribution is specified at the next level.

RESULTS Final ontology.

SUPPORT METHODS AND TOOLS Currently the integration process is not supported adequately at methodological level. Due to heterogeneity issues there is no integrated environment permitting ontology engineers to access, evaluate and jointly use multiple merging and integration tools in order to obtain the desired results.

CONTEXT-SPECIFIC ISSUES Contextual information such as certain ontology characteristics, the ontology tasks and roles, the reuse level, but also the properties of each method and tool employed are important decision factors for the way the customization, merging and integration of the selected ontologies are performed. This interaction is illustrated in more detail in the description of the four integration stages.

4.4.1 Ontology Customization

DESCRIPTION Before being forwarded to the merging/integration process, the selected ontologies might be subject of particular customization measures which have been specified along the evaluation phase. These operations relate to four of the introduced evaluation dimensions (excluding the availability question) and include changes or extensions at content, knowledge and implementation level. Programmers, domain experts and ontology engineers execute the specified operations and evaluate their outcomes.
PARTICIPANTS Ontology engineers, domain experts, programmers. The task distribution is described in regard to the evaluation dimensions introduced above:

- **Content evaluation**: Domain experts and ontology engineers insert or revise concepts and properties, perform changes at classification level and extract relevant subfragments. These tasks can be accomplished programmatically.
- **Knowledge representation evaluation**: Ontology engineers correct the ontological model at knowledge level according to the findings of this evaluation step. The operations are similar to the ones executed within the content evaluation task. However, they take into account principles of knowledge representation which are not per default considered by domain experts.
- **Technical evaluation**: Ontology engineers and programmers might decide to adapt the ontologies so as for them to fit better to the target technological infrastructure. This includes clustering operations for scalability purposes, knowledge encodings in a given formal language, simplification of the formalizations to permit efficient reasoning etc.
- **Applicability evaluation**: The customization actions suggested as a secondary result of this step aim at improving the usability of the evaluated ontologies as regards the final application context. Complementarily to the technical issues, such operations, performed by domain experts, programmers and engineers, might extend the vocabulary of the sources, simplify them in terms of a lower reuse level, translate them to a new representation language or extend their formalization.

RESULTS Customized ontologies. These are assigned to fit better to the subsequent integration activities.

SUPPORT METHODS AND TOOLS Changes at conceptual level are supported by ontology editing and visualization tools. Implementation issues require dedicated translating tools, including those handling semi-structured inputs. In order to execute pruning operations systematically the engineering team might be aided by tools extracting specific structure- and content-based views of an ontology.

CONTEXT-SPECIFIC ISSUES These induce particular customization operations as explained above.

4.4.2 Tool and Method Selection

DESCRIPTION While a basic feasibility study for the subsequent merging/integration attempt has been completed during the technical evaluation of the sources to be reused, this step complements these initial considerations with an in-depth analysis of the available methodical and technological infrastructure, followed by the specification of the ones chosen to be employed.

As indicated by the empirical findings of the ontology reuse case studies (cf. Chapter 3) this decision should take into account features of the involved ontologies as follows:[4]

- **Syntactic features**: various graph-oriented metrics such as the total size of the ontology, the number of classes, properties, axioms and instances. These have consequences on the performance and scalability of the merging process. Further on, some tools are explicitly targeted at a restricted set of ontological primitives (e.g., they do not consider axioms)
- **Semantic features**: features related to the formal semantics of the representation language and the meaning of the ontology content:
  - readability (i.e. the usage of human-readable concept names)
  - level of formality (e.g., highly informal, semi-informal, semi-formal, rigorously formal [223]).
  - type of model (upper-level, domain ontology, taxonomy, thesaurus etc.)
  - ontology domain (i.e. the domain modelled by the ontology)
  - ontology representation paradigm (i.e. the class of representation languages with respect to expressivity)
  - ontology natural language (i.e. the natural language used for labels, comments and definitions)

Further on, depending on the tool capabilities, the engineering team decides upon the expected outcomes, the pre- and post-processing of the ontologies and the way humans are required to interact with the tools during their execution. The latter is relevant if several tools are jointly applied to accomplish the merging/integration tasks or if they are operated semi-automatically.

An analysis of the dependencies between source and target ontologies in reuse processes revealed that the latter make use of the source vocabularies to a large extent (cf. Chapter 3). In the same time additional ontological primitives like properties and axioms are not supported explicitly in many knowledge sources and their integration to the target ontology is problematic because of the comprehension difficulties encountered by domain experts in getting familiar with such complex structures. On the other hand, exploiting the "lowest common denominator" of the source ontologies i.e. their vocabulary, proved to be extremely useful in

[4]These ontology-related features are explained in detail in the next chapter, in which we introduce the ontology reuse metadata model.

our reuse experiments . A solution towards a generic (semi-automatical) method for ontology reuse might be thus an incremental process which concentrates on the vocabulary of the input sources and subsequently insert additional information (semantic relationships, axioms etc., if available explicitly) corresponding to application needs. Such a process does not tap the full potential of current technologies in the corresponding research areas, but implies significant cost reductions associated with less efforts for manually pre- and postprocessing of the results. This design principle is further applied in the methods and tools we realized for the support of the integration process (cf. Chapters 6 and 7).

PARTICIPANTS Programmers with expertise in the area of ontology matching, merging and integration.

RESULTS Methods and tools to operationalize the integration of the ontological sources in the final application setting, execution workflow, additional tools such as thesauri and lexica, translators etc.

SUPPORT METHODS AND TOOLS To date the tool evaluation step is not supported technologically, but is inconceivable without documentary information on the tool candidates. This step can be further speeded-up if this information is represented in an uniform schema.

CONTEXT-SPECIFIC ISSUES The decision upon the matching, merging and integration technology applied to perform this reuse phase depends primarily on the required reuse level and on the features of the input ontologies. A linguistics-based approach is definitely helpful in scenarios which utilize ontology-driven services such as semantic annotation, query formulation, and human-targeted knowledge representation. However, it is reasonable only if the ontologies are labeled using pre-defined patterns and if the corresponding natural languages coincide. Structure-based merging and integration is adequate for integration and software engineering and on ontologies containing at least taxonomical relationships.

4.4.3 Integration Preparation

DESCRIPTION This step gives full particulars about the prospected integration activity. It is concerned with the configuration of the technical infrastructure of the procedure to be executed in the next step. Besides selecting characteristic execution parameters the preparation step clearly specifies the human-machine interactions involved in semi-automatic integration workflows. Further on, the programmers are required to detail the execution order and the dataflow among the utilized tools.

PARTICIPANTS Tools experts i.e. programmers.

RESULTS Configured, ready-to-use integration environment.

SUPPORT METHODS AND TOOLS The configuration of the tools benefits from careful documentation and user manuals.

CONTEXT-SPECIFIC ISSUES None.

4.4.4 Integration Execution

DESCRIPTION This step is dedicated to the actual execution of the ontology matching, merging and integration tools on the basis of the input parameters specified so far. During the execution of each tool domain experts and programmers are expected to support the required tool interaction and to evaluate the correct accomplishment of each integration sub-task. The detection of major problems in the configuration of the tools imposes a re-iteration of the process at the previous step.

PARTICIPANTS: Programmers, domain experts. The latter are involved in the evaluation of the preliminary results.

RESULTS: Preliminary application ontology.

SUPPORT METHODS AND TOOLS: Besides the tools completing the integration, this step takes advantage of those ones which allow a direct and flexible monitoring of the process. This includes usability aspects, but also the possibility to interact with the tools during the process execution and to evaluate intermediary results.

CONTEXT-SPECIFIC ISSUES: None.

4.4.5 Integration Evaluation

DESCRIPTION The engineering team evaluates the results of the integration operation and of the overall reuse process. The way this task is performed depends on the ontologies which were involved in the integration. If the integration covered the complete range of ontologies built during the overall engineering process (i.e. the reused ones, but also the ones manually built, or the ones extracted from text documents or other sources), the evaluation is performed in accordance to the criteria specified during the domain analysis. It might include application-specific tests, but also domain or modelling-related ones. By contrast, if only the reused ontologies have been subject to integration, this task is reduced to examining whether the merging and integration were executed correctly, thus being principally a matter of the tools and methods utilized.
PARTICIPANTS: Programmers, domain experts, ontology engineers, users.

RESULTS: Final ontology.

SUPPORT METHODS AND TOOLS: The methods and tools supporting this task highly depend on the application scenario. From a technical perspective the ontology is integrated into the target environment and used for the designated task(s) in order to test its usability. The evaluation requires test units and metrics for comparing between gold standards and actual outcomes.

CONTEXT-SPECIFIC ISSUES: The application scenario in which the ontology is embedded imposes particular evaluation strategies, which are, however, out of the scope of this work.

4.5 Requirements for Methodology-Related Support Methods and Tools

In order to enhance the added value of the proposed methodology in arbitrary ontology reuse scenarios—with respect to an efficient costs management, but also with respect to its user-friendliness—we tackled the problem of which methods and tools could be helpful to aid the ontology engineering team during the methodology application. A first methodology-independent step towards the specification of these requirements has been made during the ontology reuse feasibility study in the previous chapter. The analysis of the case studies and the conclusions of the literature survey pointed out the deficiencies of current ontology reuse-relevant technologies, be that tools to create, revise or visualize ontological content, but also means to manage them in terms of matching, merging, integration, pruning and translating. Further on, the feasibility study highlighted missing methods and tools, which are likely to ease the reuse process:

- the lack of fully-fledged ontology repositories
- the lack of a comprehensive metadata model for ontologies and of tools creating or managing this information

During the design of the methodology these requirements were complemented with an inventory of methods and tools which, in contrast to the aforementioned ones, are expected to contribute to the operationalization of this particular process. While we do not exclude their usability beyond the boundaries of this work, the methods, techniques and tools we describe in the following are primarily methodology-relevant. Their prototypical design and development will be described in the next chapters.

The requirements for ontology discovery support specified in the previous chapter have been reenforced during the design of our methodology. They are summarized in Table 4.3.[5]

With respect to the evaluation of the reuse candidates, the feasibility study revealed the necessity of methodological and technological support for the completion of this non-trivial step. The requirements emerging from this study (Table 3.7) have been refined in compliance with the proposed ontology reuse methodology as depicted in Table 4.4.

Requirements $R7.9$ to $R7.11$ state the general need for quantified measures to compare and rank ontologies based on a set of pre-defined features. Examples of such measures are provided in the next chapter, in which we elaborate a way to perform application-oriented ontology evaluation in a computer-aided manner. $R7.12$ and $R7.13$ address the question of contradictory application-driven requirements for ontologies, which can be answered with

[5]We adopt the same numbering scheme as in Chapter 3.

No	Criterion
R6.1	Semantically enabled ontology metadata
R6.2	Ontology repositories
R6.2.1	Ontology-based search
R6.2.2	Browse and navigation
R6.2.3	User rating methods for ontologies
R6.2.4	Attestation methods for ontologies
R6.3	Dedicated crawlers

Table 4.3: Requirements for Ontology Discovery Support Methods and Tools

No	Criterion
R7.1	Semantically enabled ontology metadata
R7.2	User rating methods for ontologies
R7.3	Attestation methods for ontologies
R7.4	View-enabled ontology editors
R7.5	Ontology visualization tools
R7.6	Textual descriptions of ontological content
R7.7	Query engines and reasoners
R7.8	Ontology matching and alignment tools
R7.9	Methods to compute similarities between ontologies
R7.10	Methods to rank evaluation results
R7.11	Methods to aggregate disparate evaluation results
R7.12	Methods to quantify costs and benefits of ontological features
R7.13	Methods to perform cost benefit analysis

Table 4.4: Requirements for Ontology Evaluation Support Methods and Tools

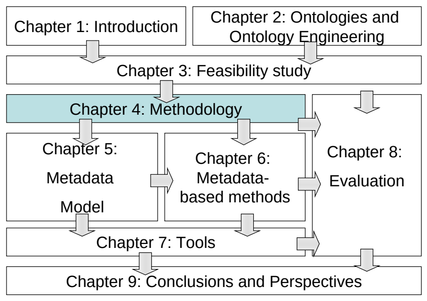
the help of cost/benefit analysis instruments. This issue is not covered by our research so far. In turn, the approach to automatize the ontology evaluation task introduced in the next chapter has shown to perform well as compared to human judgement; a deeper analysis of the consequences of such contradictory requirements will be pursued in the future.

Starting with the results presented in Chapter 3 we computed an extended requirements specification for ontology merging and integration which is illustrated in Table 4.5. This emphasizes the need for a novel approach to merging and integration from a methodological perspective. This concentrates on two notions. First we aim for tools and methods which allow a customization of the matching process depending on the current reuse context (e.g., the features of the input ontologies or the reuse level); secondly, we point out the need for an incremental merging and integration process, which permits users to directly interact with preliminary results in order to minimize the post-processing efforts (cf. R8.7).

No	Criterion
R8.1	Translation tools
R8.2	Information extraction tools
R8.3	View-enabled ontology editors
R8.4	Ontology matching tools
R8.5	Ontology merging and integration tools
R8.6	Context-sensitive matching approach
R8.7	Incremental merging and integration methodology

Table 4.5: Requirements for Ontology Customization, Merging and Integration Support Methods and Tools

4.6 Summary


Chapter 1: Introduction
Chapter 2: Ontologies and Ontology Engineering
Chapter 3: Feasibility study
Chapter 4: Methodology
Chapter 5: Metadata Model
Chapter 6: Metadata-based methods
Chapter 8: Evaluation
Chapter 7: Tools
Chapter 9: Conclusions and Perspectives

This chapter introduced our methodology for ontology reuse. It contains a fine-grained description of the reuse process and a suite of application-oriented best practices and guidelines. Further on, it points out techniques and tools required to aid ontology engineers and users during this challenging process. The last section distilled the requirements specification introduced in the previous chapter in a methodology-specific direction, with the result of having a deeper understanding of which methods and techniques are likely to increase the usability of the proposed methodology in real-world scenarios.