University of Innsbruck
Semantic Technology Institute

A Manual Annotation Approach based on Ontologies

Bachelor Thesis

Christian Ammendola
Matriculation number: 0316635

Supervised by Dr. Elena Simperl
Co-supervised by Tobias Bürger

March 18, 2008

Abstract

Ontology-based annotation is a central point in the Semantic Web. While many different methods for ontology-based annotation exist, not many focus on support for non-experienced users. In this thesis we present an approach for providing aid to end users in the process of manual creation of ontology-based annotations. In this process, non-experienced users are often faced with the problem which ontology elements to use for annotation or how to extend an ontology if elements are missing. In order to address these problems this thesis proposes an approach for supporting users in selecting ontology elements and extending ontologies if needed. This approach is based on existing semantic techniques: the element selection part is mainly based on semantic search and semantic autocompletition techniques and the ontology extension part on a work-embedded ontology engineering mechanism.

Contents

1 Introduction

Since the beginning of the World Wide Web in the 1990s, it has evolved to the biggest knowledge base nowadays. Due to the fact, that the meaning of the information within this knowledge base is not "understandable" for machines, it can be difficult and time consuming to retrieve the desired information in this bulk of data. An approach for a solution to this problem is provided by the Semantic Web, a vision of Tim Berners Lee[1] [2]. The idea is to extend the World Wide Web by assigning a well-defined meaning to the data stored in it that is accessible and processible for machines.

The assignment of machine readable meaning to the data on the Web can be realized by the creation of metadata containing semantic information. This can be done by the help of ontologies that represent formally defined collections of terms, relations among them and inference rules[2]. Metadata is then created by the use of the defined terms, assigning a certain meaning to the resources[3] of the Web. Once such additional information is created, retrieval mechanisms can be used that consider the meaning of the terms assigned to the resources as well as the additional information provided by the defined relations and the inference rules of ontologies[4].

The process of creation of such metadata is referred to as semantic annotation [6] and is a central point within the Semantic Web vision. For the creation of this additional information manual, semi-automatic, and automatic approaches exist. Semi-automatic and automatic annotation methods are the faster alternatives necessitating only of low or no intervention of humans for the creation of the metadata. They can be realized by the help of information extraction techniques. But not all semantic metadata can be annotated automatically. The problem is that not all kind of information

[1] The inventor of the World Wide Web.

[2] A detailed presentation of ontologies is provided by [5].

[3] The term resources is used to refer to documents on the Web such as Web pages, images, etc.

[4] Such retrieval methods are referred to as *Semantic Search* and are presented in appendix A.

can be extracted from resources, which is especially true for multimedia resources. Considering the example of a set of images depicting churches and the need to annotate the name of the church and the place where it is situated we encounter this problem. Because an automatic extraction of this kind of information from an image is difficult, manual annotation is needed.

In this thesis we present a methodology for the manual annotation of resources. Due to the fact, that the creation of metadata can be a difficult and time-consuming task, the methodology should provide an approach for supporting users in its execution.

Before going over to the motivations of the thesis, we first make some remarks about the used terminology: In the following we are going to use the terms *concept* to refer to ontology terms and *elements* or *ontology elements* to refer to concepts, relations and fixed instances[5] of ontologies. The term *user* or *annotator* is going to be used to refer to the person creating the semantic metadata. Further terms used in this thesis will be introduced when needed.

Now that we have introduced the most important terms being used in the following, we will present the main motivations for the development of the methodology:

- *Choose the right concept:* The selection of the right concept for the annotation of a resource can be a difficult task, especially, when working with huge ontologies or for users new to the used ontologies or ontologies in general. Considering the case of an image representing a car and an ontology providing the concepts *vehicle*, *fourWheelVehicle*, and *car*. If the user searches for a vehicle concept he perhaps does not find the concept car what could be the most adequate one. For such cases the methodology should provide a mechanism to support the user in the selection of appropriate concepts.

- *Annotation accuracy:* Annotation accuracy can also be an important factor in the creation of metadata, because it can enable more precise

[5] An instance being part of an ontology

search results. With accuracy we mean metadata containing an appropriate level of information. As an example, we consider the case of a set of images representing historical buildings that has to be annotated. In order to enable more precise search results it could be appropriate to annotate them by specifying the name of the building, the location where the building is situated, and the year of construction. So users can search for images about historical buildings of a certain year, in a certain location or with a specific name. In order to support the user in the creation of accurate semantic metadata a mechanism for proposing elements to the user, he can use to annotate a specific resource, would be helpful.

- *Vocabulary extension:* The ontologies used for annotation sometimes do not provide all the elements needed for the creation of metadata. A possible reason for this is that the annotation needs can change over time. As an example, we consider the same case of a set of images depicting historical buildings introduced before and the need to annotate them. If the set of images is extended by images representing bridges, and the used ontologies do not include a concept *Bridge* it would be appropriate to give the user the possibility to add the missing concept in an easy way.

Based on these motivations we formulated the following objectives of the methodology:

(1) Provide a mechanisms to support users in the selection of ontology elements during the manual annotation process.

(2) Provide a mechanisms allowing users to extend the ontologies during the annotation process.

Based on the presented motivations and objectives we designed a methodology providing a main structure for supporting manual annotation composed of two parts, namely the *element selection* and the *element addition* part.

The former provides a solution approach for objective (1) and the latter for objective (2). We name the methodology *element Selection and element Addition methodology* or shortly *SA methodology*.

In the next sections we are going to present the annotation methodology in more detail. In section 2 the context of the methodology is introduced, in section 3 the methodology and its parts and components are discussed and in section 4 some concluding considerations are presented.

2 Methodology Context

The scope of the methodology is represented by a content base consisting of resources and the need to manually annotate it. The entities involved in that scenario are *resources*, *ontologies*, *annotators* and an *annotation tool*. Figure 1 provides a graphical overview of that scenario.

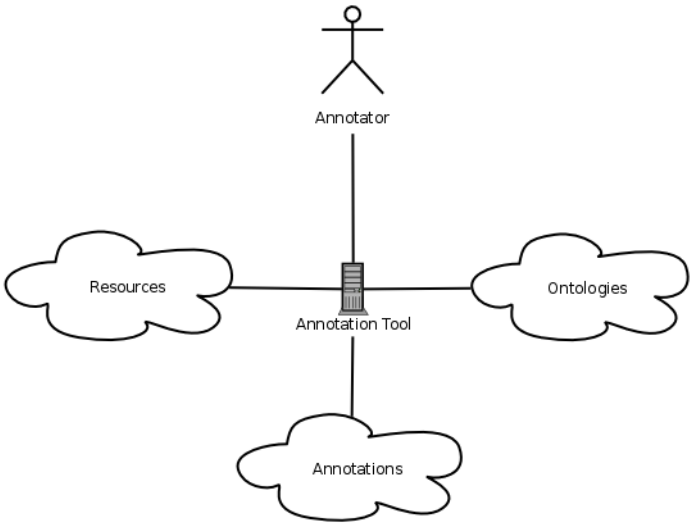

Figure 1: Methodology Context

The annotation tool plays a central role in the depicted methodology

context by interconnecting all the other involved entities. It is a software system that can be use by an annotator to create semantic metadata about resources by the use of a set of ontologies. Due to the fact, that this tool constitutes the interface between annotators and the used ontologies, it can provide support to the user in working with them. Hence, the realization of the SA methodology, aiming to support users in the selection and addition of ontology elements, has to be integrated in the tool itself. This scenario is illustrated by figure 2 where the annotation tool includes the SA methodology, consisting of the element selection and the element addition part, for interacting with the underlying ontologies used for the annotation. In this way the user gets support by the SA methodology layer in selecting elements for annotation and extending the ontologies if necessary.

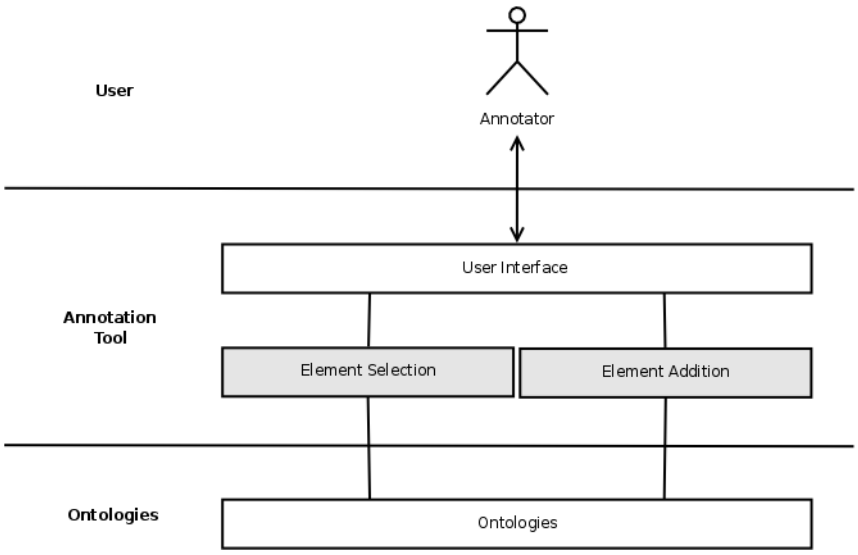

Figure 2: Methodology Realization

Now that we have seen what the context of the SA methodology is, in the next section we are going to present the structure of the methodology in more detail.

3 SA Methodology

In this section we present the SA methodology in more detail. The main structure of the methodology consists of an element selection and an element addition part as depicted in figure 2. Based on this rough structure we selected a set of components for the two parts in order to achieve the objectives presented in section 1. Questions we deal with in the process of selection of adequate components are *how can annotators be supported in the selection of the needed ontology elements* and *how can annotators achieve an extension of the ontologies during annotation time?* An overview of the chosen components is provided in figure 3.

For the selection part we identified four components that are the *Ontology Browser*, the *Ontology Filter*, the *Recommendation System* and the *Semantic Search with Autocompletition*. Not all of them need to be realized in order to achieve our goal of supporting annotators in the selection of ontology elements. However, a combination of more than one component or all of them can result in a better support.

The two main components are the semantic search with autocompletition and the recommendation system. The former enables the search of concepts using semantic retrieval techniques based on approaches presented in appendix A. The latter presents recommendations to the user of ontology elements that can be used for the annotation of a specific resource and is based on the recommendation system mechanism presented in appendix B. In order to achieve the objective of the element selection part only one of these two components would be necessary, but from our point of view the realization of both of them would represent a good combination.

The ontology browser provides the possibility of a more direct selection of ontology elements. The capabilities of this component alone is not enough to fulfill our goal of supporting users in the element selection, because the user gets no aid in the selection. The ontology browser is thought to be implemented in combination with one of the other components. Finally, the ontology filter enables to restrict the set of elements being part of the

ontologies used for annotation. This component does not provide capabilities of element selection. It has to be combined with one or more of the other components and it is intended to augment their service.

The element addition part of the methodology includes the *Classification Support* and the *Ontology Extension* components. The latter enables the possibility to add elements to ontologies during annotation time and is based on the ontology engineering approach presented in appendix E. It represents the core component of the element addition part and is necessary in order to achieve the objective of extending ontologies. In order to support the user in classify newly added elements the classification support component can be applied additionally. The approach selected for this component is based on a recommendation system and on the classification approach used in Ontogame[6]. The classification support component has not necessarily to be implemented and is thought as amplification of the ontology extension component.

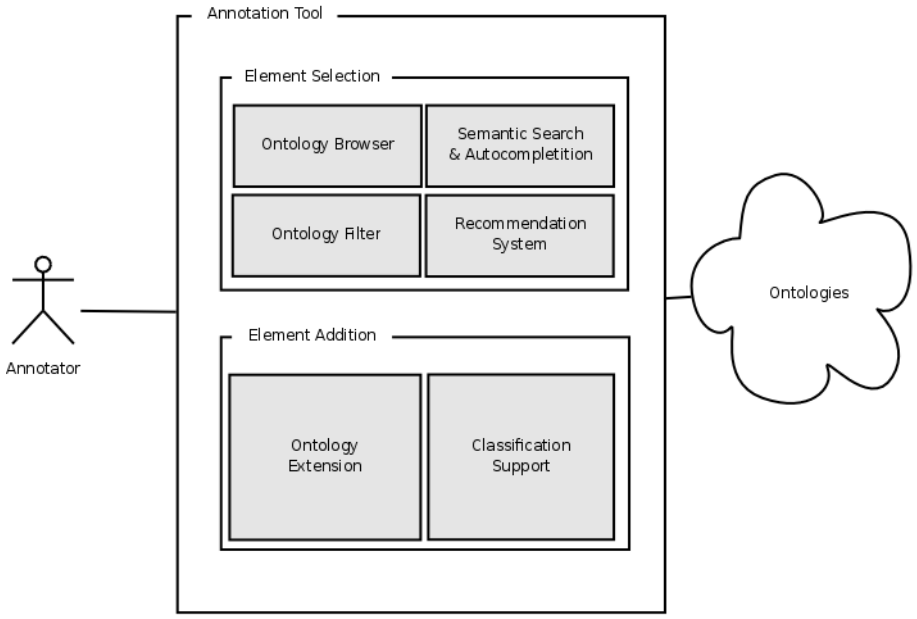

Figure 3: Methodology Components

[6]http://www.ontogame.org/

In the next sections the introduced components are going to be described in more detail. The first four components being part of the element selection part are described in section 3.1, whereas the last components belonging to the concept addition part are presented in section 3.2.

3.1 Element Selection

The selection of the right concept for the annotation of a resource is not always a simple task. Imagine to be confronted with the problem that you have to annotate resources but you are not familiar with the ontologies at your disposal. If the number of elements of that ontologies is high, it could take you a lot of time to find the needed ones. But it could also be the case that you choose improper concepts, that your annotations are not very detailed or that the created metadata is not specific enough. However, not only unexperienced users need support in the process of annotation. Dealing with large and complex ontologies can be time consuming also for adept users. In order to support annotators in the process of metadata creation, an annotation tool can provide aid in the selection of the right ontology elements.

In the following we present the components of the SA methodology concerned with supporting the element selection. As aforementioned not all the components need to be implemented in order to achieve the objective of this part of the methodology. Figure 4 provides a graphical overview of the case of all components being realized. The semantic search with autocompletition, the recommendation system and the ontology browser are concerned with the selection of ontology elements. However, the ontology filter component is concerned with restricting the set of ontology elements the other three components are working with and is situated inbetween them and the underlying ontologies.

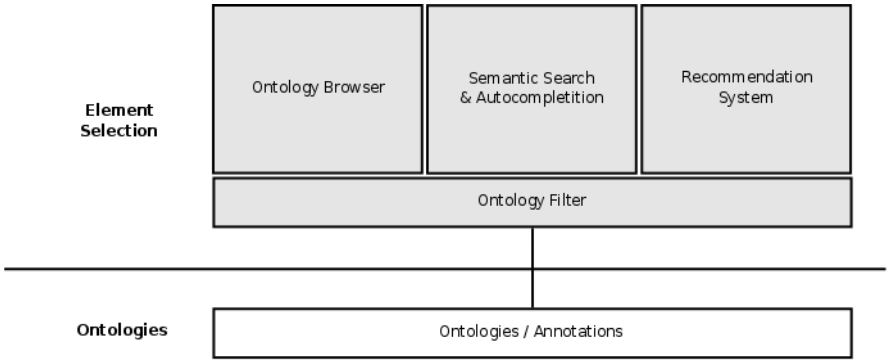

Figure 4: Element Selection

Now that the different components of the concept selection part are introduced, a more detailed discussion about them is provided by section 3.1.1 to 3.1.4.

3.1.1 Ontology Browser

The main goals of the ontology browser are to provide the facility of a direct selection of the desired ontology elements, as well as, the possibility of browsing them. The former capability is intended for experienced users to get quickly to the desired ontology elements needed for the annotation. But not only adept users take advantage of this component. The browsing possibility allows new users to navigate through the elements of the ontologies in order to get familiar with them.

An appropriate representation possibility for browsing ontologies is a *tree representation* that constitutes a suitable method for the visualization of hierarchies. It enables a fast and easy mechanism for browsing and selecting elements of a taxonomy that in our case are concepts and fixed instances. Figure 5 depicts a possible tree representation of an ontology created with the help of Protege[7]. The tree representation of Protege normally includes only concepts, so we extended the graphic to represent also instances. The

[7]Protege is an ontology editor: http://protege.stanford.edu/.

elements with a black point are concepts and the elements with a gray point are fixed instances. The structure of the represented taxonomy is based on the PROTON[8] top module but is not including all of its concepts. This ontology will be used for further examples in the following sections.

Figure 5: Tree Representation of Concepts and Fixed Instances

Additionally to the selection of concepts and fixed instances, the ontology browser should provide the possibility to select relations of ontologies that can be used for annotation. Due to the fact, that also relations could be organized in hierarchies, a tree representation would be appropriate also for these ontology elements. Figure 6 provides an example of a possible representation also created with the help of Protege. Also these elements will be used for further examples.

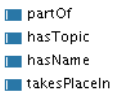

Figure 6: Tree Representation of Relations

[8]PROTON is an ontology: http://proton.semanticweb.org/

3.1.2 Semantic Search with Autocompletition

In section 3.1.1 we discussed the component allowing a direct selection of ontology elements by browsing them. In this section we describe how concepts and fixed instances can be selected by a search functionality supported by autocompletition. The selected approach is based on semantic search and semantic autocompletition techniques and can be refined by a ranking of the matching elements. An introduction to search and autocompletition methods can be found in appendix A and D, and an approach for the calculation of the semantic relevance that can be used for ranking is described in appendix C.

Both adept users and inexperienced ones can profit from the capabilities of a semantic search, in order to achieve a quick retrieval of elements. By simply typing a search string autocompletition on-the-fly[9], semantic search and ranking facilities try to find concepts and fixed instances that could be of interest to the user.

Figure 7 depicts a possible workflow for the realization of the functionality of this component. The different steps of the workflow are listed in the following:

- The process is initiated by the user that submits an input string to the *semantic autocompletition* module. More precisely, the semantic autocompletition facility detects changes in the inserted string, and triggers a semantic search. While the user is typing, new search invocations are performed. The results of the search are presented to the user.

- The retrieval of concepts is performed by the *semantic search* module, that builds a set of matching items. Thereby, not only direct matching concepts are considered, but also concepts being in a certain relation (e.g. sub-concepts, siblings, etc.) with the directly matching ones are included in the result.

[9]Concepts are proposed to the user while typing the search string. The result set is refined with each character newly inserted.

- Before the search results are returned to the semantic autocompletition module, the *semantic relevance* module ranks the found concepts which results into the following advantages: Firstly, the set of items can be presented in an ordered way, and secondly, in the case that the search result is to big, and the number of elements has to be reduced, the less relevant ones can be removed.

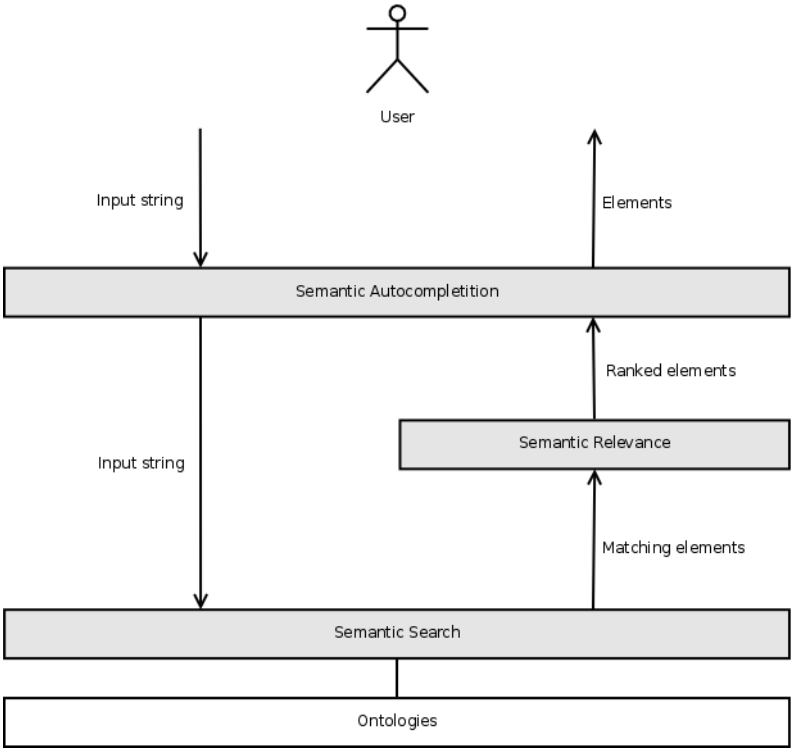

Figure 7: Semantic Search with Autocompletition

In order to provide the possibility to search for ontology elements a simple input field can be used, into which the user types an input string. The results of the search could be presented as a list or in a hierarchical representation. The latter would provide to the user more information about the found elements by presenting also its context. We now present some examples of

semantic search with autocompletition using a hierarchical representation of the search results. Thereby, the input string is autocompleted to categories of view projections. View projections are hierarchies constructed out of concepts and instances of ontologies and its elements are called categories. An advantage of using view projections is that these hierarchies are probably more understandable for the user than the hierarchies directly provided by ontologies. For a more detailed presentation of view projections see appendix A.

Figure 8 depicts the view projections we constructed out of our ontology of figure 5. The *Event Type* view has as root elements the concepts *Event* and *Situation* and the relation used to find the elements being part of the view is the sub-concept relation. Hence, starting from the root elements concepts being sub-concepts of them are added to the view. Also for each new element being part of the view its sub-concepts are added. This is repeated until no new elements are found resulting in the depicted hierarchy. The *Event Location* view has as root elements the instances of the concept *Country*, that in our case is only *Europe*. The hierarchy is constructed using the part-of relation. Due to the fact, that the instances *Austria* and *Italy* are defined to be part of *Europe* in our ontology, they are associated as its children in this view. Finally, *Vienna* and *Rome*, being part of *Austria* and *Italy* respectively, are also added.

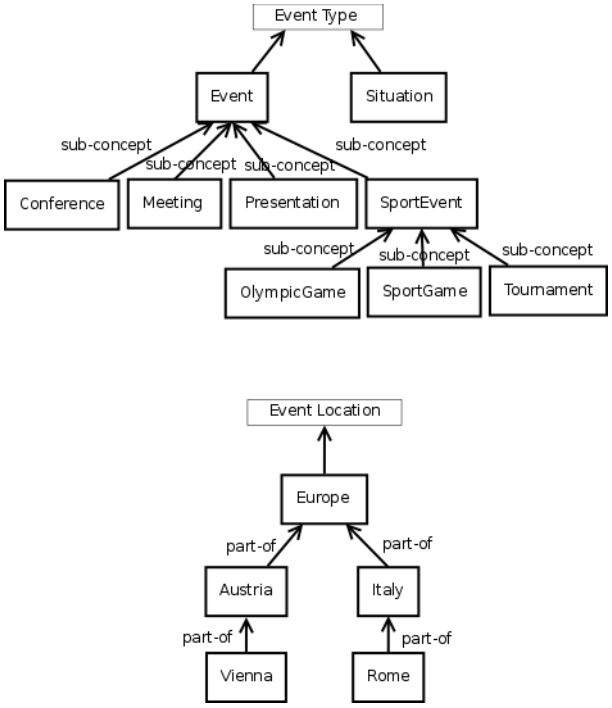

Figure 8: View Projections

Figure 9 depicts three examples of semantic autocompletition. In these examples the text in the white boxes represents the input string of the user and the results are presented in the gray boxes. In the following the three scenarios are described in more detail:

- *Example 1:* The user starts typing the string *Tourn* which triggers a semantic search. The category *Tournament* is found because the inserted string is a sub-string of its label. Based on this category other categories that could be proposed to the user. In this example the sibling relation between categories is used returning to the user also the categories *SportGame* and *OlympicGames*. The result set is presented in a hierarchical structure including all parents of the found elements until reaching the root of the view.

- *Example 2:* The user inserts the string *Talk*. There are no categories

including the sub-string talk in its label, but *Presentation* is specified to be a synonym of talk in our ontology. Therefore, it is proposed to the user including also its siblings.

- *Example 3:* The user inserts the string *Europe*. The category *Europe* is found. By using the part-of relation other categories are retrieved. Due to the fact, that the categories *Austria* and *Italy* are in a part-of relation with *Europe*, they are also presented to the user.

Figure 9: Autocompletition Examples

3.1.3 Recommendation System

The semantic search with autocompletition described in section 3.1.2 enables a fast and easy way to get to the desired concepts and instances, but sometimes users don't exactly know what elements to use for annotation or they use only a small part of the available ontology elements for the description of the resources. For such cases, a recommendation system can present proposals to the user about ontology elements that can be use for the annotation of a specific resource. The facilities provided by such a system can be helpful for experienced and unexperienced users in order to create accurate annotations.

For the calculation of recommendations there exist several methods. Approaches described in appendix B are profile-based recommendations, similarity-based recommendations, and rule-based recommendations. Another possibility could be an annotation-based mechanism for analyzing the available metadata and extracting relevant elements to be proposed. In order to realize a system producing high quality recommendations, the combination of several approaches in one application would be appropriate.

Figure 10 depicts a possible structure of a recommender system. After getting information such as user profile information or information about the resource to be annotated the system searches for related ontology elements and returns them. Different approaches for the calculation of related elements can be combined, so for example a system including a similarity-based and a rule-based approach can be implemented or a system including only a profile-based one. Based on the application context the most appropriate approaches can be realized.

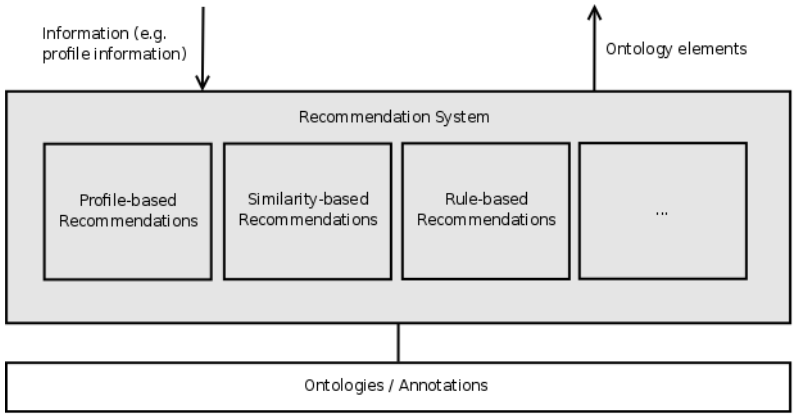

Figure 10: Recommendation System

Now that we have learned about some recommendation approaches and we have seen a few possibilities to combining them into one system we are going to present some examples:

- *Profile-based recommendations:* In this example we consider a user doing annotations using the ontology presented in figure 5 on page 11. The annotation tool used by this user includes a system for analyzing the users behavior checking for often and recently used concepts. So if the user prevalently annotates resources using sub-concepts of *SportEvent* the system could recommend the concepts *OlympicGames*, *SportGame*, and *Tournament* to the user. Changes in the users behavior would result in different recommendations.

- *Similarity-based recommendations:* In this example we consider a user annotating a resource about a presentation. If there already exist annotations about presentations and the relation *hasTopic* is generally used in these annotations, then *hasTopic* can be proposed to the user. Changes in the metadata could result in different recommendations.

- *Annotation-based recommendations:* In this example we consider a recommender system analyzing all the available annotations. Results of the analysis are that most of the resources are annotated using the concepts *SportGame* and *Tournament*, so the recommendations could be the concepts *SportGame* and *Tournament* as well as its siblings that in this case is *OlympicGames*. Changes in the metadata could result in different recommendations.

The presented examples use simple approaches for the retrieval of relevant ontology elements. The use of more sophisticated methods could improve the recommendation quality.

3.1.4 Ontology Filter

Browsing through ontologies can be fast and effective for users familiar with them. But it can also be time consuming when working with lots and large taxonomies, especially if the needed concepts are spread all over the ontologies. A filter capability can help in such cases by restricting the set of

concepts the user is working with, resulting in a more compact representation.

The ontology filter component does not provide element selection capabilities such as the other three components of the element selection part, therefore, it has always to be implemented in combination with one or more or them. The fundamental idea concerning this component is that it can be interposed between the ontologies and the other components of the methodology. Hence, all constituents, such as the ontology browser, the semantic autocompletition and the recommendation system can take advantage of the ontology filter, by providing the benefit of their capabilities coupled with that of the filter. Figure 4 on page 10 provides a graphical overview of the idea in a scenario where all the components are implemented.

For the definition of a filter we propose two approaches, that are the *manual filter* and the *automatic filter*.

The manual filter definition constitutes the possibility to manually specify concepts being part of the set of elements used for doing annotation. Thereby, single concepts or whole hierarchies can be included.

The idea of automatic filter definition is to support the user in the selection of the concepts to be included and is based on the use of a profile-based recommendation system as presented in appendix B. Depending on stored user preferences or his behavior the set of concepts is automatically selected. As an example, we consider a user using our ontology of figure 5 on page 11 for doing annotation. Taking the case that this user often annotates sport events and specifies also its location, the automatic filter definition could include the concept *SportEvent* and its sub-concepts as well as the concept *Location* and its sub-concepts and fixed instances.

We propose to combine the manual and the automatic filter into one solution. Therefore, if a user creates an automatic filter and he wants to add further concepts not included automatically, he can do this manually. Referring to the example abovementioned of a user often annotating sport events, the automatic filter would not include the concept *Conference*. If the user

additionally wants to have this concept, he can add it manually. This scenario is depicted in figure 11 representing the ontology of figure 5 on page 11 after the application of the described filter example. The hierarchy includes the automatically selected concepts as well as the concept *Conference* that is included manually.

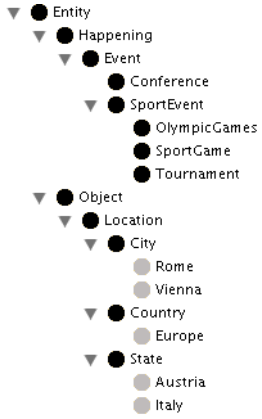

Figure 11: Ontology Filter Example

3.2 Element Addition

So far we presented an approach supporting the selection of ontology elements in a process of manual annotation. However, the needs for metadata creation can change over time and it can happen that the used ontologies do not include a needed concept. As an example, we consider a user annotating a resource about a music event using the ontology of figure 5 on page 11. The user wants to state that the resource is about a concert, but the ontology does not provide an adequate concept for doing this. For such cases it would be appropriate to have the possibility to add the needed concept immediately. In order to face this problem this part of the methodology provides an approach for the addition of new concepts and fixed instances during annotation time.

When extending an ontology by adding new elements the user is faced with two important tasks: *How to label the new element* and *where to put the new element within the existing taxonomy.* In order to approach these problems we developed an approach consisting of two components: the *Ontology Extension* component and the *Classification Support* component (see figure 12).

The former enables the user to add new concepts and fixed instances to ontologies during annotation time. It is based on the collaborative and work-embedded ontology engineering method presented in appendix E. Work-embedded means that users have the possibility to add new elements to ontologies while working with them. Collaborative is referred to the fact, that the ontologies are used by a group of people that are all allowed to extend them and also to reuse and adapt[10] elements added by other users. Based on the reuse and the adaptations a consensus about the label and the classification of the new items is reached over time. Hence, this first component is concerned with the actual process of ontology extension and is needed in order to achieve the objectives of the element addition part of the methodology.

The second component is intended for supporting users in the classification of the new ontology elements during the process of extension. As an example, we consider the aforementioned case of a user annotating a resource about a music event. The user decides to add a concept *Concert* to the ontology and has to decide where to situate the element within the existing taxonomy. With the help of the classification support component the user should find an adequate extension point[11], that in this case could be the concept *Event*. For doing this the classification supporter component combines two approaches. One approach provides aid to the user by automatically proposing him a set of extension points he can use for the classification. This capability is based on the use of a profile-based recommendation system pre-

[10] Adaptations of the labels or the classification can be performed.

[11] We define an extension point to be an ontology concept. The user can classify a new element to be a sub-concept or and instance of a specific extension point.

sented in appendix B. The second approach provides aid to the user in the manual selection of an extension point and is based on the classification approach used in Ontogame[12]. The implementation of this component is not mandatory in order to achieve the objectives of this part of the methodology but from our point of few the implementation of both components represents a good combination.

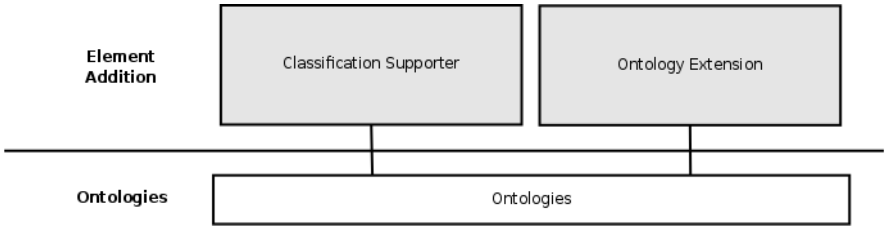

Figure 12: Concept Addition

In the following sections the two components are treated in more detail. In section 3.2.1 the ontology extension component is presented and in section 3.2.2 the classification support component.

3.2.1 Ontology Extension

As described in appendix E the creation, maintenance, and evaluation of ontologies are activities treated by the area of ontology engineering. A common approach for ontology engineering methodologies includes a requirements specification, a conceptualization, an implementation, and an evaluation phase [5]. Such approaches are effective for the creation and extension of ontologies, but they are too time consuming for the addition of ontology elements during annotation time. In order to achieve this goal a more work-embedded solution is necessary.

Our proposal is thought for the extension of ontologies by concepts and fixed instances during annotation time. It is based on the ontology engi-

[12]http://www.ontogame.org/

neering methodology presented by [4], a collaborative and work-embedded approach for the extension of ontologies named *Ontology Maturing*. It consists of four phases that are the *emergence of ideas*, the *consolidation in communities*, the *formalization*, and the *axiomatization* phase. For a more detailed presentation of the model see appendix E. For the solution of our problem we use only the first three phases and adapt them to our needs. Therefore, our ontology extension process is composed of the following two phases:

1. *Emergence of ideas and classification:* In this first phase new ontology elements are created. Thereby, the user has to decide if a new element is a concept or an instance. The newly added element has also to be classified. Concepts can be defined to be sub-concepts of other ones or they can also be defined as uppermost concepts. For example a new concept *Concert* can be classified to be a sub-concept of *Event* in our ontology of figure 5 on page 11. For new instances the type has to be specified. For example a new instance *Spain* can be classified to be of the type *Country* in our ontology. Once a new element is added to the ontology, it can immediately be used for annotation.

2. *Consolidation in communities:* In this phase a consolidation of the elements added by users is performed. This is done in two ways, that are reusing and adapting the new elements.

   In the first case the reuse of elements in a consensual way by different users can show that several annotators associate the same meaning to an element. So for example this could be the case for a concept *Concert* being used by several users to annotate resources about music performances where players and singers are involved. Such an element can be considered to be mature.

   In the second case changes of the elements can contribute to the maturing process of them. Adaptations can be performed only for elements added by users and include the following cases:

- Changes of the label of an element can be performed. So for example misspellings can be corrected.
- Changes in the classification can be done. This can contribute to have elements situated at the right level within the existing taxonomy. As an example, we consider a concept *PoliticalMeeting* added by a user as sub-concept of *Event* in our ontology depicted in figure 5 on page 11. Another user being of the opinion that the definition of the concept as sub-concept of *Meeting* is more appropriate can perform this change.
- Two or more elements can be merged. This can be necessary if two different elements with the same meaning exist. If for example the concepts *Concert* and *MusicPerformance* are added and both are used in order to state that a resource is about a music performance where players and singers are involved, than it would be appropriate to merge them.
- An element can be split. This can be necessary if an element is used in annotations stating different things. So for example if a concept *Brand* is used in annotations of resources about a product brand and in annotations of resources about a brand of animals it would be appropriate to split the concept for example into the concepts *ProductBrand* and *AnimalBrand*.

With this ontology engineering approach users have the possibility to extend ontologies immediately while working with them. Over time the new extensions get solid and form a well-defined shared meaning.

3.2.2 Classification Support

In the previous section we presented an approach for extending ontologies while working with them. One task the user has to perform when adding new elements is to classify them. In order to provide aid in this task we present an approach to support users in the selection of adequate extension

points for newly added ontology elements. The approach is divided into two parts.

The first part uses a mechanism for an *automatic* generation of extension points that are presented to the user. These proposals can be used to classify the newly added ontology elements. In order to automatically calculate such proposals a profile-based recommendation system, as introduced in appendix B, is used. Based on user preferences or his behavior extension points can be computed. Considering the example of a user often using the concepts *Meeting* and *Conference* of the ontology depicted in figure 5 on page 11, this approach could propose the concept *Event* as extension point for a new concept. If the user selects this extension point the new item would be classified as a sub-concept of *Event*. This approach of automatically generated proposals for extension points works only if user preferences or information about his behavior are available. Moreover, it can be the case that the automatically generated proposals are not adequate for the classification of a certain element. For situations where no proposals can be generated or the generated ones are not adequate we propose an approach of manual classification that is presented in the following.

This second part of *manual* selection of extension points is based on the classification approach used in Ontogame[13]. Ontogame is an online computer game realizing a collaborative ontology engineering approach where the players are participating. Aim of the game is to identify randomly selected Wikipedia[14] articles as ontology concepts or instances and to classify them. The classification of new concepts with the Ontogame approach works as follows:

1. The user decides if an article is about a concept or an instance.

2. The user is asked to classify the article. He gets a list of the upper most concepts of the used ontology. Due to the fact, that Ontogame uses the PROTON ontology, the list of concepts presented to the player

[13]http://www.ontogame.org/
[14]http://www.wikipedia.org/

are *Abstract*, *Happening* and *Object*. These are the three uppermost concepts of the PROTON ontology.

3. Once the user selects one of these concepts, e.g. *Object*, the list of its sub-concepts is presented to the user. This step is repeated until the user does not reaches the undermost level of the ontology structure or he does a classification error[15]

This approach does not work only with the PROTON ontology, however, the structure of PROTON is very adequate. In our approach we base on the Ontogame mechanism and additionally use view projections in order to guide the user in the classification:

1. The user adds a new element by defining a label to be used.

2. The user is asked to classify the new element to be part of one of the available views. If none of the views is adequate the user can select to add the element to a special view of unclassified elements. Applied to our view projections presented in figure 8 on page 15 the user would be asked to classify the new element as an *Event Type*, and *Event Location* or as a category of the view of unclassified elements (e.g. a view named *Ungrouped*).

3. Once a view is selected the user is confronted with the task to find the adequate extension point within the view. For doing this we differentiate between three cases:

   (a) The selected view consists of concepts:

   - The user gets a list of the root categories of the view. He can define the new concept to be a root category or he can select one of the presented categories. In the first case the classification is done. In the second case the user gets a list of the sub-concepts of the selected one and continues with the next step.

[15]See [11] for a more information about how classification errors are detected.

- The user can define the new element to be a sub-category of the selected one or he can select one of the listed categories. In the first case the classification is done. In the second case the user gets a list of the sub-categories of the selected one and this step is repeated.

(b) The selected view consists of fixed instances:

- The user gets the list of concepts used for the classification of the fixed instances of the view. In our *Event Location* example view this would be the concepts *Country*, *State*, and *City*. The user is asked to classify the new element to be an instance of the presented concepts. Depending on the selected concept he is also asked to put the new element in relation with one of the existing. For example if the user decides to classify the new element to be a state in our *Event Location* view, he is asked to which country the new state belongs. Once this steps are done, the classification is terminated.

(c) The selected view is the special view for unclassified elements:

- The user can define the new element as root element of this view or as a sub-category of one of the existing ones. The mechanism for classifying the new element as a sub-category of an existing one is the same as described in case (a). New elements are defined to be concepts within the ontology.

Once the new concept or instance is classified it can be used immediately for annotation.

4 Conclusions

In this paper we presented a methodology for supporting users in the annotation of resources based on ontologies. The main objectives of our approach are to provide aid in the selection of ontology elements and to provide the

possibility to add new elements if necessary. In order to face these problems we designed a methodology composed of two parts, namely the *Element Selection* and the *Element Addition* parts.

The element selection part of the SA methodology is concerned with supporting the element selection process. For doing this we presented four components, but not all of them have to be implemented in order to fulfill the objectives of this part of the methodology. Three components provide element selection capabilities. These are the *Ontology Browser*, the *Semantic Search with Autocompletition*, and the *Recommendation System* components. On the one hand we have the ontology browser that allows to directly select elements necessary for annotation. This component does not provide support in its selection and, therefore, it is not enough to fulfill the objectives of the selection part of the methodology. On the other hand we have the semantic search and the recommendation system. These components provide support facilities and the implementation of only one of them would be enough to fulfill the goals of the selection part. Finally, we have the *Ontology Filter* that does not provide element selection capabilities. It enables to restrict the set of ontology elements that are used by the other components. The implementation of the filter is not necessary, but in combination with the other components it could enable a better support.

The element addition part of the SA methodology is concerned with providing the possibility to add new ontology elements during annotation time. In order to achieve this goal we presented two components that are the *Ontology Extension* and the *Classification Support* component. The former provides the capabilities necessary in order to achieve an extension of an ontology and is therefore necessary in order to fulfill the objectives of the addition part of the methodology. The latter component provides support to the user in the classification process of new elements. It has not to be implemented but from our point of view the combination of the two components provides a good combination.

In order to test some of the approaches of the SA methodology a proto-

typical implementation of some parts of it will be done as a next step. The selection of which components will be implemented has still to be performed.

A Semantic Search

Retrieval systems such as the one used by Google[16] or Yahoo[17] have reached a high level of maturity providing search results in an efficient way. Search techniques applied by those systems reside mainly on mechanisms for matching keywords in documents and on statistical approaches for the selection of adequate search results. Due to the fact, that such retrieval approaches do not consider the meaning of the content of the documents, the results not always contain the information desired by the user. In order to improve the quality of the search results, especially in huge knowledge bases such as the World Wide Web, new approaches have to be found. Semantic Web technologies provide the possibility to improve the quality of search results by assigning a meaning to the resources that is understandable for machines. This meaning can be taken into consideration in the retrieval process.

Due to the fact, that in order to benefit from semantic techniques in retrieval processes complex queries need to be formulated, systems has to be developed to allow users to simply formulate such queries. Actually, there exist several approaches for the solution of this problem such as natural language processing, user interfaces centered methods as well as techniques for the formulation of queries out of simple keywords. In the following we present a multi-facet, or view-based, semantic search method that represents a user interface centered approach.

In [9] an approach for multi-facet, or view-based, semantic search is discussed. The idea behind a view-based search is to behold the data from different perspectives. In order to achieve this goal, several views are created that present the underlying information from different aspects.

Traditional view-based searches can be realized by associating the search objects with keywords. The keywords are taken from orthogonal hierarchies, whereas each hierarchy represents a facet on the underlying data. The semantic view-based search is an extension of a traditional view-based search

[16] http://www.google.com/

[17] http://www.yahoo.com/

by the use of ontologies instead of keyword hierarchies. A challenge in this approach is the projection of ontologies onto views.

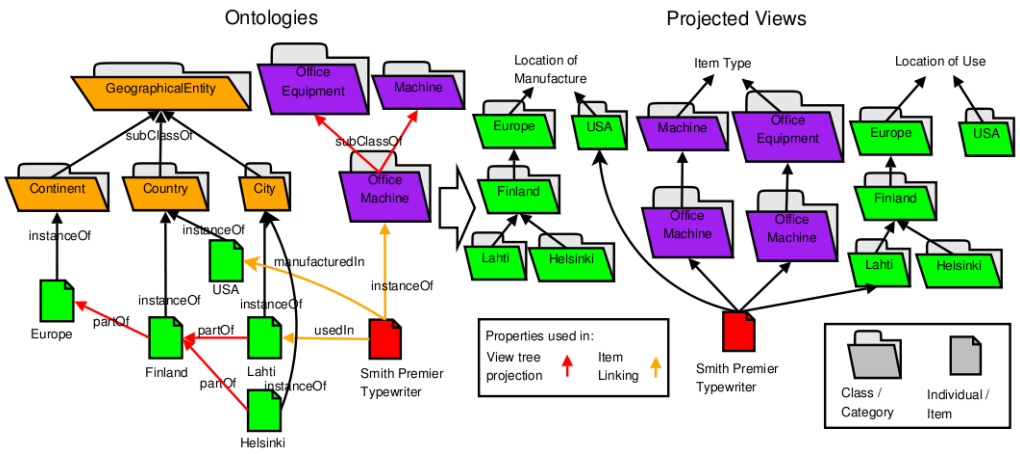

Figure 13: View Projection [7]

Figure 13 depicts an example of view projection. On the left side we have an ontology providing concepts and instances for the description of items that are machines or office equipments. On the right side we have three hierarchical views constructed out of the ontology that are the *Location of Manufacture*, the *Item Type* and the *Location of Use* view. The elements of the views are called *categories* that in this example are either ontology concepts or fixed instances. The first view can be used for searching for items manufactured in a certain location, the second for items of a certain type and the last for items with a specific location of use. These facets are constructed considering the *subClassOf* and the *partOf* relations.

Based on these three views queries can be performed. As an example, we consider a user searching for office equipments used in Finland. This search request can be formulated by selecting *Office Equipment* from the *Item Type* view and *Finland* from the *Location of Use* view. The result of the search would be the *Smith Premier Typewriter*.

According to [9] a view-based search can help the user in the retrieval

process in several ways:

- The user gets an overview of the information that can be found.
- The different hierarchies act as guide in the definition of search queries or offer aid in navigating through the resources.
- Facets can disambiguate search queries refining them.

B Recommendation System

The retrieval of the desired information can be a difficult task even if supported by a search functionality. Sometimes, the user has not enough information in order to formulate queries for the retrieval of the desired information. In such cases a recommendation systems can provide aid to the user.

A recommendation system is concerned with the search and presentation of items that could be of interest to the user without the need to directly search for them. As an example, we consider a user searching for artists playing the same kind of music as his preferred artist but not knowing names of them. In an information retrieval application using a recommender system a list of bands related to the specified one, e.g. by playing the same kind of music, can be presented to the user.

In [9] an approach for the creation of a recommender system based on ontologies is presented. Strategies for the creation of recommendations, discussed in this paper are *profile-based recommendations*, *similarity-based recommendations* and *rule-based recommendations*.

Profile-based recommendations reside on different techniques for considering the interests of a user. On the one hand, preferences and characteristics can be explicitly defined by the user, and, on the other hand, information about the users behavior are collected. Both approaches can contribute to provide personalized recommendations.

Similarity-based recommendations can be calculated by the computation of the semantical distance of resources based on its metadata. The semantical distance is a size for example stating how much a resource is related to another resource. The difficulty of this technique is to find an appropriate mechanism to calculate it. In appendix C such an approach is presented.

Finally, in [9] rule-based recommendations based on the definition of generic logic rules are presented. This kind of approach allows to represent any kind of recommendation. An advantage of this method is that it allows to add additional information to the recommendation rules. So for example a description can be added that can be presented to the user in order to explain in which way recommended items are related to a specific resource.

C Semantic Relevance

Information retrieval and recommendation systems try to guide the user to the desired information within a knowledge base. The facilities provided by these systems can be augmented or realized by weighting the information, for example by calculating the relevance of the items in the knowledge base.

Applied to a search the relevance can be used to realize a better presentation of search results to the user. Result sets can be sorted arranging the most important objects at the beginning. Moreover, if the presentation of the result set is limited to a certain number of elements, the relevance of the items can be used to eliminate the less important ones from the matching set. In recommender systems the relevance calculation provides a mechanism to find items that could be important for the user such as items similar to a given one.

In [10] an approach for the calculation of the relevance of resources annotated by means of an RDF(S) language is presented. It bases on the *tf-idf* method [1] that allows the calculation of the relevance of documents in relation to a keyword and is adapted to the use with ontologies.

In order to support RDF(S) based metadata the method has to consider the following criteria. Firstly, in knowledge bases *concepts* and *instances* are typically hold separate. Instances can be shared among annotations, due to the fact, that each resource gets a unique identifier. This can result in over-domination of commonly referred instances. Secondly, concepts in ontologies normally are organized in *subsumption hierarchies*[18]. A resource being an instance of a concept x is also an instance of the super-concepts of x. The relevance calculation method has to consider the most specific concepts to be the most relevant ones. Thirdly, ontologies can include *part-of* relations that express that an item is part of another item. These relations has to be handled separately from subsumptions.

Considering these criteria the method described in [10] provides a mechanism for the calculation of the relevance of an *instance* with respect to another instance, a *triple* with respect to another triple and an *annotation* with respect to another annotation.

D Semantic Autocompletition

Autocompletition is the completition of an input string, consisting of one or more words or partial words, into a word or sentence of an underlying vocabulary. A search of the inserted string is performed against a given vocabulary trying to find similar words. Autocompletition is used in lots of applications, starting from development environments for programmer, where typed strings are autocompleted into commands, to Web browsers providing the autocompletion of inserted URLs in the location bar[19] and Web interfaces with autocompletition in form fields, providing a list of already inserted values.

Approaches used for the autocompletition in such applications, are autocompletition *by-request* and autocompletition *on-the-fly*. In the first case

[18]Taxonomies

[19]Place where the URL of a Web page is inserted.

the user types a string and then starts the autocompletition, for example by activating the corresponding button or key combination. This approach is often used in programming environments. In the second case autocompletition triggered automatically while the user is typing the input string. Therefore, the result list of the corresponding concepts is continuously updated and refined. This is the case in several Web browsers offering the mentioned autocompletition in the location bar.

Semantic autocompletition is the extension of the traditional autocompletiton to a semantical level. Input strings are autocompleted into terms of ontologies instead of keywords of vocabularies. Moreover, not only similarity checks are performed for the retrieval of these concepts but also more sophisticated ontology-based techniques are applied. In [8] an approach for semantic autocompletition based on the view-based semantic search, introduced in appendix A, is presented. They propose to autocomplete input strings into categories of projection views by the following mechanisms:

- *Autocompletition based on equivalence relations:* Equivalence relations represent one of the first steps in the process of semantic information retrieval. According to [8], there exists several kinds of equivalence relations that are *lexical variants*, *synonymy*, *polysemy*, and *homonymy*.

  Lexical variants and synonymy are terms that represent the same ontological concept. In autocompletition a mechanism can be provided, that autocompletes the lexical variants and synonymy into a preferred term. In this way a more consistent use of concepts can be achieved. Because morphological variants can have a very different spelling, a morphological analysis[20] may be necessary for a well working autocompletition, especially in the case of morphologically rich languages.

  Polysemy denotes single terms that have different but related meanings (e.g., 'arrow head' and 'human head'), whereas in homonymy the

[20] Analysis method developed by Fritz Zwicky, that looks at all possible solutions to a problem.

meanings are totally different (e.g., 'river bank' and 'blood bank'). In this cases autocompletition can not disambiguate the meaning. It has to provide a list to the user from where he can choose the more appropriate category.

One problem in determining the equivalence between input text and categories is how to deal with phrases. A possibility is to search within the categories by building all permutations and to return only the best matching result items. Also in such phrasal searches a morphological analysis is necessary as a pre-step.

- *Indirect semantic autocompletition:* Another possibility of autocompletition is indirect semantic autocompletition. This approach uses the relations between ontology elements for proposing categories to the user. Once categories matching an input string are found, e.g. based on equivalence relations, other categories related to the found ones can be retrieved. An example for indirect semantic autocompletition is to return a result set containing all the cities of a country, when the input was the name of the country. This example of autocompletition makes use of *part-of* relation. However, any desired relation can be used, as long as it can make sense to the end user.

- *Semantic role completition:* A further possibility of autocompletition is provided by the consideration of *semantic roles*. This can be done by taking into account the roles in which the categories are used. For example a city is related to a product either as place of manufacture or as place of sale. In such cases a choice can be provided to the user in order to disambiguate the result. Once the user has taken a decision, different results are presented, depending on his choice (unless the manufacture and sale place are the same).

E Work-Embedded Ontology Engineering

Ontology Engineering is the area concerned with the creation, maintenance and evaluation of ontologies. Several approaches for a structured solution of these problems exist. An overview of methodologies for building ontologies is provided in [5].

Ontology engineering processes are divided into different activities and most of them include management, development and support activities [3]. Figure 14 provides a graphical overview of this scenario.

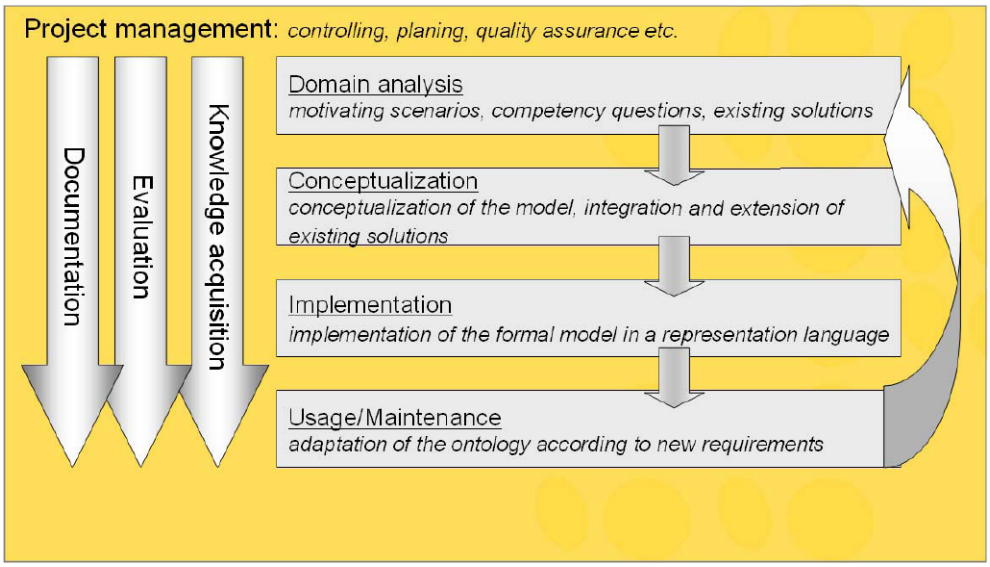

Figure 14: Ontology Engineering Activities [3]

The activities concerned with the development of the ontology are depicted in the middle of figure 14 and are introduced in the following:

- *Domain analysis:* Formulation of motivating scenarios and competency questions, collection of important terms and analysis of existing solutions.

- *Conceptualization:* Definition of concepts, properties, relations between

the concepts and axioms defining constraints in the model. The result of this activity is a conceptualization of the domain.

- *Implementation:* Implementation of the conceptualization in a specific representation language.

- *Usage/Maintenance:* Application of the ontology and adaption to new requirements.

In current approaches based on such a structure, ontologies are supposed to be created by ontology and domain experts. They don't consider the possibility to apply changes to ontologies by the persons using them and while they are using them [4]. Due to the fact, that in daily work the needs of changes in the used ontologies can emerge, an easy to use approach for the extension through the user would be appropriate. Considering an annotation scenario where an ontology does not provide a concept needed for the annotation of a given resource, it would be convenient for the user to have the possibility to add this concept immediately instead of waiting for domain and modeling experts extending the ontology to meet the new needs. In order to face this problem, [4] presents a work-embedded ontology engineering approach focused on collaborative maintenance named *Ontology Maturing*.

Within communities the collaboration of users often results in a shared meaning about specific topics. The main idea is to find an approach to formalize this shared meaning in order to result in formal ontologies. For the solution of this problem [4] proposes an ontology maturing model consisting of four phases:

- *Emergence of ideas:* In this phase new concept ideas are introduced. For example missing concepts are created or existing ones are adapted.

- *Consolidation in communities:* Reuse and adaption of concepts results into a shared vocabulary commonly accepted by a community.

- *Formalization:* In this phase concepts are structured by the definition

of relations among them. Taxonomical and other user-defined relations can be used.

- *Axiomatization:* Additional background knowledge is added in order to improve the inference process what requires knowledge about logical formalisms. This phase is thought to be executed by domain and ontology experts.

Based on this model two approaches for collaborative ontology maintenance are presented in [4]: The *IMAGINATION EU project*[21] and the *IM WISSENSNETZ - IN THE KNOWLEDGE WEG research project*[22]. The former project concentrates on the consolidation of descriptions in communities based on concepts and relations and the latter project concentrates on the formalization based on consolidated concepts.

[21] http://www.imagination-project.org
[22] http://im-wissensnetz.de

References

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[2] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284:34–43, 2001.

[3] Elena Paslaru Bontas, Christoph Tempich, and York Sure. Ontocom: A cost estimation model for ontology engineering. In I. Cruz et al., editors, *Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, volume 4273 of *Lecture Notes in Computer Science (LNCS)*, pages 625–639. Springer-Verlag Berlin Heidelberg, 2006.

[4] Simone Braun, Andreas Schmidt, and Andreas Walter. Ontology maturing: a collaborative web 2.0 approach to ontology engineering. In *Proceedings of the Workshop on Social and Collaborative Construction of Structured Knowledge at the 16th International World Wide Web Conference (WWW 07), Banff, Canada*, 2007. URL `http://www.andreas-p-schmidt.de/publications/ontology_maturing_braun%_schmidt_walter_www07.pdf`.

[5] Asuncion Gomez-Perez, Oscar Corcho-Garcia, and Mariano Fernandez-Lopez. *Ontological Engineering*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003. ISBN 1852335513.

[6] Siegfried Handschuh and Steffen Staab. *Annotation for the Semantic Web*. IOS Press, 2003.

[7] Eero Hyvönen. Semantic web applications in the public sector in finland - building the basis for a national semantic web infrastructure, 2006. Paper presented at the Norwegian Semantic Days, April 26-27, Stavanger, Norway.

[8] Eero Hyvönen and Eetu Mäkelä. Semantic autocompletion. In *Proceedings of the first Asia Semantic Web Conference*

(ASWC 2006), Beijing. Springer-Verlag, New York, August 4-9 2006. URL http://www.seco.hut.fi/publications/2006/hyvonen-makela-semantic-autocompletion-2006.pdf.

[9] Eero Hyvönen, Samppa Saarela, and Kim Viljanen. Application of ontology techniques to view-based semantic search and browsing. In *The Semantic Web: Research and Applications. Proceedings of the First European Semantic Web Symposium (ESWS 2004)*, 2004. URL http://www.seco.hut.fi/publications/2004/hyvonen-saarela-et-al-application-of-ontology-techniques-2004.pdf.

[10] Tuukka Ruotsalo and Eero Hyvnen. A method for determining ontology-based semantic relevance. In *Proceedings of the International Conference on Database and Expert Systems Applications DEXA 2007, Regensburg, Germany*. Springer, September 3-7 2007.

[11] Katharina Siorpaes and Martin Hepp. Ontogame: Towards overcoming the incentive bottleneck in ontology building. In *Proceedings of the 3rd International IFIP Workshop On Semantic Web & Web Semantics (SWWS '07) co-located with OTM Federated Conferences*. Springer LNCS, 11 2007.