

# Improving semantic interoperability through analysis of model extension

*Xiaomeng Su, Sari Hakkarainen and Terje Brasethvik*  
*Dept. of Computer and Information Science*  
*Norwegian University of Science and Technology (NTNU)*  
*N-7491 Trondheim, Norway*  
*{xiaomeng, sari, brase}@idi.ntnu.no*

## *Abstract*

*The overall problem addressed in this paper is to improve semantic interoperability in heterogeneous systems. Normally, the semantic of data are carried by ontology (concept model). Thus reconciling data semantics therefore boils down to reconciling relevant ontologies. A candidate solution is proposed using extensional information, i.e. instance information of the ontology to enrich the ontology and further to calculate similarities between concepts in two ontologies. Text categorization is used to automatic assign instance to the concepts in the ontology. Information retrieval techniques are used to calculate similarity between concepts. Based on the similarities measure, a heuristic method is used to establish mapping assertions for the two ontologies.*

## **1 Introduction**

Lately, there has been much research related to the new generation Web – Semantic Web. The hope is that the semantic web can alleviate some of the problems with the current web, where the information is not given well-defined meaning. The goal is to enable computers process the interchanged data in a more intelligent way. In an open system like the Internet, which is a network of heterogeneous and distributed information systems (IS), mechanisms have to be developed in order to enable systems to share information and cooperate. The essential requirement for the semantic web is interoperability of IS.

Ontology is a key factor for enabling interoperability in the semantic web [Bernees-Lee01]. One of the most cited definitions of ontology is that “an ontology is an explicit specification of a conceptualisation” [Uschold96]. It includes an explicit description of the assumptions regarding both the domain structure and the terms used to describe the domain. Ontologies are central to the semantic web because they allow applications to agree on the terms that they use when communicating. Ontology facilitates communication by providing precise notions that can be used to compose message (queries, statements) about the domain. For the receiving party, ontology helps to understand messages by providing the correct interpretation context. Thus, ontologies, if shared among stakeholders, will improve system interoperability across IS in different organizations and domains.

However, there has long been argued that there is and will be no one single universal shared ontology, which will be applauded by all players. Besides, when ontologies are developed independent of each other in a large, distributed environment such as the web, it is inevitable that the same piece of domain knowledge will be captured in different ontologies. The problem of improving system interoperability will therefore, rely on the reconciliation of different ontologies used in different systems. The reconciliation is often approached by manual or semi-automated integration of ontologies.

In order to achieve integration of ontologies, it is necessary to integrate both syntax and semantics of the involving ontologies. There is a wide agreement on syntactical issues in the software community, and syntax problem may be solved if there is a willingness among the actors to do so. For instance, [Gruber93] describes a mechanism for defining ontologies that are portable over representation systems. Definitions written in a standard format for predicate calculus are translated by a system called Ontolingua into specialized representations, including frame based languages as well as relational languages. The deep and unsolved problems are thus with the semantic integration issue. As is stated in [Heflin99], the integration of ontologies remains an expensive, time consuming and manual activity, even though ontology interchange formats exist.

One of the fundamental elements of ontology integration process is establish mapping between ontologies. Mapping processes typically involve analysing the ontologies and comparing them to determine the correspondence among concepts and detect possible conflicts. A set of mapping assertions is the main output of a mapping process. The mapping assertions can be used directly in a translator component, which translates statements that are formulated by different ontologies. Or a follow-up integration process can use the mappings to detect merging points.

It is implausible that one can develop a fully automatic implementation of mapping. The main reason is that most models are incomplete. Despite this inherent incompleteness, there is still quite a lot of information that a mapping algorithm can use to produce a draft mapping, for review by a human designer [Bernstein01]. Thus, much of the research in this area is focusing on semi-automating the mapping process. The approach we are going to introduce is one of such research.

## **2 An architecture for ontology mapping through analysing extensional resources.**

In this section, we propose that the problem of integrating ontologies can be tackled with mapping methods based on extension analysis. We first describe the overall approach. Second, a meta model for organizing and describing the mapping results is proposed. The third sub-section deals with how to discover mappings by employing IR and text categorization techniques. Finally, some scenarios, where this approach can be applied, are suggested.

### **2.1 Overall approach.**

The word ontology has been used to describe artefacts with different degrees of structure. These range from simple taxonomies (such as the Yahoo hierarchy), to metadata schemes (such as the Dublin Core[DC]), to logical theories. In our context, the scope and assumption of our work are the following:

- 1) To begin with, we are focusing on the so-called lightweight ontology. Hence, an *ontology*<sup>1</sup> is defined as a set of elements connected by some structure. Among the structures, we single out hierarchical IS-A-relation and all the others we call them “related” relations, which is merely an indication of relatedness. A *classification*

---

<sup>1</sup> We use the term ontology and concept model interchangeably in the rest of the paper unless otherwise explicitly specified.

*hierarchy* is a typical example of ontology organized only by hierarchical IS-A-relation.

- 2) The ontologies that are subject to be mapped, are expressing overlapping knowledge in a common domain.
- 3) Ontologies can be expressed in different representational languages [Su02]. Here, we assume that it is possible to translate between different formats. In practice, a particular representation must be chosen for the input ontologies.
- 4) Our approach is based on Referent Modelling Language (RML) [Solvberg98], which is an ER-like language with strong abstraction mechanism and sound formal basis. The language has a XML representation.

The overall process of ontology mapping can then be defined as:

Given two ontologies A and B, mapping one ontology with another means that for each concept (node) in ontology A, try to find a corresponding concept (node), which has same or similar semantics, in ontology B and vice versa. To be more exact, we need to

- a) Define the different aspects of the mapping statements, i.e. to answer the questions: what is a mapping assertion and what kind of information will it reveal?
- b) Develop algorithm, which can discover concepts that have similar semantic meaning, i.e., to answer the question: how to get the mapping rules?

Thus, the result of a mapping process is a set of mapping rules. Those mapping rules connect concepts in ontology A to concepts in ontology B.

Approaches from different communities have been proposed in the literature to deal with this problem[Rahm01][McGuinness00][Stumme01][Wiederhold99]. The intention of this work is to draw experiences from the related areas and base on those experiences, to formulate our own solution.

The first step in handling semantic heterogeneity should be the attempt to enrich the semantic information of concepts in ontologies, as it is well understood that the richer information the ontologies possesses, the higher probability that high quality mappings will be derived. We do that by taking into account the extension, i.e. instance information a concept possesses. The instance here we use are documents that have been classified to the concepts. The idea behind is that written documents that are used in a domain inherently carry the conceptualizations that are shared by the members of the community.

On the other hand, we also consider information retrieval (IR) technique as one of the promising components of our approach. With information retrieval, a concept node in the first ontology is considered as a query to be matched against the collection of concept nodes in the second ontology. Ontology mapping thus becomes a question of finding concept nodes from the second ontology that best relate to the query node. One of the major advantages of employing IR is domain independence.

Converging the above two ideas, it is evident that the instance information of a concept needs to be represented in a way that is compatible with IR framework. Given that vector space model is the most used one in IR, it is natural to think of representing the instance information in vectors, where the documents under one concept become building material for the vector of that concept.

In some cases, ontologies exist without any available instance information. We tackle that by assigning instance to the ontologies. That is where text categorization come into play, aiming at automate the process of assigning documents to concept nodes.

## 2.2 Meta model of mapping

A general implementation of the mapping process compares each element in ontology A with each element in ontology B and determines a similarity metric per pair. Only the combinations with a similarity value greater than the threshold value  $\tau$  (or top- ranked lists) are considered as match candidates. In general, the different mapping methods can be classified with respect to *what* information is used to compute the similarity value and *how* this value is computed.

In order to discuss definitions of similarity and to support development of the novel mapping approaches, we need to define a meta model for mapping. In [Hakkarainen99], a notion of correspondence assertion is introduced for that purpose. We adopt that correspondence assertion meta model as a base for discussing different types of mappings in our method.

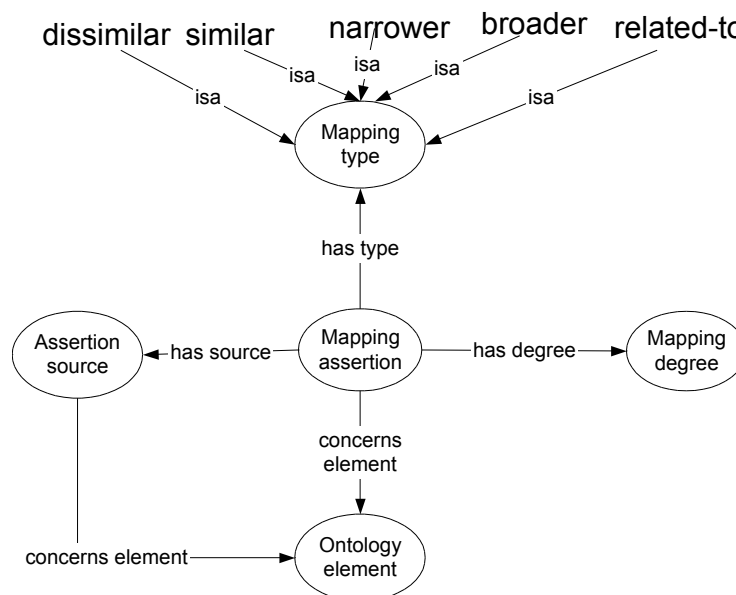


Figure 1 mapping assertion metamodel

The meta model in Figure 1 has the following meaning: a mapping assertion describes the relationship between two ontology elements and supports further description of the involved resources. A mapping assertion is uniquely assigned to two ontology elements. It has also a mapping degree in order to provide a way of ranking the outputs. A mapping type is also attached to a mapping assertion, which specifies how the pair of ontology elements is related. There are five types of mapping assertions defined – similar, narrower, broader, related-to and dissimilar. The first four types are commonly used in thesaurus [Wilson02]. Technically, the term dissimilar is used to specify the situation when two concepts have the same (similar) names, but denote to different things. The intention of the assertion source is to provide an explanation why the particular assertion is chosen (derived by linguistic information of names, for instance). For a mapping process, mapping assertion is the core output of the process. How to generate the mapping assertions is discussed below.

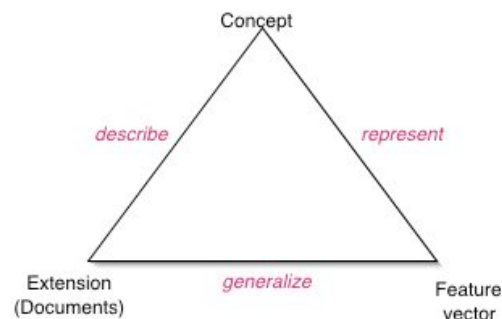
## 2.3 Mapping Discovery Method

In the context of database, a schema defines the *intension* of the database and the instances of data define *extensions*. The same can apply to ontology, where an ontology is a intentional

description of a Universe of Discourse (UoD), and the set of instances, which conform to that ontology is the extension of the UoD.

A straight forward account of our approach is enriching the semantic meaning of each node by looking at the instances (in our case, documents are counted as instances) it possesses; then representing the node as a generalization of the information its instances provide; and calculating similarity based on the generalizations. The underlining believe of this approach is that the semantic meaning of a concept node is best described by the instances that have already been classified under that node [Labrou99]<sup>2</sup>.

Speaking more specifically, each concept node is enriched with a feature vector and the documents, which belong to that concept node, are “building materials” for the feature vector. In other words, the feature vector becomes a generalization of the extension of the concept. As is illustrated in figure 2, a concept is described by its extension, i.e. the instances (documents), which have been assigned to this concept. These documents will then be used to generate a representative feature vector. This feature vector is then used as a computational representation of the concept.



**Figure 2 Representing a concept by generalization of its extension**

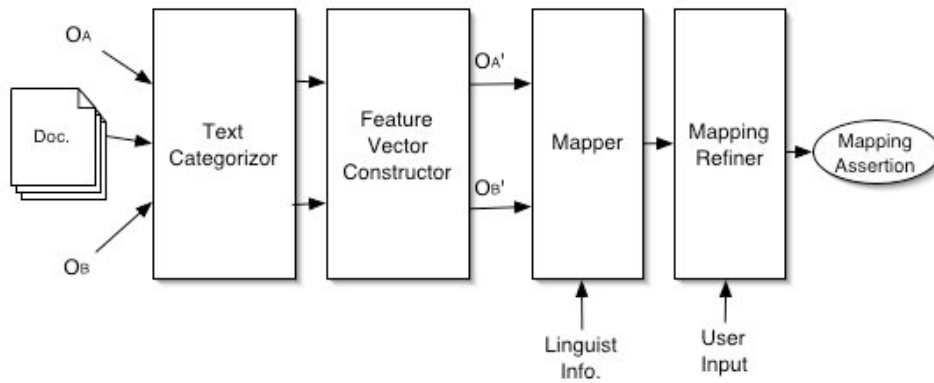
The intuition is that given two ontologies A and B, we first construct a representative feature vector for each node in the two ontologies. With the feature vectors at hand, we calculate a similarity measure  $\text{sim}(a_i, b_j)$  pair wise for the two ontologies. Then the node with the highest similarity will be ranked on top. Information retrieval technique is used when  $\text{sim}(a_i, b_j)$  is calculated.

Figure 3 depicts the general architecture of the suggested mapping process. The approach takes the two ontologies and document sets as input. There can be one or two document sets. In the former case, we assume the documents are relevant to both ontologies, while in the latter, it is assumed that the two document sets share the same vocabulary. There are mainly four steps in the process, namely:

1. Text Categoriser assigns documents to the concepts nodes of the two ontologies respectively.
2. FVC (Feature Vector Constructor) builds up feature vectors for each concept nodes in the two ontologies.
3. Mapper calculates similarity value for each pair of the concept nodes
4. Mapping Refiner formulates mapping assertions, presents them to the user and manages the user feedback.

---

<sup>2</sup> Though in certain approaches, e.g. Description Logics, an intentional definition is believed to best describe the semantic meaning.

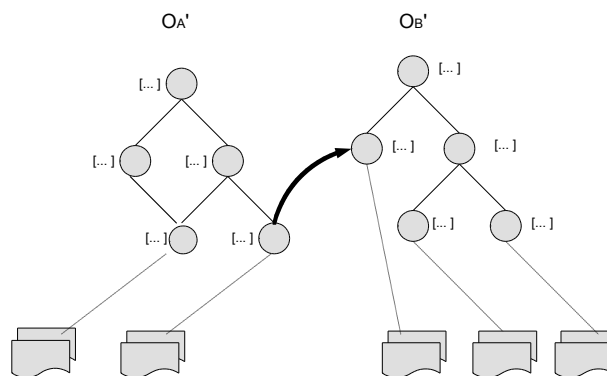


**Figure 3 Architecture of Ontology Mapping**

The first step is to assign documents to concept nodes of the ontology. Text categorization technique is the first natural candidate for that task. Alternatively, for a basic method of assigning documents to concepts, we may consider each concept as a query that is fired against a general-purpose search engine, which maintains the documents in question. Each document in the result set that has a ranking value greater than a certain minimum threshold is then assigned to the query concept. Which method to choose is partially determined by the kind of resources that are readily accessible.

The assigning of documents to concept nodes is necessary when no instance knowledge of the ontology is available. However, if documents have already been assigned to specific nodes, we can skip the first step and construct feature vector for the concept nodes directly<sup>3</sup>.

The second step concerns building up feature vectors for each concept nodes in the two ontologies. The intuition is that for each node a feature vector can be calculated based on the document assigned to it. Following a classic Rocchio algorithm[Aas99], the feature vector for node  $a_i$  is computed as the average vector over all document vectors that belong to node  $a_i$ . Following the same idea, the feature vector of a non-leaf node is computed as the centroid vector of its sub nodes. Thus, hierarchical information is partially taken into consideration. The output of this step is two intermediate ontologies,  $O_A'$  and  $O_B'$ , where each concept node has been associated with a feature vector as depicted in figure 4.



**Figure 4 Ontology Mapping based on feature vectors for concepts.**

The third steps take the two intermediate ontologies as input and produces initial mapping assertions as the main output. The algorithm used in the Mapper is based on information retrieval techniques.

<sup>3</sup> An example, where documents have already been assigned to concept node is Open Directory Project [ODP].

With the feature vector at hand, the similarity of the two nodes is measured by calculating the cosine measure of the two associated vectors. Further, as indicated in [Rahm01], using just one approach is unlikely to achieve as many good mapping candidates as one that combines several approaches. In our approach, supplementary information is used as well in order to acquire better mapping candidates. Here, we use linguistic information of names. A matching algorithm of class names is deployed to give a boost for nodes, which have the same or similar names (prefix, suffix, or word root) with the compared one. WorldNet is used to provide synonym information.

The final step involves presenting the mapping assertion to users and managing user feedback. Here, the user is in control of accepting, rejecting or changing the assertions. In case where the system was unable to find satisfactory match candidates, the users will be able to specify mappings. The user's actions are recorded in the application log, and the feedback information can be studied by other learning components in order to improve the mapping process in the future.

## **2.4 Scenarios**

The approach can be applied in different contexts. One context is documents retrieval and publication between different web portals. Users may conform to their local ontologies through which the web portals are organized. It is desirable to have support for automated exchange of documents between the portals and still let the users keep their perspectives.

Another area is product catalogue integration. In accordance with [Fensel01], different customers will make use of different classification schemas (UNSPSC, UCEC, and EClass, to name a few). We need to define links between different classification schemas that relate the various concepts. Establishing such a connection helps to classify new products in other classification schemas and this in turn will enable a full fledged B2B business, where a certain number of well know standard vocabularies coexist among business partners and mapping relates their mutual vocabularies [Hoferiter02].

Service matching is yet another good candidate to apply the method, though we have to assume that there are some service description hierarchies (the MIT process handbook for instance[PH]) and that the provider and the requester are using different classification schemas. By using the extension description of the service hierarchy, we can compute a feature vector for each service node. Then the matching can be conducted by calculating the distance between the representative feature vectors.

## **3 System functional view**

In this section, we discuss the four steps of our method in more detail. We illustrate the method with an example taken from a domain in information systems engineering, where two ontologies are considered for that domain. A text corpus, containing articles from information system, is used as an extensional description of the two ontologies. The text corpus, together with the ontologies, provide input to the system as depicted in figure 3.

### **3.1 Text categorization**

Text categorization techniques are used to automatically assign documents to one or more predefined categories based on their contents. We use a linguistic based classifier CnS (Classification and Search) [Brasethvik01] to categorise documents into the ontologies.

Figure 5a shows a user working on one single document. Suggestions are marked with a green triangle. Users may accept and reject a single suggestion by clicking either on the green or

white part of the concept respectively, or all suggestions at once by using the toolbar buttons. When examining classifications for one document, the user can examine the full text of the document by clicking on the document tab (figure 5b).

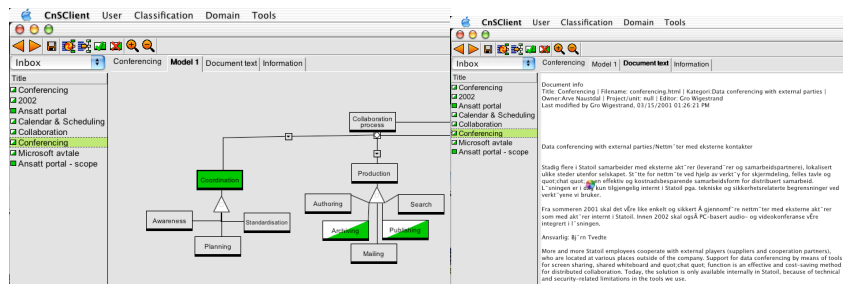


Figure 5 Classification of one document (a) and examining one document text related to the highlighted concept in a (b)

### 3.2 Feature vector construction

The above step provides as output two ontologies, where documents have been assigned to each concept nodes in the ontologies. The next step is to calculate a feature vector for each concept node in the two ontologies respectively. To calculate a feature vector for each concept node, we first need to establish feature vectors for each document that belongs to the concept node.

#### *Pre-processing.*

The first step is to transform documents, which typically are strings of characters, into a representation suitable for the task. The text transformation is of the following kind: remove HTML (or other) tags; remove stop words; perform word stemming (lemmatization).

A linguistic workbench developed at Norwegian University of Science and Technology (NTNU) is employed [Kaada02]. Documents are processed through a chain of components that each analyze or transform the document contents. The workbench consists of 8 components: POS(Part-Of-Speech)tagging, Lemmatization, Phrase detection, Language detection, Stop-word removal, word class filtering, weirdness test and XSL style sheet. Each component is running as an XML-RPC server and each has a specific port number for addressing. XML is used as data exchange format between various components. This architecture assures us to use the workbench in a flexible way, since we can run each task independently and control the order of the tasks.

#### *Document representation.*

We use the vector space model [Salton83] to construct the generalization of the documents. In vector space model, documents are represented by vectors of words. There are several ways to determining the weight of word  $i$  in a document  $d$ , we use the standard  $tf/idf$  weighting [Salton83], which assigns the weight to word  $i$  in document  $d$  in proportion to the number of occurrences of word in the document, and inverse proportion to the number of documents in the collection for which the word occurs.

The basic statistics about word frequency is provided by using the linguistic workbench. Thus, for each document  $d$  in a document collection  $D$ , a weighted vector is constructed as follows:

$$\vec{d} = (w_i) \text{ where } w_i \text{ is the weight of word } i \text{ in document } d.$$

$w_i = f_i * \log(\frac{N}{n_i})$  where  $f_i$  is the frequency of word  $i$  in document  $d$ ,  $N$  is the number of documents in the collection  $D$  and  $n_i$  is the number of documents that contains word  $i$ .

### Concept vector construction.

We differentiate here leaf node and non-leaf node in the ontology.

For each leaf node, the feature vector is calculated as an average vector on the documents vectors that have already been assigned to this node. Let  $C^k$  be the feature vector for concept node  $K$  and let  $D_j$  be the collection of documents that have been assigned to that node  $K$ . Then for each feature  $i$  of the concept node vector, it is calculated as:

$$C_i^k = \frac{\sum_{D_j \in k} w_{ij}}{\|D_j\|}$$

When it comes to non-leaf node, the feature vector  $C^k$  for a non-leaf concept node  $K$  is calculated by taking into consideration contributions from the documents that have been assigned to it, its direct sub nodes and the nodes to which node  $K$  hold a related relation. Let  $D_j$  be the collection of documents that have been assigned to that node  $K$ , let  $S_t$  be the collection of its direct sub nodes and let  $S_r$  be the collection of its related nodes. The  $i$ th element of  $C^k$  is defined as:

$$C_i^k = \alpha * \frac{\sum_{D_j \in k} w_{ij}}{\|D_j\|} + \beta * \frac{\sum_{S_t \in k} w_{it}}{\|S_t\|} + \gamma * \frac{\sum_{S_r \in k} w_{ir}}{\|S_r\|}$$

where  $\alpha + \beta + \gamma = 1$ .  $\alpha$ ,  $\beta$  and  $\gamma$  are used as tuning parameters to control the contributions from the concept's instances, sub concepts, and related concepts respectively.

### 3.3 Similarity calculation.

The similarity of two nodes in two ontologies is directly calculated as the cosine measure between them. Let two feature vectors for concept node  $a$  and  $b$  respectively, both of length  $n$ , be given. The cosine similarity between node  $a$  and node  $b$  is defined as:

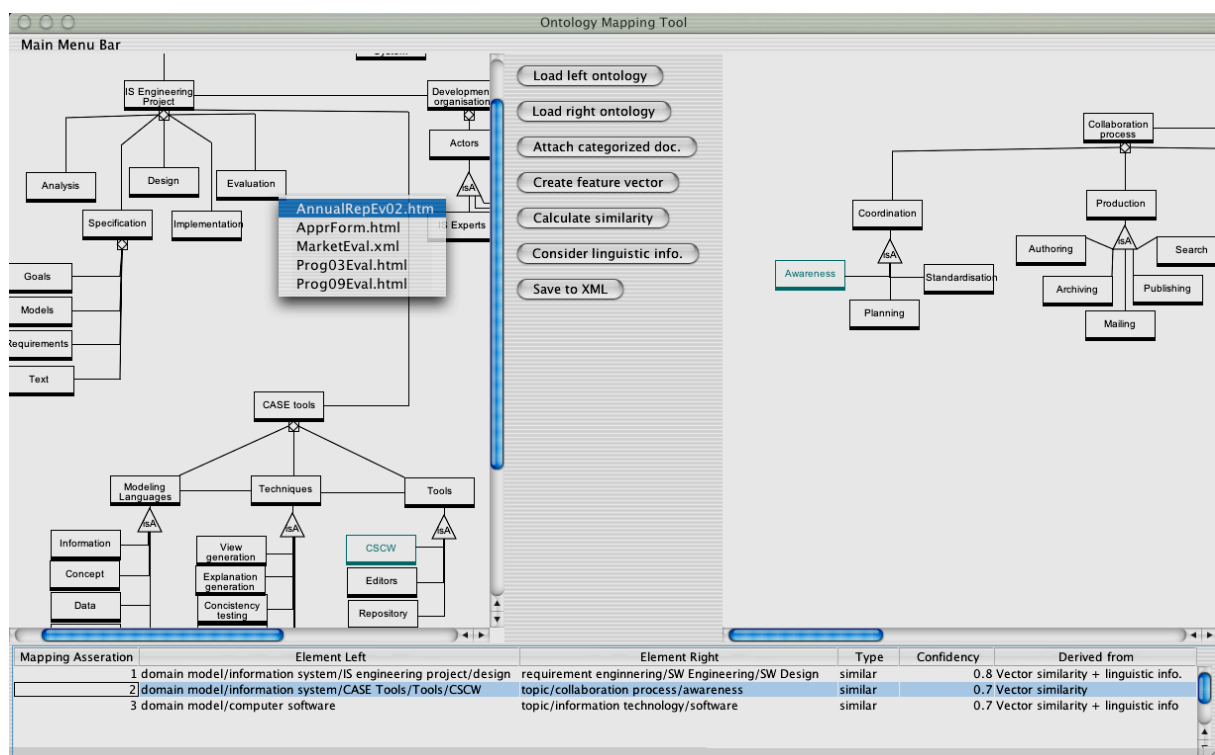
$$\text{sim}(a,b) = \text{sim}(C^a, C^b) = \frac{\sum_{i=1}^n (C_i^a * C_i^b)}{\sqrt{\sum_{i=1}^n C_i^a{}^2} * \sqrt{\sum_{i=1}^n C_i^b{}^2}} = \left( \frac{C^a}{\|C^a\|} \right) \cdot \left( \frac{C^b}{\|C^b\|} \right)$$

For concept  $a$  in ontology  $A$ , to find the most related concept  $b$  in ontology  $B$ , the top  $k$  ranked concept nodes in ontology  $B$  are selected according to the initial similarity measure calculated above. Those promising concepts are further evaluated as follows. If one concept has a linguistically similar name as concept  $a$ , its similarity measure will get a boost. How large the boost is can be tuned by a parameter.

### 3.4 Mapping assertion generation and management.

Recall that each mapping assertion describes the relationship between two ontology elements and supports further description of involved resources. Recall figure 2, the mapping assertion meta model. That meta model is used to describe the mapping results.

Figure 6 shows a preliminary prototype of the ontology mapping tool. The figure illustrates a mapping process and the obtained mapping assertions. Both ontologies are represented visually in RML (Referent Modelling Language) [Solvberg98][RML]. The CnS software performs the document categorization process and the result is being stored in a XML file. This information is then loaded into the mapping system. The user can view the documents assigned to a particular concept by right clicking that concept, as is illustrated in figure 8. Next, the mapping system constructs the feature vector for each document and the concept node consequently. In this process of constructing representative vector, the linguistic workbench is used to remove non-informative features and to provide basic statistical counting of word frequency. Next, the mapper calculates the similarity pair wise for the elements in the two ontologies based on the vectors. It is up to the user to decide whether the concept naming similarity shall be counted in or not. If the user decides to consider that similarity, he/she triggers the process and the naming similarity is calculated and added as a boost to the vector similarity measure. As a result, a list of top ranked mapping assertion will be generated and listed in the table at the lower part of the frame (see figure 6).



**Figure 6 A prototype of Ontology Mapping Tool**

Each mapping assertion is uniquely identified by an ID. It concerns two concepts, one from the ontology in the left, the other from the right. The fourth column of the table describes what kind of mapping relation holds between these two. A confidence level is given to indicate the probability that this relation is true. An explanation about the source of the mapping assertion is given in the last column. Note that each field in the table correspond to the meta model of the mapping assertion, as illustrated in figure 1.

When the user selects on one mapping assertion (by clicking the row of that assertion in the mapping table), the corresponding concepts in the two ontologies are highlighted. It is also possible for the user to edit, delete or add mapping assertions.

The four automated functions are controlled by the user through selection and validation actions. The user can refine the result of the text categoriser in order to validate the categorising with respect to the ontologies. Mapping assertion generation is also controlled

interactively, in the sense that the user is able to exclude automatically generated mapping assertions or add mappings that have not been proposed by the system.

The integration of these functions within a single process as depicted in figure 6 results in a method that supports the acquisition and management of mapping links between two ontologies.

To implement the functions presented above, we have used several components and try to integrate them into an architecture using XML representation for exchanging data between different components. Currently parts of the system have been implemented, including the importing of ontology, text categorization and attaching categorization information into concept nodes. The feature vector construction and similarity calculation are under development.

### 3.5 Validation

In order to be a solid contribution, it is necessary to evaluate the work. We have a two steps evaluation process.

To begin with, we conduct an experiment on the product catalogue task [Fensel01], which is explained in section 2.4. The two product catalogues in question are UNSPSC<sup>4</sup> (United Nations Standard Products and Services Code) and Eclass<sup>5</sup>. UNSPSC contains about 20.000 categories organized into four levels. Each of the UNSPSC category definitions contains a category code and a short description of the product. For example, category 43191500 “Personal communication devices”. Eclass defines more than 12.000 categories and is organized in a four-level taxonomy. It is generally understood that UNSPSC classifies products from a supplier’s perspective, while Eclass is from a user’s perspective. The reason for choosing this task is largely because the necessary experiment resources are readily accessible.

For our experiment, we use a fraction of the product catalogues, related to the domain of information technology. Two document sets, each of which contains 500 product descriptions, are accumulated from online IT products store. The two catalogues, together with the document sets are loaded into the system and let the algorithm run upon them to derive mappings between the two catalogues. We assume the effectiveness of our approach can be evaluated using the IR metrics of precision and recall where the correctness and completeness of the mapping assertions are measured. This experiment is currently on going.

In the long run, our plan is to explore the performance in large real-world domains such as the KITH medical patient documents domain [KITH] and the library resource management domain. Finally, given the heuristic nature of the method, we would additionally need some empirical indications that the method really is useful for ontology merging. This, however, requires a field study with real user involvement.

## 4 Related work

There has been a number of previous works on ontology (schema) mapping developed in the context of schema translation and integration, knowledge representation, machine learning ,data mining and digital library.

In the multi database and information systems area, there exist approaches dealing with semantic heterogeneity among distributed, autonomous information sources. In [Batini86], a

---

<sup>4</sup> [www.unspsc.org](http://www.unspsc.org)

<sup>5</sup> [www.eclass.de](http://www.eclass.de)

variety of database schema integration methods were studied and the schema integration process is divided into three major phases: schema comparison, schema conforming and schema merging. [Rahm01] claims that a fundamental operation in the manipulation of schema information is match. The match operation takes two schemas as input and produces a mapping between elements of the two schemas that correspond semantically to each other. The mapping returned by a match operation may be used as input to operations to merge schemas or mediate between schemas. DIKE[Palopoli00], MOMIS[Bergamaschi99] and Cupid[Madhavan01] are systems, which focus on schema matching. OBSERVER[Mena00] uses inter-ontology relationships like synonyms, hyponyms, and hyponyms across terms in different ontologies to translate a query specified using one ontology into another query which uses target ontologies. In [Rahm01], schema matching approaches were classified into schema-level matchers and instance-level matchers<sup>6</sup>. It pointed out that instance-level approaches can be used to enhance schema-level matchers in that evaluating instances reveals a precise characterization of the actual meaning of the schema elements. And in general more attention should be given to the utilization of instance-level information to perform match [Rahm01].

In the research area of knowledge engineering, a number of ontology integration methods and tools exist. Among them, Chimaera [McGuinness00] and PROMPT[Noy00] are the few which have working prototypes. Both tools support the merging of ontological terms i.e. class and attribute names from various sources. The processes start by running a matching algorithm on class names in the pair of ontologies to suggest the merging points. The matching algorithm either looks for an exact match in class names or for a match on prefixes, suffixes, and word root of class names. A user can then choose from these matching points, or proceed on his/her own initiative. PROMPT provides more automation in ontology merging than Chimaera does. For each merging operation, PROMPT suggests the user to perform a sequence of actions on copying the classes and their attributes, creating necessary subclasses and putting them in the right places in the hierarchy. More recent work includes OntoMerger, an ontology translation service [OntoMerge]. The merge of two ontologies is obtained by taking the union of the axioms defining them, and then adds bridging axioms that relate the terms in one ontology to the terms in the other. XML namespaces are used to avoid name clashes. The service accepts a dataset as a DAML file in the source ontology, and will respond with the dataset represented in the target ontology also as a DAML file.

Recently, there have emerged some preliminary studies trying to perform ontology mapping via analysing an extensional description of concepts and deriving mappings by comparing extensional descriptions. [Stumme01] proposes a method called FCA-MERGE, based on the theory of formal concept analysis, for merging ontologies following a bottom up approach and the method is guided by application-specific instances of the given source ontologies that are to be merged. [Agrawal01] uses techniques well known from the area of data mining (association rules) for the task of catalogue integration. [Prasad02] presents an approach using Naïve Bayesian for ontology mapping between two classification hierarchies. Our method is in the line of approaches in this category. Yet we are different from them in the sense that first, information retrieval modals are used to represent concept extension and calculate similarity between concepts and text categorization is used when no extensional description of the ontology is available. Second, we use graphical notations to represent the ontology and

---

<sup>6</sup> Schema-level matchers only consider schema information, including the usual properties of schema elements, such as name, description, data types, relationship types constraints and schema structure. As complementary methods, instance-level approaches can give important insight into the contents and meaning of schema elements.

that makes it easier for the user to understand the model and get an overview. In addition, hierarchical information of the concept is being considered by adding contributions from its sub nodes and related nodes.

## 5 Conclusions

This paper approaches the problem of ontology mismatch that occurs when seeking to introduce semantic interoperability in a network of heterogeneous, distributed ISs. The more specific problems of semantic heterogeneity and anomaly are identified and discussed. A preliminary solution is achieved by introducing and utilising extension analysis in order to facilitate ontology mapping.

In this paper, we present a heuristic mapping method and a prototype mapping system that support the process of semi-automatic ontology mapping. A framework for ontology intension and extension, for mapping assertions and for mapping discovery are introduced. The functional settings for the mapping system are discussed and techniques for text categorisation, feature vector construction and similarity calculation are presented. A prototype implementation of the approach is proposed to fulfil the task of deriving, storage and handling of the mappings.

The mapping method has been inspired by both information retrieval and text categorisation techniques. This approach introduces functionality to support ontology mapping through extension analysis in order to facilitate the utility of the automated part of the process. First, text categorisation is used where no other ontology extension is available. Second, concept hierarchy and relations are used in addition to the extension analysis to generate mapping assertions. Third, a visual language is used to present the ontology to a user. The proposed functionality facilitates the mapping process in that the semantic heterogeneity and the risk for anomalies are reduced. Furthermore, the cognitive overload of the user is reduced in the manual part of the mapping process.

The next logical step in our research would be to gather empirical data about the performance and the applicability of the mapping system as is explained in section 3.5 validation. Further, we should improve the non-functional performance of our prototype implementation with more sophisticated natural language analysis in order to accomplish the challenges of scaling up our mapping approach. In the future, we also plan to investigate the idea of using the obtained mappings within an agent system to facilitate transparent interaction between buyers and vendors in the market place.

## References

1. [Aas99] Aas, K., Eikvil, L.: Text Categorisation: A Survey. Norwegian Computing Center, Oslo (1999)
2. [Agrawal01] Agrawal, R., Srikant, R.: On Integrating Catalogs. Proceeding of the WWW-11, Hong Kong (2001)
3. [Batini86] Batini, C., Lenzerini, M.: A comparative Analysis of methodologies for Database Schema Integration. ACM Computer Surveys. 18(4) (1986)
4. [Bergamaschi99] Bergamaschi, S., Castano, S., Vincini, M.: Semantic Integration of Semistructured and Structured Data Sources. SIGMOD Record 28(1) (1999)
5. [Berners-Lee01] Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American, May (2001) Online at: <http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html>
6. [Bernstein01] Bernstein, P.A.: Generic Model Management: A database infrastructure for schema manipulation. Proceeding of the 9<sup>th</sup> International Conference Cooperative Information Systems (2001)
7. [Brasethvik01] Brasethvik T. and Gulla J. A.: Natural language analysis for semantic document modelling. Data & knowledge Engineering (2001)
8. [DC] <http://www.dublincore.org>
9. [Fensel01] Fensel, D., Ding, Y., Omelayenko, B., Schulten, E., Botquin, G., Brown, M., Flett, A.: Product Data Integration for B2B E-Commerce. IEEE Intelligent Systems 16 (2001)

10. [Hakkarainen99] Hakkarainen, S.: Dynamic Aspects and Semantic Enrichment in Schema Comparison. PhD Thesis. Stockholm University (1999)
11. [Hoferiter02] Hofreiter, B., Huemer, C.: Towards syntax-independent B2B. ERCIM News October (2002)
12. [Joachims98] Joachims, T.: Text categorization with support vector machines: learning with many relevant features. Proceeding of European Conference on Machine Learning (1998)
13. [Kaada02] Kaada, H.: Linguistic workbench for document analysis and text data mining. Master thesis. Norwegian University of Science and Technology (2002)
14. [KITH] <http://www.kith.no>
15. [Klein01] Klein, M.: Combining and relating ontologies: an analysis of problems and solutions. Proceedings of the 17<sup>th</sup> International Joint Conference on Artificial Intelligence (IJCAI-01). Ontologies and Information Sharing Workshop. Seattle, USA. (2001)
16. [Labrou99] Labrou, Y., Finin, T.: Yahoo! as an ontology- using yahoo! categories to describe documents. Proceedings of the 8<sup>th</sup> international conference on Information and knowledge Management. (1999)
17. [Madhavan01] Madhavan, J., Bernstein, P.A., Rahm, E.: Generic Schema matching using Cupid. Proceedings of VLDB (2001)
18. [McGuinness00] McGuinness D., Fikes R., Rice J., Wilder S.: An environment for merging and testing large ontologies. Proceedings of the 7<sup>th</sup> International Conference on Principles of Knowledge Representation and Reasoning. Colorado, USA. (2000)
19. [Mena00] Mena, E. and Illarramendi, A. and KashyapV., Sheth, A.P.: OBSERVER: an approach for query processing in global information systems based on interoperation across pre-exist ontologies. International Journal on Distributed and Parallel Databases (2000)
20. [Noy00] Noy N., Musen M.: PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. Proceedings of the AAAI-00 Conference. Austin, USA (2000)
21. [ODP] <http://dmoz.org/>
22. [OntoMerge] <http://cs-www.cs.yale.edu/homes/dvm/daml/ontology-translation.html>
23. [Palopoli00] Palopoli, L., Terracina, G., Ursino, D.: The System DIKE: Towards the Semi-Automatic Synthesis of Cooperative Information Systems and Data Warehouses. ADBIS-DASFAA Symposium 2000. Matfyz Press (2000)
24. [PH] <http://css.mit.edu/ph/>
25. [Prasad02] Prasad, S., Peng, Y., Finin, T.: Using explicit information to map between two ontologies. Proceedings of the Workshop of Ontology in Agent Systems (OAS 2002). Bologna, Italy (2002)
26. [Rahm01] Rahm E., Bernstein P.A.: A survey of approaches to automatic schema matching. VLDB journal. 10(4) 334-350 (2001)
27. [RML] Referent Modeling Language <http://www.idi.ntnu.no/~ppp>
28. [Salton83] Salton, G., McGill, M. J.: An Introduction to Modern Information Retrieval. McGraw-Hill (1983)
29. [Solvberg98] Solvberg A.: Data and what they refer to. In: Chen, P., Akoka, J., Kangassalo, H., Thalheim, B., (eds.): Conceptual Modeling: Current Issues and Future Trends. LNCS 1565. Springer Verlag (1999)
30. [Stumme01] Stumme, G., Maedche, A.: FCA-Merge: Bottom-up Merging of Ontologies. Proceedings of the International Joint Conference on Artificial Intelligence IJCAI'01. Seattle, USA (2001)
31. [Su02] Su, X., Ilebrikke, L.: A Comparative Study of Ontology Languages and Tools. Proceeding of the 14<sup>th</sup> Conference on Advanced Information Systems Engineering (CAiSE'02). Toronto, Canada, May (2002)
32. [Uschold96] Uschold, M., Gruninger, M.: Ontologies: Principles, Methods and Applications. Knowledge Engineering Review 11(2) (1996)
33. [Wache01] Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: Ontology-Based Integration of Information - A Survey of Existing Approaches. Proceedings of the 17<sup>th</sup> International Joint Conference on Artificial Intelligence (IJCAI-01): Ontologies and Information Sharing Workshop. Seattle, USA (2001)
34. [Wiederhold99] Wiederhold, G., Mitra, P., Jannink, J.: Semi-automatic Integration of Knowledge Sources. Proceedings of the 2<sup>nd</sup> International Conference on Information Fusion (FUSION'99). California, USA (1999)
35. [Wilson02] Wilson, M., Matthews, B.: Migrating Thesauri to the semantic web. ERCIM News October (2002)