Department of Information Engineering and Computer Science DISI

UNIVERSITY OF TRENTO - Italy

DISI - Via Sommarive 14 - 38123 Povo - Trento (Italy)
http://www.disi.unitn.it

REPORT ON METHODS AND ALGORITHMS FOR LINKING USER-GENERATED SEMANTIC ANNOTATIONS TO SEMANTIC WEB AND SUPPORTING THEIR EVOLUTION IN TIME

Tobias Burger and Olga Morozova and Ilya Zaihrayeu and Pierre Andrews and Juan Pane

January 2010

Technical Report # DISI-10-010

INSEMTIVES
FP7-ICT-2007-3
Contract no.: 231181
www.insemtives.eu

INSEMTIVES

Deliverable <2.2.2 / 2.2.3>

Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time

Editor:	Tobias Bürger, STI, University of Innsbruck
Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	30.11.2009
Actual delivery date:	30.11.2009
Version:	0.5
Total number of pages:	21
Keywords:	semantic annotation, Linked Data, interlinking algorithms, ontology evolution

Abstract

The INSEMTIVES project is concerned with increasing the amount of annotations available for various kinds of content, predominantly in a structured form using controlled vocabularies and aligned to content on the Semantic Web. In this deliverable we discuss solutions for generating links to external resources on the Semantic Web from structured annotations and how these annotations can evolve in time if controlled vocabularies, referred in these annotations, change. We discuss existing techniques for both problems and propose a set of methods (a) applicable to interlink various kinds of content and (b) to support the evolution of annotations in time.

Disclaimer

This document contains material, which is the copyright of certain INSEMTIVES consortium parties, and may not be reproduced or copied without permission.

All INSEMTIVES consortium parties have agreed to full publication of this document.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the INSEMTIVES consortium as a whole, nor a certain party of the INSEMTIVES consortium warrant that the information contained in this document is capable of use, or that use of the information is free from risk, and accept no liability for loss or damage suffered by any person using this information.

Impressum

[Full project title] INSEMTIVES – Incentives for Semantics

[Short project title] INSEMTIVES

[Number and Title of Workpackage] WP2: Models and methods for creation of lightweight, structured knowledge

[Document Title] D2.2.2 / D2.2.3 Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time

[Editor: Name, company] Tobias Bürger, STI, University of Innsbruck

[Work-package leader: Name, company] Ilya Zaihrayeu, University of Trento

Acknowledgement

The project is co-funded by the European Union, through the ICT Cooperation programme http://cordis.europa.eu/fp7/cooperation/home_en.html

Executive summary

The aim of the INSEMTIVES project is to increase the number of semantic annotations on the Web and thus to overcome the knowledge acquisition bottleneck that prevents the development of the Semantic Web. The two main approaches implemented for dealing with this problem in the INSEMTIVES project are (1) to propose incentives that motivate humans to provide semantic annotations, (2) to propose methods to bootstrap semantic annotations automatically with low involvement of humans in the annotation process. The methods for annotations bootstrapping are discussed in deliverable D2.2.1 ("Report on methods and algorithms for bootstrapping Semantic Web content from user repositories and reaching consensus on the use of semantics"). Methods discussed there are supposed to generate annotations by exploiting the context of resources and to support users by simplifying the annotation task by providing annotation suggestions or auto completion of annotations both as a means to lower the users' effort in providing annotations. Possible annotation suggestions could be offered by controlled vocabularies as well as could be taken from external resources which are semantically interlinked with given data. Based on the outcome of D2.2.1, the present deliverable investigates approaches for linking user-generated semantic annotations and for supporting their evolution in time. Interlinking of data enables the shared use of semantics on the Web and reflects Tim Berners-Lee's vision of "the Web of Data" in which data on the Web is machine readable and interlinked. The procedure of interlinking of data can be done manually, but it is, in general, too labor intensive. Therefore, in order to interlink large data sets, automatic and semi-automatic methods were developed. The present deliverable investigates methods and algorithms for linking data as well as supporting the evolution of annotations in time. It discusses the state of the art and proposes future directions on how to advance it for both topics.

List of authors

Company	**Author**
University of Innsbruck	Tobias Bürger
University of Innsbruck	Olga Morozova
University of Trento	Pierre Andrews
University of Trento	Ilya Zaihrayeu
University of Trento	Juan Pane

Contents

List of Figures

1 Introduction

Semantic content is the backbone of the Semantic Web, a Web in which content is available in a formally represented, machine readable way. Annotations are one form of semantic content, which are supposed to make information which is implicit and inherit in content, explicit [6].

The life cycle of resource annotations consists of seven phases as explained in [1]: (1) *Publication*, (2) *bootstrapping*, (3) *annotation*, (4) *ontology maturing*, (5) *annotation evolution*, (6) *linking to external repositories*, and finally (7) *use*. In the present deliverable we consider two phases of this process, namely *evolution* and *linking* as schematically illustrated in Figure 1.

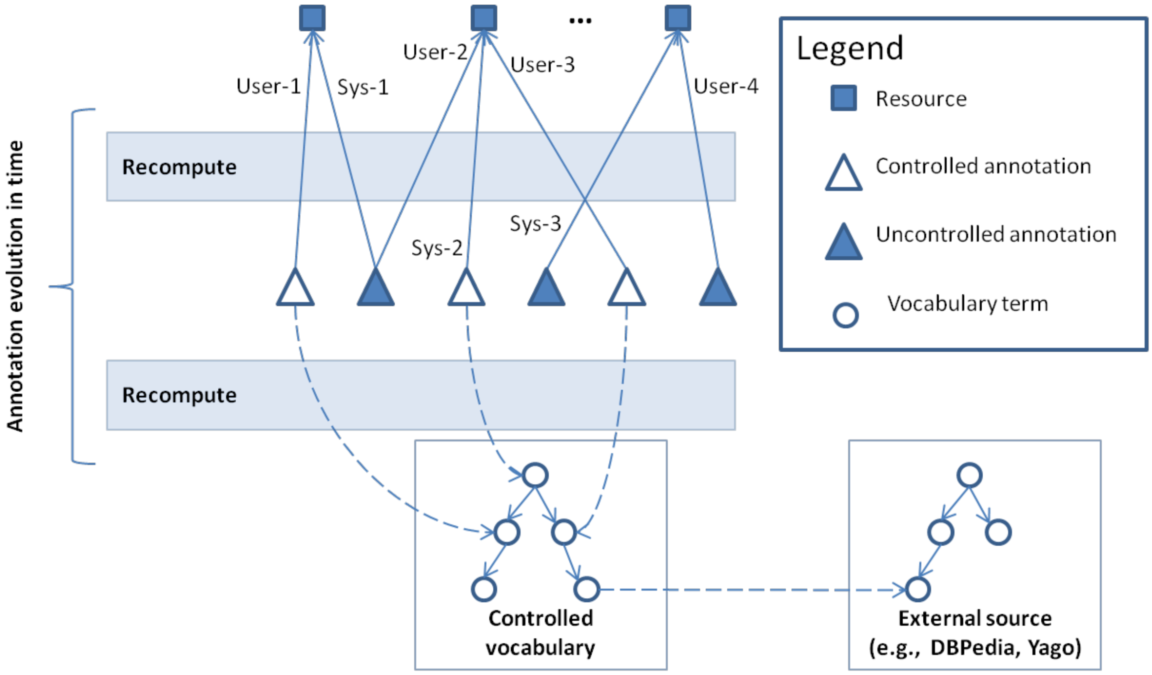

Figure 1: Resource annotation graph

In this figure we show a model of annotated resources in the form of a graph which has two kinds of vertices: resource vertices and (controlled and uncontrolled) annotation vertices; and whose edges are directed from annotation vertices to resource vertices and labeled with the names of users or systems who created these annotations. Annotation vertices that represent controlled annotations are further mapped to the concepts in the underlying controlled vocabulary, whose concepts, in turn, can be linked to external sources.

This deliverable combines the deliverables D2.2.2 ("Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time (internal deliverable)") and D2.2.3 ("Report on methods and algorithms for linking user-generated semantic annotations to Semantic Web and supporting their evolution in time (final version)"). It is dedicated to analyze and propose methods to link user generated annotations to Semantic Web content and to support the evolution of annotations in time. Both phases of the annotation life cycle covered in this deliverable are described in more detail in [1], but will also be motivated in subsequent sections of this deliverable.

The remainder of the document is organized as follows: Section 2 discusses how the links from the controlled vocabulary concepts to external sources are computed, whereas Section 3 discusses how the links from annotations to resources as well as links from controlled vocabulary concepts to controlled annotations are recomputed when the structure of the controlled vocabulary changes. Section 4 concludes the deliverable with an outlook to future steps.

2 Linking

2.1 Problem statement

Linking to external repositories, as defined in deliverable D2.2.1 [1], is a process in which (internal) controlled annotations are mapped to entries in external repositories through the definition of links from the concepts and instances defined in the controlled vocabulary (to which controlled annotations are mapped to) to external entries and the definition of semantic relations that hold between them. Linking as such is an integral part of the relation annotation model described in [6].

In this deliverable we define linking more precisely and adhere to the definition of "linking" as used in the research on Linked Data. Technically, Linked Data refers to data published on the Web in such a way that it is machine-readable, its meaning is explicitly defined, it is linked to other external datasets, and can in turn be linked to from external datasets.

A huge momentum has recently been gained in the Semantic Web research by the ongoing implementation of a vision of a Web of Data formulated by Tim Berners-Lee in which formerly fragmented data is connected and interlinked with each other based on the so-called Linked Data principles. Since a few years from now, the so-called Linked Open Data (LOD) cloud which represents a huge interconnected dataset is steadily growing (cf. Figure 2).

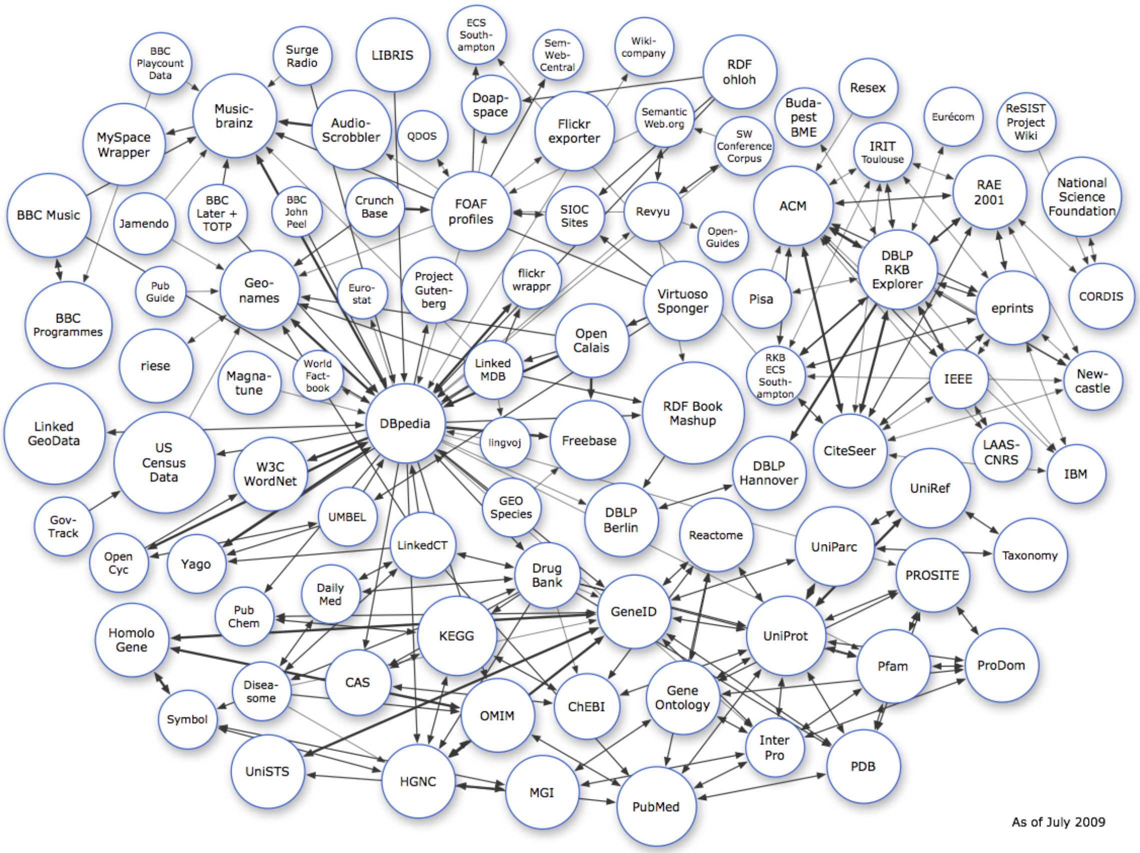

Figure 2: The Linked Data cloud (as of July 2009)

In early 2007 the LOD community project has been launched within the W3C Semantic Web Education and Outreach group. It bootstraps the Web of Data by publishing datasets using the Resource Description Framework (RDF), the metadata model primarily used on the Semantic Web. RDF enables automated software to store, exchange, and use machine-readable information distributed throughout the Web, in turn allowing users to deal with the information with greater efficiency and certainty. Currently, the LOD project includes nearly 100 different datasets (cf. Figure 2), ranging from rather centralized ones, such as DBpedia[1], a structured

[1] http://dbpedia.org/About

version of WikiPedia, to those that are very distributed, for example the FOAF-o-sphere. The current LOD cloud contains data from diverse domains such as people, companies, books, scientific publications, films, music, television and radio programs, genes, online communities, statistical or scientific data [2]. Datasets were contributed both by researchers as well as by industry. From one billion triples and 250k links in mid-2007 the LOD dataset has grown to more than 4.5 billion triples and 124 million links in 2009, representing a steadily growing, open implementation of the Linked Data principles.

The key success factor of the LOD movement is the simplicity of its underlying principles:

1. Use URIs as names for things;
2. Use HTTP URIs so that people can look up those names;
3. When someone looks up an URI, provide useful information, using the standards (RDF, SPARQL);
4. Include links to other URIs, so that they can discover more things.

The main tasks that have to be performed in order to publish data as Linked Data are (i) to assign consistent URIs to data published, (ii) to generate links, and (iii) to publish metadata which allows further exploration and discovery of relevant datasets. The steps 1-3 can to some extent be automated, using tools such as D2R or Virtuoso, therefore this deliverable focuses on the issue of link generation as the major problem. The main problem which arises is the issue of finding the matching concepts in datasets to be interlinked and to name the relationships between the interlinked concepts (using defined relations such as owl:sameAs, foaf:birthPlace, foaf:homeTown or others).

Below we provide two examples detailing how linking structured annotations to external resources can look like.

- **Linking on the instance level:** An annotation is used to identify the appearance of an instance in a resource such as the character *Bing*. This means that a controlled annotation uses the instance Bing of type character. A link could now connect using a pre-defined relation such as *owl:sameAs* with an external dataset such as DBPedia in which information about *Bing* is available. This type of link reflects that the same individual thing is referenced both in an annotation as well as an external resource (dataset).
- **Linking on the conceptual level:** A structured annnotation identifies the presence of a specific concept in a resource, e.g., *dog*. Another external vocabulary might define more specific dogs such as *shepherds* or *labradors*. A link between both could indicate that one concept is more general than the other using for instance relations such as *skos:broader* or *skos:narrower* from the SKOS mapping vocabulary.[2]

In the meanwhile several approaches exist for semantically linking data: RDF links can either be set manually supported by a set of tools including URI search and recommendation engines such as Uriqr[3], Sindice[4], or MOAT[5]. Furthermore they can be generated using automated linking algorithms. We will discuss both manual and (semi-) automatic approaches in the following.

2.2 State of the art on interlinking algorithms

2.2.1 Manual approaches

In order to manually set links three steps have to be followed: (1) choose a suitable dataset to link to, (2) search in the chosen data set for URI references you intend to link to, and (3) define a suitable predicate for link between both datasets. A list of tutorial which illustrate the manual interlinking task can be found online.[6] Many software tools have been developed in order to support data publishers in the interlinking task. This includes Linked Data browsers such as Tabulator or Disco, URI search engines such as Uriqr or Sindice or more specific frameworks for manually interlinking tags with Semantic Web URIs such as MOAT.[7]

[2] `http://www.w3.org/TR/skos-reference/`
[3] `http://dev.uriqr.com/`
[4] `http://www.sindice.com/`
[5] `http://moat-project.org/`
[6] `http://linkeddata.org/guides-and-tutorials`
[7] `http://linkeddata.org/tools`

2.2.2 Semi-automatic approaches

Semi-automatic approaches for interlinking data have been proposed that require human involvement for accepting, rejecting or modifying interlinking suggestions. One of such examples is user-contributed interlinking.

The term "User-contributed interlinking" (UCI) was suggested in [10] and exploits the wisdom of the crowds to correct or extend automatically generated links. The UCI approach was implemented for interlinking the riese dataset, which contains statistics about the EU, with the other datasets from the LOD cloud. The datasets were interlinked automatically and users were invited to add new links or remove existing ones. This option is available on every page by clicking the button "I know more" and is supported by an easy-to-use link editor that allows deleting of existing links or adding new ones. Incentives behind this approach are similar to the ones that drive humans to write Wikipedia articles.

2.2.3 Automatic approaches

For interlinking large datasets automatic interlinking algorithms were proposed. Typically these algorithms work based on the similarity of entities within both datasets. These approaches are generally motivated by related work on record linkage and duplicate detection from the database community, as well as on ontology matching techniques from the knowledge representation community. Automatic interlinking algorithms can be divided into four main classes:

- String matching algorithms
- Common key or pattern-based algorithms
- Graph matching algorithms
- Template-based matching algorithms

String-matching algorithms compare labels using similarity metrics. String matching was, for instance, implemented for interlinking the Jamendo and Geonames datasets. Jamendo contains information about location of artists in a dense form (city, country). String matching algorithms can be applied if literal strings reliably provide sufficient disambiguation which is only rarely the case. On the contrary, many book titles, author names, songs, etc. have the same names. **Common key matching, or pattern-based algorithms** are used to interlink datasets in which generally accepted naming schemata can be found (e.g., ISBN numbers in the publication domain, ISIN identifiers in the financial domain, or IDs in Musicbrainz). These type of algorithms were, for instance, applied for connecting DBpedia with the corresponding Book Mashup URIs. The prerequisite for applying this type of algorithms is the availability of common identifiers. More complex **graph-matching algorithms** are based on similarity of several properties in time, so-called graph similarities or cluster similarities. Such kind of algorithm was applied, for instance, by Yves Raimond in [28] for interlinking Jamendo and Musicbrainz. The developed sophisticated graph-based interlinking algorithm outputs interlinking decisions for a whole graph of resources: a matching artist, the corresponding matching records and the matching tracks on these records. The algorithm performed in [28] works quite well as it makes low number of incorrect positive links, so it doesn't require human post-processing. However, the algorithm works on the assumption that the two datasets to interlink conform to the same ontology, or that there is a one-to-one mapping between terms in the first ontology and terms in the second. For applying this algorithm by interlinking the datasets built on the different ontologies, an ontology matching task should first be performed and the resulting correspondences between ontology terms should be included in the algorithm. The algorithm could be extended by using weights. For example, in [15] by interlinking CIS dataset with DBpedia weights were additionally associated with concepts, in order to start from the most likely matches, whereas Wikipedia inter-article links played the role of weight indicators. A **template-based matching algorithm** is implemented in the Silk link discovery engine described in [32]. Silk is able to generate owl:sameAs and other link types and it is based on a declarative language for specifying link conditions to be met in the automated link discovery process. It furthermore is capable of resolving heterogeneities based on the use of different schemata. Link conditions in Silk can be expressed using a combination of RDF path expressions, similarity metrics, and aggregation functions. It supports a set of translation functions to resolve syntactic mismatches and to translate between different vocabularies. As indicated in the publication, the link discovery framework shields quite good results.

2.3 Proposed solution

Given the requirements for annotation and linking in INSEMTIVES formulated in [6], linking is relevant as a means for annotating both textual and media resources with external resources on the Semantic Web, or more precisely the Web of Data.

The process of interlinking is generally perceived as a 5-step process as depicted in Figure 3. The steps can be grouped into three parts, reflecting the publication, annotation, and the linking phase of the annotation life cycle. The steps to be taken are as follows:

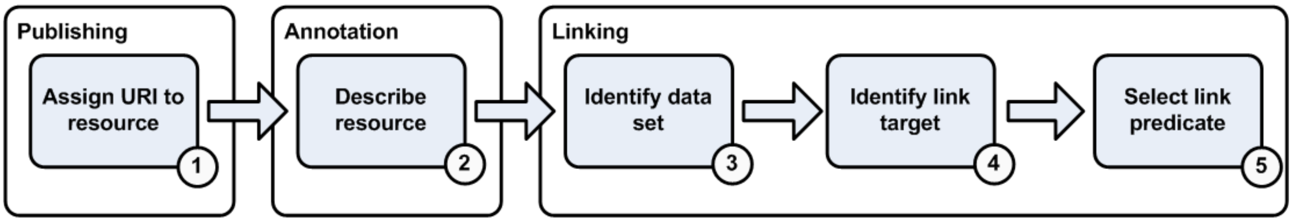

Figure 3: The process of interlinking

1. **Assign URI**: In order to interlink a resource with another resource, a URI has to be assigned to it in order to identify it.
2. **Describe resource**: The resource identified with an URI has to be described in an interoperable way.
3. **Identify target dataset:** A suitable dataset has to be choosen to which links will be made.
4. **Identify link target:** A search has to be performed to identify a URI reference to link to in the target dataset.
5. **Select link predicate:** Finally a predicate has to be selected which reflects the relation between the source URI and the link target.

The linking phase focuses on the final three steps of the process and is the main focus of methods for interlinking.

Revisiting the principles for Linked Data and methods for interlinking listed in the previous section, we can observe that both have been mainly applied to textual resources so far. For this type of resource a considerable amount of methods have been proposed, that will be tested for their applicability in the course of the INSEMTIVES project. For interlinking media, methods are lacking. This is why we put an emphasis on linking methods for this resource type. We presented in [13] how the Linked Data principles can be applied to media resources and list a set of possible interlinking methods applicable for interlinking multimedia resources at a fine-grained level. In the following we review various methods and tools used for interlinking resources on the Web of Data.[8]

2.3.1 Manual methods

Recently *User Contributed Interlinking* (UCI) has been introduced [10, 11], a manual interlinking methodology which relies on the end user as a source of qualitative information. UCI has been applied to enrich the Eurostat dataset [10]. A recent proposal, called CaMiCatzee [12] implements UCI for multimedia. CaMiCatzee allows people to semantically annotate picture on Flickr and to query for person's using their FOAF documents, URIs or person names.

Manual method for interlinking multimedia could be combined with incentives such as *Game Based Interlinking* (GBI), following the principles set forward by Louis van Ahn with his *games with a purpose*[9] [33]. One approach is to make the *interlinking* of resources *fun* and to hide the task of interlinking behind games. This

[8] The following discussion on interlinking methods also appears in [13], a paper co-authored by Hausenblas, Troncy, Bürger, and Raimond which is in turn based on an earlier publication by Bürger and Hausenblas [4].

[9] `http://www.gwap.com/`

is related to UCI but with the main difference that the user is not aware of him contributing links as his task is hidden behind a game.

GBI seems to be a promising direction for multimedia interlinking. The most interesting examples to build on are Ahn's ESP games in which users are asked to describe images, or Squigl[10] in which users are asked to trace objects in pictures. Another interesting approach is followed by OntoGame whose general aim is to find shared conceptualizations of a domain. OntoGame players are asked to describe images, audio or video files. Users are awarded if they describe content in the same way. Further exemplary games are OntoTube, Peekaboom, or ListenGame which hide the complexity of the annotation process of videos, images or audio files respectively, behind entertaining games. These approaches together with appropriate browsing interfaces for multimedia assets could be a promising starting point to let users draw meaningful relations between objects and their parts.

2.3.2 Collaborative interlinking

Collaborative approach to interlinking of resources could be followed using Semantic Wikis. Semantic Wikis extend the principles of traditional Wikis such as collaboration, easy use, linking and versioning with means to type links and articles via semantic annotations [29]. Some of the systems support the annotation of multimedia objects including Semantic Wikis with dedicated multimedia support such as Ylvi [26], MultiMakna [20]. Most of these systems however treat a multimedia object as part of an article in which they appear. Thus, they do not allow specific annotations of it or treat them in the same manner like articles which can be only annotated globally. MultiMakna allows to assign annotations to temporal segments in videos through the use of an $appliesTo$-relation. While annotations may be constrained to its temporal context, to the best of our knowledge, links can only be established between articles and not segments.

Another Semantic Wiki with multimedia support is MetaVidWiki (MVW)[11] which enables community engagement with audio/visual media assets and associative temporal metadata. MVW extends the popular Semantic MediaWiki [16] with media specific features such as streaming, temporal metadata, and viewing and editing of video sequences. MVW supports the addressing and linking between temporal fragments. Segments of videos can be treated like "articles", referenced via URIs which support time intervals according to the temporalURI specification [23] and metadata about them can be exported in CMML [24].

2.3.3 Semi-automatic methods

Semi-automatic interlinking methods consist in combining multimedia-analysis techniques with human feedback. Analysis techniques can process the content itself or the context surrounding the content such as the user profile in order to suggest potential interlinking. The user would need to accept, reject, modify or ignore those suggestions. Inspiration for this type of approach can be found in the area of semi-automatic multimedia annotation.

Emergent Interlinking (EI) is another approach based on the principles of Emergent Semantics whose aim is to discover semantics through observing how multimedia information is used [4]. This can be essentially accomplished by putting multimedia resources in context-rich environments being able to monitor the user and his behavior. In these environments, two different types of context are present: (i) static or structural context, which is derived from the way how the content is placed in the environment (e.g. a Web page) and (ii) dynamic context, which is derived from the interactions of the user in the environment (e.g. his browsing behavior, which links he follows, or on which object he zooms). The assumption is that in appropriate environments, the browsing path of a user is semantically coherent and thus allows to derive links between objects which are semantically close to each other.

2.3.4 Automatic methods

Finally, automatic interlinking of fragments of a multimedia resource can be achieved by purely analyzing its content. For example, in the case of such a musical audio content, the audio signal can be analyzed in order to derive a temporal segmentation. The resulting segments can be automatically linked to musically relevant

[10] `http://www.gwap.com/gwap/gamesPreview/squigl/`

[11] `http://metavid.org/wiki/`

concepts, e.g. keys, chords, beats or notes. An application of automatic interlinking of media fragments in the music domain is Henry[12] [27]. Henry aggregates music processing workflows available on the Web and applies them on audio signals to dynamically derive temporal segmentations and interlink these different segments with Web identifiers for music-related concepts.

2.4 Going beyond the state of the art

As of today, most interlinking methods use heuristics to generate links between datasets almost fully automatically. These methods can be applied if RDF descriptions are already existing or if these can be easily generated, e.g., from texts. In INSEMTIVES each case study demands for different interlinking methods, given the dominant media types presented in processed collections, be that media files, Web services, or textual documents. We will select and develop methods on a case-by-case basis for each of them. A considerable challenge are methods for interlinking media, because on binary data such as multimedia content, automatic interlinking algorithms can not be applied.

The proposed methods might differ considerably in terms of efficiency meaning quality and amount of produced annotations versus effort needed to generate the annotations. The methods discussed in the previous section can be arranged in a three-dimensional matrix with the dimensions time, quality and amount of annotations as depicted in Figure 4:

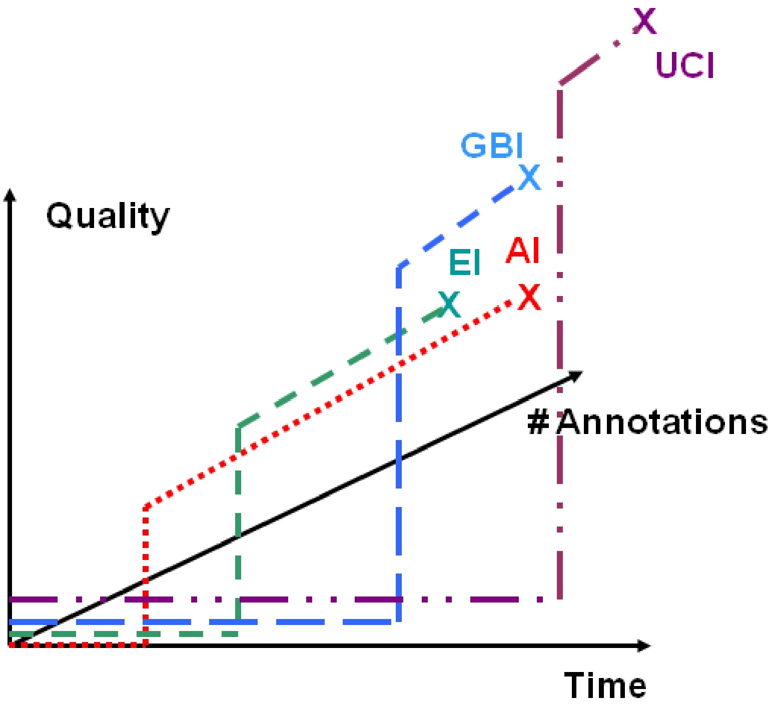

Figure 4: Prospected effectiveness of interlinking methods for multimedia

While UCI might reach the highest quality and needs the highest amount of time from an end user perspective, automatic interlinking might produce the greatest amount of annotation and thus links with the least amount of time and manual effort needed.

Automatic approaches could be used to initialize the interlinking process, while end users have to be involved as a source of high-quality initial annotations on which automatic media analysis solutions can be trained and further improved. Potential candidates for methods to involve end users are collaborative methods or methods for emergent interlinking. The usefulness of these methods has again be determined on a case-by-case basis. Game-based interlinking methods are considered as a further promising option for interlinking of various kinds of content.

[12] http://dbtune.org/henry

3 Evolution in Time

3.1 Problem statement

As defined in deliverable D2.2.1, the *annotation evolution* is a process in which links from controlled and uncontrolled annotations to resources are recomputed as the structure of the controlled vocabulary changes. The goal of annotation evolution is to maintain and possibly improve the quality of annotations that might degrade as the result of the above described dynamic factor. The annotation evolution process primarily concerns two kinds of annotations: (1) manually added or automatically extracted uncontrolled annotations which can be converted to controlled ones; and (2) automatically generated controlled annotations such as those extracted by the bootstrapping process or those previously recomputed by the annotation evolution process and which can now be recomputed given the new knowledge added to the controlled vocabulary. The annotation evolution process does not try to improve (unless explicitly required by the user) manually added controlled annotations as they are expected to be of a higher quality than annotations of the two kinds described above.

Below we provide a non exhaustive list of examples of situations which have to be handled by the annotation evolution process:

- two or more concepts from the controlled vocabulary were merged into a new concept. Controlled annotations that used any of the merged concepts need to be updated to the new concept or be converted to uncontrolled annotations;

- a new term is added to a concept in the controlled vocabulary. Syntactically, the new term is equal to preexisting uncontrolled annotations (i.e., it has the same spelling). The problem is to compute which of the uncontrolled annotations need to be re-mapped to the term concept because they have the same meaning (and, as the result, become controlled annotations) and which ones need to remain uncontrolled annotations as they have a different (yet to be explicitly defined) meaning;

- following the annotation evolution process, the system re-mapped some uncontrolled annotations $an_1, an_2, \ldots, an_n$ (e.g., which used the term “Java”) to a newly added concept c_1 (e.g., “the Java island”) with term t_1 (e.g., “Java”) in the controlled vocabulary. In a later moment in time, yet another concept c_2 (e.g., “the Java beverage”) with the same term t_1 (i.e., “Java”) was added to the vocabulary. The problem is to compute which of the previously re-mapped annotations $an_1, an_2, \ldots, an_n$ need to be re-mapped to c_2;

- a concept is deleted from the controlled vocabulary. The problem is to compute which controlled annotations that used this concept need to be converted to uncontrolled annotations and which of those need to be re-mapped to a more general concept that remain in the controlled vocabulary;

3.2 Related work

The problem presented in Section 3.1 focuses on how the evolution of the controlled vocabulary affects the existing annotations in the system. In this regard, to the best of our knowledge, there is not much work done to deal with this *propagation problem* in annotation or tagging systems (except for [19]), since most of the current models do not treat annotations as semantic annotations, but as syntactic ones [34]. Recent approaches use semantics in annotations, but most of them assume a predefined controlled vocabulary or ontology as the basis for the semantics [22] [3], therefore the problem this document tries to focus on is not present since the underlying controlled vocabulary is assumed to be static.

Similar problems were addressed in ontology evolution approaches [17] [30] [25]. Particularly related to the problem presented in Section 3.1 is the *propagation* of new knowledge when this is included into existing ontologies. According to [7], ontology evolution is “*the process of modifying an ontology in response to a certain change in the domain or its conceptualization*”. In the context of our work, the modification of the controlled vocabulary is done in response to a manual change made by the user or via an automatic consensus process that extracted a new concept and relation according to users’ annotations. Most ontology evolution approaches consider a step or phase where this new knowledge has to be propagated to the affected resources; this is the phase of the ontology evolution which is most related to the problem presented in this Section.

The ontology evolution strategy presented in [30] consists of six phases for evolving the ontology, with the main goal of maintaining the consistency of the ontology after the changes have been applied. The phases are:

capturing : knowing which changes are necessary,

representation of the changes in a suitable format with the correct granularity, e.g., elementary versus composite changes,

semantics of change tries to understand the implications of a change, for example, if a class is deleted, what should happen with its instances,

implementation asking the user for confirmation showing the implications of the change,

propagation to the affected instances and possible depending ontologies,

validation showing the user the performed changes for validation.

However, this strategy was created for a mono-user environment, normally for helping an expert in the task of changing the ontology; therefore, it does not take into account issues that are important in collaborative schemas such as concurrent modification, conflicts an others.

In [31] the authors presented a model to specify the changes in terms of desired outcome, in contrast to desired steps to achieve this desired outcome. The system then uses the conceptual architecture based on a reconfiguration problem using graph search to select between several possible changes.

The work presented in [18] focuses more its attention on the effects of possible changes to an ontology over depending applications and services. The authors propose to log the usage of the ontology to detect “hot spots” to trace which applications and services might be affected by a change in the particular “hot spot”. The work also considers that a log file that traces all the steps and modifications of the ontology is necessary in order to inform the ontology users of why the ontology has changed, in contrast to informing the users only of how the ontology has changed. Similarly, [17] proposes to keep track of the changes on the ontology, as well as the causes of these changes. This could help users to understand the changes, and whether the new meaning still conforms to the original use.

Haase et al. [9] propose an ontology evolution mechanism by suggesting ontology changes based on similarities and correlation between users, their resources and ontologies. The changes are suggested to users when needed, for instance, when certain conditions have been met, e.g., when a defined number of new concepts are created in a similar ontology, and the user has many similar resources that could use these new concepts. This approach could be relevant to our work by showing users the changes suggested by the consensus process defined in the deliverable 2.2.1 [1], letting them know the implications of the changes over their annotations, and the causes of these changes [17][18].

In [25] the authors present a framework for detecting inconsistencies during the process of evolution of an ontology. They focus in logical consistencies, i.e., that the ontology conforms to the set of rules defined in the logical theory. The main contribution of this work is the definition of an approach to identify the set of rules that causes the ontology to be logically inconsistent (as opposed to only knowing that the ontology is inconsistent by using some logical reasoner) and the definition of a set of rules to guide the ontology manager in the process of resolving the inconsistencies by weakening the restrictions in the new or altered axioms.

Even though [21] argues that ontology evolution is not the same as schema evolution, the authors also state that some similarities can be drawn and much of the theory for schema evolution could be reused. The work of [8] presents a semantic approach for schema evolution in object oriented databases that proposes a framework to deal with change propagation in data instances. The scope of the work is similar to the scope of the current document, where the focus is on the effects of changes of the controlled vocabulary over the existing annotations. The work of [8] uses description logics to define the formal framework to support reasoning for checking consistency of changes.

The approach presented in [19] is the most similar work to the scope of this document. The authors presented a rule-base model for dealing with changes in an underlying ontology and the propagation of these changes to the referring annotations for existing resources. The work focuses mainly on: **i.** how to detect which annotations are affected by the change in the ontology by using the *Corese* semantic search engine, and **ii.** how to automatically update the affected annotations based on a set of predefined rules. These two steps are very relevant in our problem statement, and will be adopted in our model. The difference with this work and our problem rely on the fact that the authors did not focus on how the changes are made, which in most cases should not be relevant, except for the fact that in collaborative settings and when the changes are a result of an automatic consensus mechanism, the evolution strategy could be dependent on the annotators, therefore, a fully automatic evolution strategy could not always be applied. The authors also assume that all annotations have to be related to a specific part of the ontology, which is not our case (although is our goal).

3.3 Proposed solutions

Considering the usage pattern and the type of the controlled vocabulary proposed to be used in Insemtives, we focus only on a subset of changes over the controlled vocabulary (from the full list in [30] and a more detailed list in [19]) and therefore only a reduced (simpler) version of the ontology evolution problem. Mainly, we focus on the addition of new concepts in the form of more general or less general relations associated to existing concepts of a taxonomy. This addition of new concepts is based on the use of free-text tags (uncontrolled annotations as defined in deliverable 2.1.1 [5])that were applied during the annotation of resources, and for which we try to extract semantics, either via a (semi) automatic process of consensus, of by allowing users to manually relate the new terms to the existing controlled vocabulary.

As mentioned in [17] and [19], changes in the controlled vocabulary could cause side effects in the existing annotations. Some of these side effects are explained in more details in Section 3.1 and [19]. In order to deal with these side effects, we propose to adapt and extend the ontology evolution process defined in [30] to include collaborative features as can be seen in Figure 5. A more detailed architecture is also presented in [19].

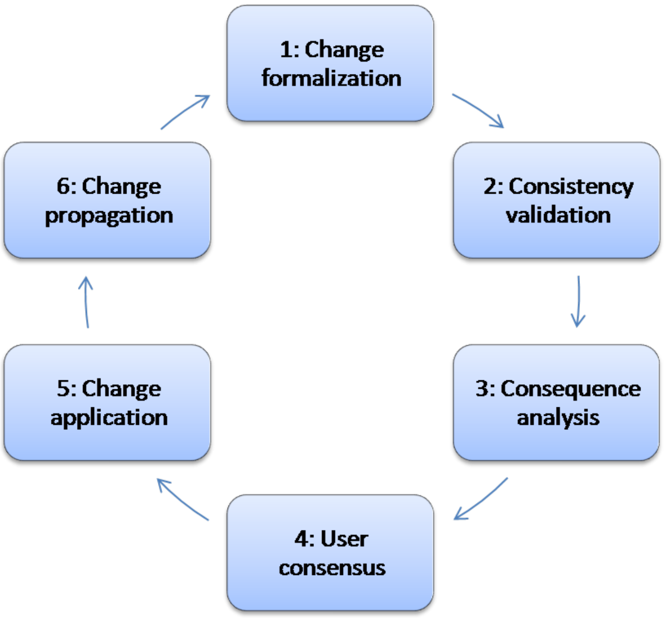

Figure 5: Annotation evolution process

The annotation evolution process (Figure 5) is defined as a cyclic process that continuously specifies the changes needed to evolve the underlying controlled vocabulary and the effects of these changes over the annotations. The process consists of six steps:

1. **Change formalization**: In this step the change to be applied is defined. This change my be defined as a result of an automatic consensus process on the use of uncontrolled annotations using some correlation analysis, or by a manual change request defined by the user.

2. **Consistency validation**: Once the change is defined, we need to check whether this change is compatible (consistent) with the current structure in the controlled vocabulary. According to [25] there are several types of consistencies, but we focus mainly in the *logical consistency* which is defined as "the conformity of the ontology to the underlying logical theory of the ontology language". For example,

considering the use of taxonomies in our work (and that taxonomies are trees), we cannot allow cycles in the relations between concepts. Therefore, when manually relating an uncontrolled annotation to the existing controlled vocabulary, e.g., by stating that A is more specific than B, we need to check whether the controlled vocabulary does not already contains a rule that defines B as more specific than A. If this is the case, the system should tell the user interactively that this manually defined change is not possible, possibly showing the user the rules that are in conflict with desired change and possible solutions (e.g., by merging the two concepts). When changes are defined as result of an automatic process, this consistency check should already be part of the underlying algorithm. The work presented in [25] defines a framework for consistency checking.

3. **Consequence analysis**: once we are sure that the change will not leave the controlled vocabulary in a inconsistent state, we need to evaluate what are the consequences and possible effects of these changes. From the annotation perspective we need to: **i)** list possible affected annotations, and **ii)** list the possible changes to these annotations (steps 1 and 2 from [19]). For example, for manual changes, we need to list which are the annotations that also contain the same uncontrolled text as the new concept. This list is needed in order to know if these other annotations are also referring to the same concept, or they refer to another concept (homonyms) (which is not considered in [19]). For automatic changes, most of the algorithms use some correlation and/or clustering mechanism in order to derive the new concepts; this means that the new concepts will be already related to the uncontrolled annotations that caused the new concept to be defined, and, in the case of homonyms, two (or more) concepts must have been derived from the annotations, each of which will have a reference to the uncontrolled annotations that caused the concept to emerge. The work of [30] and [19] present frameworks to define evolution strategies, i.e., what to do in the presence of certain changes. For example, what to do when a new concept A more specific than B is added; should the annotations related to the concept B be re-checked to see whether A is a better concept to describe the annotation, or does this rule apply only if B was added as a consequence of a consensus. The work on [19] presents a more detailed architecture and model to list the changes and define the correction rules. The evolution strategy should be defined by each use case partner.

4. **User consensus**: once the change and the consequences are known, and before applying the changes, we need to define a mechanism by which we ensure these changes are commonly accepted by the affected users. The system could ask users whether this new concept is a correct representation of their understanding of the annotation when it was applied, or have a predefined set of conditions to define when the change is good enough to be incorporated into the controlled vocabulary automatically. In the case of multiple alternatives, a consensus model for decision making similar to [14] could be used as already mentioned in a previous Insemtives deliverable 2.2.1 [1].

5. **Change application**: once the change is accepted, we need to apply the changes into the controlled vocabulary. We need to keep track of all the changes (and the underlying cause) made to the controlled vocabulary in order to be able to revert these changes if necessary as proposed in [19].

6. **Change propagation**: once the changes are made effective in the controlled vocabulary, these changes need to be propagated to the affected annotations. Notice that at this stage the system already knows which are the affected annotations, and the users have already accepted or discarded (in the *consequence analysis* step) the effects of the propagation of the newly created concept, converting the uncontrolled annotations to controlled annotations. These changes should also be logged in order to keep track of which controlled annotations are the result of a consensus and evolution mechanism and also to be able to undo changes. This information is needed in the *consequence analysis* step.

3.4 Going beyond the state of the art

In this section we proposed a model to identify changes in a controlled vocabulary and to propagate these changes to the affected annotations (controlled and uncontrolled) in collaborative settings. The proposed model is based on state of the art models borrowed from ontology maturing research, semantic annotations evolution and collaborative systems, considering that individually none of them fully addressed our problem as defined in Section 3.1).

The new proposed model consists on 6 steps that **i)** capture the changes in the controlled vocabulary, **ii)** analyses the viability of these changes and **iii)** its effects on the existing annotations, **iv)** considering the user in the loop when necessary to check whether the propagation of the changes to the annotation reflects the initial intended meaning of the user's annotations, **v)** only then making effective the changes on the controlled vocabulary **vi)** with the consequent propagation to the annotations.

The novelty of our proposed model is to consider that the changes in the controlled vocabulary can be a consequence of a collaborative effort, therefore, the changes to be propagated to the annotations should also consider collaborative elements. As already mentioned in Section 2.4 different use cases require different ad-hoc solutions; therefore, the model proposed here is general enough to adapt to the needs of each use case, leaving each room for fine tuning according to the scenario of each partner.

4 Conclusions

Two prominently discussed topics in the Semantic Web research are the interlinking of content on the so-called Web of Data and the evolutions of both annotations and links (as a special form of annotations) in time. The main concerns of the former problem are how large data sets can (preferably automatically) be interlinked with other data sets on the Web to form a giant global interconnected graph. Means to do that are provided by interlinking algorithms, an active area of research. In this deliverable we reviewed existing techniques and propose a set of novel approaches applicable if the prerequisites for currently applied techniques are not fulfilled.

Besides that, we discuss the issue of (structured) annotation evolution in cases in which the controlled vocabularies, which are used in the annotations, change. The model proposed in this deliverable is based on ideas borrowed from ontology maturing research, semantic annotations evolution and collaborative systems.

Our next steps consist of proposing concrete models for interlinking applicable in the case studies which are capable of identifying things to interlink based on automated methods and which propose links to other external datasets to interlink with and to increase the amount of available annotations. For annotation evolution we aim to realize the method proposed in the second part of the document.

References

[1] Pierre Andrews, Ilya Zaihrayeu, and Juan Pane. *Insemtives Deliverable 2.2.1: Report on methods and algorithms for bootstrapping SemanticWeb content from user repositories and reaching consensus on the use of semantics*, 2009.

[2] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2009.

[3] Simone Braun, Claudiu Schora, and Valentin Zacharias. Semantics to the bookmarks: A review of social semantic bookmarking systems. In *International Conference on Semantic Systems (I-SEMANTICS 2009), Graz, Austria*, pages 445–454, 2009.

[4] Tobias Bürger and Michael Hausenblas. Interlinking Multimedia - Principles and Requirements. In *International Workshop on Interacting with Multimedia Content in the Social Semantic Web (IMC-SSW'08)*, pages 31–36, Koblenz, Germany, 2008.

[5] Tobias Burger, Ilya Zaihrayeu, Pierre Andrews, Denys Babenko, Juan Pane, and Borislav Popov. Insemtives deliverable 2.1.1: Report on the state-of-the-art and requirements for annotation representation models. Technical report, UIBK, UNITN, ONTOTEXT, 2009.

[6] Tobias Bürger, Ilya Zaihrayeu, Pierre Andrews, Denys Babenko, Juan Pane, and Borislav Popov. Report on the state-of-the-art and requirements for annotation representation models. INSEMTIVES Project Deliverable D2.1.1; available online: http://www.insemtives.eu, June 2009.

[7] Giorgos Flouris, Dimitris Manakanatas, Haridimos Kondylakis, Dimitris Plexousakis, and Grigoris Antoniou. Ontology change: Classification and survey. *Knowl. Eng. Rev.*, 23(2):117–152, 2008.

[8] Enrico Franconi, Fabio Grandi, and Federica Mandreoli. A semantic approach for schema evolution and versioning in Object-Oriented databases. In *Computational Logic CL 2000*, pages 1048–1062. 2000.

[9] Peter Haase, Andreas Hotho, Lars Schmidt-Thieme, and York Sure. Collaborative and Usage-Driven evolution of personal ontologies. In *The Semantic Web: Research and Applications*, pages 486–499. 2005.

[10] Wolgang Halb, Yves Raimond, and Michael Hausenblas. Building Linked Data For Both Humans and Machines. In *International Workshop on Linked Data on the Web (LDOW'08)*, Beijing, China, 2008.

[11] Michael Hausenblas, Wolfgang Halb, and Yves Raimond. Scripting User Contributed Interlinking. In *4^{th} Workshop on Scripting for the Semantic Web (SFSW'08)s*, Tenerife, Spain, 2008.

[12] Michael Hausenblas and Wolgang Halb. Interlinking Multimedia Data. In *Linking Open Data Triplification Challenge at the International Conference on Semantic Systems (I-Semantics'08)*, 2008. http://triplify.org/Challenge/Nominations?v=51f.

[13] Michael Hausenblas, Raphael Troncy, Yves Raimond, and Tobias Bürger. Interlinking Multimedia: How to Apply Linked Data Principles to Multimedia Fragments. In *WWW 2009 Workshop: Linked Data on the Web (LDOW2009)*, Madrid, Spain, 2009.

[14] E. Herrera-Viedma, F. Herrera, and F. Chiclana. A consensus model for multiperson decision making with different preference structures. *IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS PART A SYSTEMS AND HUMANS*, 32(3):392–402, 2002.

[15] Georgi Kobilarov, Tom Scott, Yves Raimond, Silver Oliver, Chris Sizemore, Michael Smethurst, Christian Bizer, and Robert Lee. Media meets semantic web — how the bbc uses dbpedia and linked data to make connections. In *ESWC 2009 Heraklion: Proceedings of the 6th European Semantic Web Conference on The Semantic Web*, pages 723–737, Berlin, Heidelberg, 2009. Springer-Verlag.

[16] Markus Krötzsch, Denny Vrandecic, Max Völkel, Heiko Haller, and Rudi Studer. Semantic Wikipedia. *Journal of Web Semantics*, 5:251–261, 2007.

[17] Y. Liang, H. Alani, and N R Shadbolt. Change management: The core task of ontology versioning and evolution. *IN PROCEEDINGS OF POSTGRADUATE RESEARCH CONFERENCE IN ELECTRONICS, PHOTONICS, COMMUNICATIONS AND NETWORKS, AND COMPUTING SCIENCE 2005 LANCASTER, UNITED KINGDOM. (PREP 2005*, pages 221—222.

[18] Yaozhong Liang, Harith Alani, and Nigel Shadbolt. Ontology versioning and evolution for semantic Web-Based applications. 9-month progress report. http://eprints.ecs.soton.ac.uk/13067/, July 2005.

[19] P.-H. Luong and R. Dieng-Kuntz. A RULE-BASED APPROACH FOR SEMANTIC ANNOTATION EVOLUTION. *Computational Intelligence*, 23(3):320–338, 2007.

[20] Lyndon Nixon and Elena Simperl. Makna and MultiMakna: towards semantic and multimedia capability in wikis for the emerging web. In *Semantics'06*, Vienna, Austria, 2006.

[21] Natalya F Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. *KNOWLEDGE AND INFORMATION SYSTEMS*, 6:428—440, 2003.

[22] A Passant and P Laublet. Meaning of a tag: A collaborative approach to bridge the gap between tagging and linked data. *Proceedings of the WWW 2008 Workshop Linked Data on the Web (LDOW2008), Beijing, China, Apr*, 2008.

[23] Silvia Pfeiffer, Conrad Parker, and A. Pang. Specifying time intervals in URI queries and fragments of time-based Web resources. `http://www.annodex.net/TR/draft-pfeiffer-temporal-fragments-03.html`, 2005.

[24] Silvia Pfeiffer, Conrad Parker, and A. Pang. The Continuous Media Markup Language (CMML), Version 2.1. `http://www.annodex.net/TR/draft-pfeiffer-cmml-03.html`, 2006.

[25] Peter Plessers and Olga De Troyer. Resolving inconsistencies in evolving ontologies. In *The Semantic Web: Research and Applications*, pages 200–214. 2006.

[26] Niko Popitsch, Bernhard Schandl, A. Amiri, S. Leitich, and W. Jochum. Ylvi - Multimediaizing the Semantic Wiki. In *1st International Workshop on Semantic Wikis (SemWiki'06)*, 2006.

[27] Yves Raimond and Marc Sandler. A Web of Musical Information. In *9th International Conference on Music Information Retrieval (ISMIR'08)*, Philadelphia, USA, 2008.

[28] Yves Raimond, Chris Sutton, and Marc Sandler. Automatic Interlinking of Music Datasets on the Semantic Web. In *International Workshop on Linked Data on the Web (LDOW'08)*, Beijing, China, 2008.

[29] Sebastian Schaffert, Joachim Baumeister, Francois Bry, and Malte Kiesel. Semantic Wikis. *IEEE Software*, 25(4):8–11, 2008.

[30] Ljiljana Stojanovic, Alexander Maedche, Boris Motik, and Nenad Stojanovic. User-Driven ontology evolution management. In *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, pages 133–140. 2002.

[31] Ljiljana Stojanovic, Alexander Maedche, Nenad Stojanovic, and Rudi Studer. Ontology evolution as reconfiguration-design problem solving. In *Proceedings of the 2nd international conference on Knowledge capture*, pages 162–171, Sanibel Island, FL, USA, 2003. ACM.

[32] Julius Volz, Chris Bizer, Martin Gaedke, and Georgi Kobilarov. Discovering and maintaining links on the web of data. In *Proceedings of the 8th International Semantic Web Conference (ISWC2009)*, October 2009.

[33] Luis von Ahn. Games with a Purpose. *IEEE Computer*, 39(6):92–94, 2006.

[34] Thomas Vander Wal. Folksonomy: Coinage and definition. http://www.vanderwal.net/folksonomy.html.

[end of document]